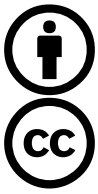
# THE OPEN LOGIC TEXT

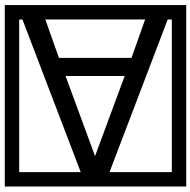
Complete Build

**Open Logic Project**

Revision: 89307f1 (master)
2019-02-27



*The Open Logic Text* by the Open Logic Project is licensed under a Creative Commons Attribution 4.0 International License.



# About the Open Logic Project

The *Open Logic Text* is an open-source, collaborative textbook of formal metalogic and formal methods, starting at an intermediate level (i.e., after an introductory formal logic course). Though aimed at a non-mathematical audience (in particular, students of philosophy and computer science), it is rigorous.

The *Open Logic Text* is a collaborative project and is under active development. Coverage of some topics currently included may not yet be complete, and many sections still require substantial revision. We plan to expand the text to cover more topics in the future. We also plan to add features to the text, such as a glossary, a list of further reading, historical notes, pictures, better explanations, sections explaining the relevance of results to philosophy, computer science, and mathematics, and more problems and examples. If you find an error, or have a suggestion, please let the project team know.

The project operates in the spirit of open source. Not only is the text freely available, we provide the LaTeX source under the Creative Commons Attribution license, which gives anyone the right to download, use, modify, rearrange, convert, and re-distribute our work, as long as they give appropriate credit.

Please see the Open Logic Project website at openlogicproject.org for additional information.

# Contents


**I Sets, Relations, Functions** **18**

**1 Sets** **20**
1.1 Basics . . . 20
1.2 Some Important Sets . . . 21
1.3 Subsets . . . 22
1.4 Unions and Intersections . . . 23
1.5 Pairs, Tuples, Cartesian Products . . . 25
1.6 Russell's Paradox . . . 27
Problems . . . 27

**2 Relations** **29**
2.1 Relations as Sets . . . 29
2.2 Special Properties of Relations . . . 30
2.3 Orders . . . 31
2.4 Graphs . . . 33
2.5 Operations on Relations . . . 34
Problems . . . 35

**3 Functions** **36**
3.1 Basics . . . 36
3.2 Kinds of Functions . . . 38
3.3 Inverses of Functions . . . 39
3.4 Composition of Functions . . . 40
3.5 Isomorphism . . . 41
3.6 Partial Functions . . . 41
3.7 Functions and Relations . . . 42
Problems . . . 42

**4 The Size of Sets** **43**
4.1 Introduction . . . 43
4.2 Enumerable Sets . . . 43
4.3 Non-enumerable Sets . . . 49
4.4 Reduction . . . 52

4.5 Equinumerous Sets . . . 53
4.6 Comparing Sizes of Sets . . . 54
Problems . . . 55

**II Propositional Logic 58**

**5 Syntax and Semantics 60**
5.1 Introduction . . . 60
5.2 Propositional Formulas . . . 61
5.3 Preliminaries . . . 63
5.4 Valuations and Satisfaction . . . 64
5.5 Semantic Notions . . . 65
Problems . . . 66

**6 Derivation Systems 67**
6.1 Introduction . . . 67
6.2 The Sequent Calculus . . . 69
6.3 Natural Deduction . . . 69
6.4 Tableaux . . . 71
6.5 Axiomatic Derivations . . . 72

**7 The Sequent Calculus 74**
7.1 Rules and Derivations . . . 74
7.2 Propositional Rules . . . 75
7.3 Structural Rules . . . 76
7.4 Derivations . . . 77
7.5 Examples of Derivations . . . 78
7.6 Proof-Theoretic Notions . . . 82
7.7 Derivability and Consistency . . . 84
7.8 Derivability and the Propositional Connectives . . . 85
7.9 Soundness . . . 86
Problems . . . 90

**8 Natural Deduction 91**
8.1 Rules and Derivations . . . 91
8.2 Propositional Rules . . . 92
8.3 Derivations . . . 93
8.4 Examples of Derivations . . . 94
8.5 Proof-Theoretic Notions . . . 98
8.6 Derivability and Consistency . . . 99
8.7 Derivability and the Propositional Connectives . . . 101
8.8 Soundness . . . 102
Problems . . . 105

**9 Tableaux** **106**
9.1 Rules and Tableaux . . . 106
9.2 Propositional Rules . . . 107
9.3 Tableaux . . . 108
9.4 Examples of Tableaux . . . 109
9.5 Proof-Theoretic Notions . . . 113
9.6 Derivability and Consistency . . . 115
9.7 Derivability and the Propositional Connectives . . . 117
9.8 Soundness . . . 119
Problems . . . 122

**10 Axiomatic Derivations** **123**
10.1 Rules and Derivations . . . 123
10.2 Axiom and Rules for the Propositional Connectives . . . 125
10.3 Examples of Derivations . . . 125
10.4 Proof-Theoretic Notions . . . 127
10.5 The Deduction Theorem . . . 128
10.6 Derivability and Consistency . . . 130
10.7 Derivability and the Propositional Connectives . . . 131
10.8 Soundness . . . 131
Problems . . . 132

**11 The Completeness Theorem** **133**
11.1 Introduction . . . 133
11.2 Outline of the Proof . . . 134
11.3 Complete Consistent Sets of Sentences . . . 135
11.4 Lindenbaum's Lemma . . . 136
11.5 Construction of a Model . . . 137
11.6 The Completeness Theorem . . . 138
11.7 The Compactness Theorem . . . 138
11.8 A Direct Proof of the Compactness Theorem . . . 139
Problems . . . 140

**III First-order Logic** **141**

**12 Syntax and Semantics** **143**
12.1 Introduction . . . 143
12.2 First-Order Languages . . . 144
12.3 Terms and Formulas . . . 146
12.4 Unique Readability . . . 148
12.5 Main operator of a Formula . . . 150
12.6 Subformulas . . . 151
12.7 Free Variables and Sentences . . . 152

12.8 Substitution . . . 153
12.9 Structures for First-order Languages . . . 155
12.10 Covered Structures for First-order Languages . . . 156
12.11 Satisfaction of a Formula in a Structure . . . 157
12.12 Variable Assignments . . . 161
12.13 Extensionality . . . 164
12.14 Semantic Notions . . . 165
Problems . . . 167

**13 Theories and Their Models** **169**
13.1 Introduction . . . 169
13.2 Expressing Properties of Structures . . . 171
13.3 Examples of First-Order Theories . . . 171
13.4 Expressing Relations in a Structure . . . 174
13.5 The Theory of Sets . . . 175
13.6 Expressing the Size of Structures . . . 177
Problems . . . 178

**14 Derivation Systems** **180**
14.1 Introduction . . . 180
14.2 The Sequent Calculus . . . 182
14.3 Natural Deduction . . . 182
14.4 Tableaux . . . 184
14.5 Axiomatic Derivations . . . 185

**15 The Sequent Calculus** **187**
15.1 Rules and Derivations . . . 187
15.2 Propositional Rules . . . 188
15.3 Quantifier Rules . . . 189
15.4 Structural Rules . . . 190
15.5 Derivations . . . 190
15.6 Examples of Derivations . . . 192
15.7 Derivations with Quantifiers . . . 196
15.8 Proof-Theoretic Notions . . . 197
15.9 Derivability and Consistency . . . 199
15.10 Derivability and the Propositional Connectives . . . 200
15.11 Derivability and the Quantifiers . . . 201
15.12 Soundness . . . 202
15.13 Derivations with Identity predicate . . . 207
15.14 Soundness with Identity predicate . . . 208
Problems . . . 208

**16 Natural Deduction** **209**
16.1 Rules and Derivations . . . 209

16.2 Propositional Rules . . . 210
16.3 Quantifier Rules . . . 211
16.4 Derivations . . . 212
16.5 Examples of Derivations . . . 213
16.6 Derivations with Quantifiers . . . 217
16.7 Proof-Theoretic Notions . . . 220
16.8 Derivability and Consistency . . . 222
16.9 Derivability and the Propositional Connectives . . . 223
16.10 Derivability and the Quantifiers . . . 225
16.11 Soundness . . . 225
16.12 Derivations with Identity predicate . . . 229
16.13 Soundness with Identity predicate . . . 231
Problems . . . 231

**17 Tableaux** **233**
17.1 Rules and Tableaux . . . 233
17.2 Propositional Rules . . . 234
17.3 Quantifier Rules . . . 235
17.4 Tableaux . . . 236
17.5 Examples of Tableaux . . . 237
17.6 Tableaux with Quantifiers . . . 241
17.7 Proof-Theoretic Notions . . . 245
17.8 Derivability and Consistency . . . 247
17.9 Derivability and the Propositional Connectives . . . 248
17.10 Derivability and the Quantifiers . . . 251
17.11 Soundness . . . 252
17.12 Tableaux with Identity predicate . . . 254
17.13 Soundness with Identity predicate . . . 255
Problems . . . 256

**18 Axiomatic Derivations** **257**
18.1 Rules and Derivations . . . 257
18.2 Axiom and Rules for the Propositional Connectives . . . 259
18.3 Axioms and Rules for Quantifiers . . . 259
18.4 Examples of Derivations . . . 260
18.5 Derivations with Quantifiers . . . 261
18.6 Proof-Theoretic Notions . . . 262
18.7 The Deduction Theorem . . . 263
18.8 The Deduction Theorem with Quantifiers . . . 265
18.9 Derivability and Consistency . . . 266
18.10 Derivability and the Propositional Connectives . . . 267
18.11 Derivability and the Quantifiers . . . 267
18.12 Soundness . . . 268
18.13 Derivations with Identity predicate . . . 269

Problems . . . 270

**19 The Completeness Theorem** **271**
19.1 Introduction . . . 271
19.2 Outline of the Proof . . . 272
19.3 Complete Consistent Sets of Sentences . . . 274
19.4 Henkin Expansion . . . 275
19.5 Lindenbaum's Lemma . . . 277
19.6 Construction of a Model . . . 278
19.7 Identity . . . 280
19.8 The Completeness Theorem . . . 282
19.9 The Compactness Theorem . . . 283
19.10 A Direct Proof of the Compactness Theorem . . . 285
19.11 The Löwenheim-Skolem Theorem . . . 286
Problems . . . 287

**20 Beyond First-order Logic** **288**
20.1 Overview . . . 288
20.2 Many-Sorted Logic . . . 289
20.3 Second-Order logic . . . 290
20.4 Higher-Order logic . . . 294
20.5 Intuitionistic Logic . . . 296
20.6 Modal Logics . . . 300
20.7 Other Logics . . . 302

**IV Model Theory** **303**

**21 Basics of Model Theory** **305**
21.1 Reducts and Expansions . . . 305
21.2 Substructures . . . 306
21.3 Overspill . . . 306
21.4 Isomorphic Structures . . . 307
21.5 The Theory of a Structure . . . 308
21.6 Partial Isomorphisms . . . 309
21.7 Dense Linear Orders . . . 312
Problems . . . 313

**22 Models of Arithmetic** **314**
22.1 Introduction . . . 314
22.2 Standard Models of Arithmetic . . . 315
22.3 Non-Standard Models . . . 317
22.4 Models of **Q** . . . 318
22.5 Computable Models of Arithmetic . . . 320

Problems . . . 322

**23 The Interpolation Theorem** **324**
23.1 Introduction . . . 324
23.2 Separation of Sentences . . . 324
23.3 Craig's Interpolation Theorem . . . 326
23.4 The Definability Theorem . . . 328

**24 Lindström's Theorem** **331**
24.1 Introduction . . . 331
24.2 Abstract Logics . . . 331
24.3 Compactness and Löwenheim-Skolem Properties . . . 333
24.4 Lindström's Theorem . . . 334

**V Computability** **337**

**25 Recursive Functions** **339**
25.1 Introduction . . . 339
25.2 Primitive Recursion . . . 340
25.3 Composition . . . 342
25.4 Primitive Recursion Functions . . . 343
25.5 Primitive Recursion Notations . . . 346
25.6 Primitive Recursive Functions are Computable . . . 346
25.7 Examples of Primitive Recursive Functions . . . 347
25.8 Primitive Recursive Relations . . . 350
25.9 Bounded Minimization . . . 352
25.10 Primes . . . 352
25.11 Sequences . . . 353
25.12 Trees . . . 356
25.13 Other Recursions . . . 357
25.14 Non-Primitive Recursive Functions . . . 358
25.15 Partial Recursive Functions . . . 360
25.16 The Normal Form Theorem . . . 361
25.17 The Halting Problem . . . 362
25.18 General Recursive Functions . . . 363
Problems . . . 364

**26 The Lambda Calculus** **365**
26.1 Introduction . . . 365
26.2 The Syntax of the Lambda Calculus . . . 366
26.3 Reduction of Lambda Terms . . . 367
26.4 The Church-Rosser Property . . . 368
26.5 Representability by Lambda Terms . . . 369

26.6 Lambda Representable Functions are Computable . . . . . 369
26.7 Computable Functions are Lambda Representable . . . . . 370
26.8 The Basic Primitive Recursive Functions are Lambda Representable . . . . . 370
26.9 Lambda Representable Functions Closed under Composition . 371
26.10 Lambda Representable Functions Closed under Primitive Recursion . . . . . 371
26.11 Fixed-Point Combinators . . . . . 373
26.12 Lambda Representable Functions Closed under Minimization . 374

**27 Computability Theory** **376**
27.1 Introduction . . . . . 376
27.2 Coding Computations . . . . . 377
27.3 The Normal Form Theorem . . . . . 378
27.4 The *s-m-n* Theorem . . . . . 379
27.5 The Universal Partial Computable Function . . . . . 379
27.6 No Universal Computable Function . . . . . 380
27.7 The Halting Problem . . . . . 380
27.8 Comparison with Russell's Paradox . . . . . 381
27.9 Computable Sets . . . . . 383
27.10 Computably Enumerable Sets . . . . . 383
27.11 Definitions of C. E. Sets . . . . . 384
27.12 Union and Intersection of C.E. Sets . . . . . 386
27.13 Computably Enumerable Sets not Closed under Complement . 387
27.14 Reducibility . . . . . 388
27.15 Properties of Reducibility . . . . . 389
27.16 Complete Computably Enumerable Sets . . . . . 390
27.17 An Example of Reducibility . . . . . 391
27.18 Totality is Undecidable . . . . . 392
27.19 Rice's Theorem . . . . . 393
27.20 The Fixed-Point Theorem . . . . . 395
27.21 Applying the Fixed-Point Theorem . . . . . 398
27.22 Defining Functions using Self-Reference . . . . . 399
27.23 Minimization with Lambda Terms . . . . . 400
Problems . . . . . 401

**VI Turing Machines** **402**

**28 Turing Machine Computations** **404**
28.1 Introduction . . . . . 404
28.2 Representing Turing Machines . . . . . 406
28.3 Turing Machines . . . . . 410
28.4 Configurations and Computations . . . . . 410

28.5 Unary Representation of Numbers . . . 412
28.6 Halting States . . . 413
28.7 Combining Turing Machines . . . 414
28.8 Variants of Turing Machines . . . 416
28.9 The Church-Turing Thesis . . . 417
Problems . . . 418

**29 Undecidability** **420**
29.1 Introduction . . . 420
29.2 Enumerating Turing Machines . . . 422
29.3 The Halting Problem . . . 422
29.4 The Decision Problem . . . 424
29.5 Representing Turing Machines . . . 425
29.6 Verifying the Representation . . . 428
29.7 The Decision Problem is Unsolvable . . . 433
Problems . . . 433

**VII Incompleteness** **435**

**30 Introduction to Incompleteness** **437**
30.1 Historical Background . . . 437
30.2 Definitions . . . 441
30.3 Overview of Incompleteness Results . . . 445
30.4 Undecidability and Incompleteness . . . 447
Problems . . . 449

**31 Arithmetization of Syntax** **450**
31.1 Introduction . . . 450
31.2 Coding Symbols . . . 451
31.3 Coding Terms . . . 453
31.4 Coding Formulas . . . 455
31.5 Substitution . . . 456
31.6 Derivations in **LK** . . . 456
31.7 Derivations in Natural Deduction . . . 460
31.8 Axiomatic Derivations . . . 464
Problems . . . 468

**32 Representability in Q** **469**
32.1 Introduction . . . 469
32.2 Functions Representable in **Q** are Computable . . . 471
32.3 The Beta Function Lemma . . . 472
32.4 Simulating Primitive Recursion . . . 475
32.5 Basic Functions are Representable in **Q** . . . 476

32.6 Composition is Representable in **Q** . . . 479
32.7 Regular Minimization is Representable in **Q** . . . 480
32.8 Computable Functions are Representable in **Q** . . . 483
32.9 Representing Relations . . . 484
32.10 Undecidability . . . 484
Problems . . . 485

**33 Theories and Computability** **487**
33.1 Introduction . . . 487
33.2 **Q** is C.e.-Complete . . . 487
33.3 $\omega$-Consistent Extensions of **Q** are Undecidable . . . 488
33.4 Consistent Extensions of **Q** are Undecidable . . . 489
33.5 Axiomatizable Theories . . . 490
33.6 Axiomatizable Complete Theories are Decidable . . . 490
33.7 **Q** has no Complete, Consistent, Axiomatizable Extensions . . . 490
33.8 Sentences Provable and Refutable in **Q** are Computably Inseparable . . . 491
33.9 Theories Consistent with **Q** are Undecidable . . . 492
33.10 Theories in which **Q** is Intepretable are Undecidable . . . 492

**34 Incompleteness and Provability** **494**
34.1 Introduction . . . 494
34.2 The Fixed-Point Lemma . . . 495
34.3 The First Incompleteness Theorem . . . 497
34.4 Rosser's Theorem . . . 499
34.5 Comparison with Gödel's Original Paper . . . 500
34.6 The Derivability Conditions for **PA** . . . 501
34.7 The Second Incompleteness Theorem . . . 502
34.8 Löb's Theorem . . . 504
34.9 The Undefinability of Truth . . . 507
Problems . . . 508

**VIII Second-order Logic** **509**

**35 Syntax and Semantics** **511**
35.1 Introduction . . . 511
35.2 Terms and Formulas . . . 512
35.3 Satisfaction . . . 513
35.4 Semantic Notions . . . 515
35.5 Expressive Power . . . 515
35.6 Describing Infinite and Enumerable Domains . . . 516
Problems . . . 518

**36 Metatheory of Second-order Logic** **519**
36.1 Introduction . . . 519
36.2 Second-order Arithmetic . . . 520
36.3 Second-order Logic is not Axiomatizable . . . 521
36.4 Second-order Logic is not Compact . . . 522
36.5 The Löwenheim-Skolem Theorem Fails for Second-order Logic 522
Problems . . . 523

**37 Second-order Logic and Set Theory** **524**
37.1 Introduction . . . 524
37.2 Comparing Sets . . . 524
37.3 Cardinalities of Sets . . . 525
37.4 The Power of the Continuum . . . 526

**IX Normal Modal Logics** **529**

**38 Syntax and Semantics of Normal Modal Logics** **531**
38.1 Introduction . . . 531
38.2 The Language of Basic Modal Logic . . . 532
38.3 Simultaneous Substitution . . . 533
38.4 Relational Models . . . 534
38.5 Truth at a World . . . 535
38.6 Truth in a Model . . . 536
38.7 Validity . . . 537
38.8 Tautological Instances . . . 538
38.9 Schemas and Validity . . . 539
38.10 Entailment . . . 540
Problems . . . 542

**39 Frame Definability** **545**
39.1 Introduction . . . 545
39.2 Properties of Accessibility Relations . . . 546
39.3 Frames . . . 547
39.4 Frame Definability . . . 548
39.5 First-order Definability . . . 551
39.6 Equivalence Relations and **S5** . . . 552
39.7 Second-order Definability . . . 554
Problems . . . 556

**40 Axiomatic Derivations** **557**
40.1 Introduction . . . 557
40.2 Normal Modal Logics . . . 558
40.3 Derivations and Modal Systems . . . 560

40.4 Proofs in **K** . . . 561
40.5 Derived Rules . . . 563
40.6 More Proofs in **K** . . . 565
40.7 Dual Formulas . . . 566
40.8 Proofs in Modal Systems . . . 566
40.9 Soundness . . . 568
40.10 Showing Systems are Distinct . . . 568
40.11 Derivability from a Set of Formulas . . . 570
40.12 Properties of Derivability . . . 570
40.13 Consistency . . . 571
Problems . . . 571

**41 Completeness and Canonical Models** **573**
41.1 Introduction . . . 573
41.2 Complete Σ-Consistent Sets . . . 574
41.3 Lindenbaum's Lemma . . . 575
41.4 Modalities and Complete Consistent Sets . . . 576
41.5 Canonical Models . . . 578
41.6 The Truth Lemma . . . 579
41.7 Determination and Completeness for **K** . . . 580
41.8 Frame Completeness . . . 580
Problems . . . 583

**42 Filtrations and Decidability** **584**
42.1 Introduction . . . 584
42.2 Preliminaries . . . 586
42.3 Filtrations . . . 587
42.4 Examples of Filtrations . . . 589
42.5 Filtrations are Finite . . . 590
42.6 **K** and **S5** have the Finite Model Property . . . 591
42.7 **S5** is Decidable . . . 592
42.8 Filtrations and Properties of Accessibility . . . 592
42.9 Filtrations of Euclidean Models . . . 593
Problems . . . 595

**43 Modal Tableaux** **597**
43.1 Introduction . . . 597
43.2 Rules for K . . . 598
43.3 Tableaux for **K** . . . 600
43.4 Soundness . . . 601
43.5 Rules for Other Accessibility Relations . . . 603
43.6 Tableaux for Other Logics . . . 605
43.7 Soundness for Additional Rules . . . 605
43.8 Simple Tableaux for **S5** . . . 606

43.9 Completeness for **K** . . . 607
43.10 Countermodels from Tableaux . . . 609
Problems . . . 611

**X Intuitionistic Logic 613**

**44 Introduction 615**
44.1 Constructive Reasoning . . . 615
44.2 Syntax of Intuitionistic Logic . . . 616
44.3 The Brouwer-Heyting-Kolmogorov Interpretation . . . 617
44.4 Natural Deduction . . . 620
44.5 Axiomatic Derivations . . . 623
Problems . . . 624

**45 Semantics 625**
45.1 Introduction . . . 625
45.2 Relational models . . . 626
45.3 Semantic Notions . . . 627
45.4 Topological Semantics . . . 627
Problems . . . 629

**46 Soundness and Completeness 630**
46.1 Soundness of Axiomatic Derivations . . . 630
46.2 Soundness of Natural Deduction . . . 631
46.3 Lindenbaum's Lemma . . . 632
46.4 The Canonical Model . . . 634
46.5 The Truth Lemma . . . 634
46.6 The Completeness Theorem . . . 635
Problems . . . 635

**47 Propositions as Types 636**
47.1 Introduction . . . 636
47.2 Sequent Natural Deduction . . . 638
47.3 Proof Terms . . . 639
47.4 Converting Derivations to Proof Terms . . . 640
47.5 Recovering Derivations from Proof Terms . . . 643
47.6 Reduction . . . 645
47.7 Normalization . . . 647

**XI Counterfactuals 651**

**48 Introduction 652**
48.1 The Material Conditional . . . 652

48.2 Paradoxes of the Material Conditional . . . 653
48.3 The Strict Conditional . . . 654
48.4 Counterfactuals . . . 656
Problems . . . 657

**49 Minimal Change Semantics** **659**
49.1 Introduction . . . 659
49.2 Sphere Models . . . 660
49.3 Truth and Falsity of Counterfactuals . . . 662
49.4 Antecedent Strengthenng . . . 663
49.5 Transitivity . . . 664
49.6 Contraposition . . . 665
Problems . . . 666

**XII Methods** **668**

**50 Proofs** **670**
50.1 Introduction . . . 670
50.2 Starting a Proof . . . 671
50.3 Using Definitions . . . 671
50.4 Inference Patterns . . . 673
50.5 An Example . . . 679
50.6 Another Example . . . 682
50.7 Proof by Contradiction . . . 683
50.8 Reading Proofs . . . 687
50.9 I Can't Do It! . . . 688
50.10 Other Resources . . . 689
Problems . . . 690

**51 Induction** **691**
51.1 Introduction . . . 691
51.2 Induction on $\mathbb{N}$ . . . 692
51.3 Strong Induction . . . 694
51.4 Inductive Definitions . . . 695
51.5 Structural Induction . . . 697
51.6 Relations and Functions . . . 698
Problems . . . 701

**XIII History** **702**

**52 Biographies** **703**
52.1 Georg Cantor . . . 703
52.2 Alonzo Church . . . 704

52.3 Gerhard Gentzen . . . . . 705
52.4 Kurt Gödel . . . . . 706
52.5 Emmy Noether . . . . . 707
52.6 Rózsa Péter . . . . . 709
52.7 Julia Robinson . . . . . 710
52.8 Bertrand Russell . . . . . 712
52.9 Alfred Tarski . . . . . 713
52.10 Alan Turing . . . . . 714
52.11 Ernst Zermelo . . . . . 715

**Photo Credits** **717**

**Bibliography** **719**

This file loads all content included in the Open Logic Project. Editorial notes like this, if displayed, indicate that the file was compiled without any thought to how this material will be presented. If you can read this, it is probably *not advisable* to teach or study from this PDF.

The Open Logic Project provides many mechanisms by which a text can be generate which is more appropriate for teaching or self-study. For instance, by default, the text will make all logical operators primitives and carry out all cases for all operators in proofs. But it is much better to leave some of these cases as exercises. The Open Logic Project is also a work in progress. In an effort to stimulate collaboration and improvemenent, material is included even if it is ony in draft form, is missing exercises, etc. A PDF produced for a course will exclude these sections.

To find PDFs more suitable for teaching and studying, have a look at the sample courses available on the OLP website. To make your own, you might start from the sample driver file or look at the sources of the derived textbooks for more fancy and advanced examples.

# Part I

# Sets, Relations, Functions

The material in this part is a reasonably complete introduction to basic naive set theory. Unless students can be assumed to have this background, it's probably advisable to start a course with a review of this material, at least the part on sets, functions, and relations. This should ensure that all students have the basic facility with mathematical notation required for any of the other logical sections. NB: This part does not cover induction directly.

The presentation here would benefit from additional examples, especially, "real life" examples of relations of interest to the audience.

It is planned to expand this part to cover naive set theory more extensively.

# Chapter 1

# Sets

## 1.1 Basics

Sets are the most fundamental building blocks of mathematical objects. In fact, almost every mathematical object can be seen as a set of some kind. In logic, as in other parts of mathematics, sets and set-theoretical talk is ubiquitous. So it will be important to discuss what sets are, and introduce the notations necessary to talk about sets and operations on sets in a standard way.

**Definition 1.1** (Set)**.** A *set* is a collection of objects, considered independently of the way it is specified, of the order of the objects in the set, or of their multiplicity. The objects making up the set are called *elements* or *members* of the set. If $a$ is an element of a set $X$, we write $a \in X$ (otherwise, $a \notin X$). The set which has no elements is called the *empty* set and denoted by the symbol $\emptyset$.

**Example 1.2.** Whenever you have a bunch of objects, you can collect them together in a set. The set of Richard's siblings, for instance, is a set that contains one person, and we could write it as $S = \{\text{Ruth}\}$. In general, when we have some objects $a_1, \ldots, a_n$, then the set consisting of exactly those objects is written $\{a_1, \ldots, a_n\}$. Frequently we'll specify a set by some property that its elements share—as we just did, for instance, by specifying $S$ as the set of Richard's siblings. We'll use the following shorthand notation for that: $\{x : \ldots x \ldots\}$, where the $\ldots x \ldots$ stands for the property that $x$ has to have in order to be counted among the elements of the set. In our example, we could have specified $S$ also as

$$S = \{x : x \text{ is a sibling of Richard}\}.$$

When we say that sets are independent of the way they are specified, we mean that the elements of a set are all that matters. For instance, it so happens

that

$$\{\text{Nicole, Jacob}\},$$
$$\{x : \text{is a niece or nephew of Richard}\}, \text{ and}$$
$$\{x : \text{is a child of Ruth}\}$$

are three ways of specifying one and the same set.

Saying that sets are considered independently of the order of their elements and their multiplicity is a fancy way of saying that

$$\{\text{Nicole, Jacob}\} \text{ and}$$
$$\{\text{Jacob, Nicole}\}$$

are two ways of specifying the same set; and that

$$\{\text{Nicole, Jacob}\} \text{ and}$$
$$\{\text{Jacob, Nicole, Nicole}\}$$

are also two ways of specifying the same set. In other words, all that matters is which elements a set has. The elements of a set are not ordered and each element occurs only once. When we *specify* or *describe* a set, elements may occur multiple times and in different orders, but any descriptions that only differ in the order of elements or in how many times elements are listed describes the same set.

**Definition 1.3** (Extensionality)**.** If $X$ and $Y$ are sets, then $X$ and $Y$ are *identical*, $X = Y$, iff every element of $X$ is also an element of $Y$, and vice versa.

Extensionality gives us a way for showing that sets are identical: to show that $X = Y$, show that whenever $x \in X$ then also $x \in Y$, and whenever $y \in Y$ then also $y \in X$.

## 1.2 Some Important Sets

**Example 1.4.** Mostly we'll be dealing with sets that have mathematical objects as members. You will remember the various sets of numbers: $\mathbb{N}$ is the set of *natural* numbers $\{0, 1, 2, 3, \dots\}$; $\mathbb{Z}$ the set of *integers*,

$$\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\};$$

$\mathbb{Q}$ the set of *rational* numbers ($\mathbb{Q} = \{z/n : z \in \mathbb{Z}, n \in \mathbb{N}, n \neq 0\}$); and $\mathbb{R}$ the set of *real* numbers. These are all *infinite* sets, that is, they each have infinitely many elements. As it turns out, $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ have the same number of elements, while $\mathbb{R}$ has a whole bunch more—$\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ are "enumerable and infinite" whereas $\mathbb{R}$ is "non-enumerable".

We'll sometimes also use the set of positive integers $\mathbb{Z}^+ = \{1, 2, 3, \dots\}$ and the set containing just the first two natural numbers $\mathbb{B} = \{0, 1\}$.

**Example 1.5** (Strings)**.** Another interesting example is the set $A^*$ of *finite strings* over an alphabet $A$: any finite sequence of elements of $A$ is a string over $A$. We include the *empty string* $\Lambda$ among the strings over $A$, for every alphabet $A$. For instance,

$$\mathbb{B}^* = \{\Lambda, 0, 1, 00, 01, 10, 11, \\ 000, 001, 010, 011, 100, 101, 110, 111, 0000, \ldots\}.$$

If $x = x_1 \ldots x_n \in A^*$is a string consisting of $n$ "letters" from $A$, then we say *length* of the string is $n$ and write $\text{len}(x) = n$.

**Example 1.6** (Infinite sequences)**.** For any set $A$ we may also consider the set $A^\omega$ of infinite sequences of elements of $A$. An infinite sequence $a_1a_2a_3a_4\ldots$ consists of a one-way infinite list of objects, each one of which is an element of $A$.

## 1.3 Subsets

Sets are made up of their elements, and every element of a set is a part of that set. But there is also a sense that some of the elements of a set *taken together* are a "part of" that set. For instance, the number 2 is part of the set of integers, but the set of even numbers is also a part of the set of integers. It's important to keep those two senses of being part of a set separate.

**Definition 1.7** (Subset)**.** If every element of a set $X$ is also an element of $Y$, then we say that $X$ is a *subset* of $Y$, and write $X \subseteq Y$.

**Example 1.8.** First of all, every set is a subset of itself, and $\emptyset$ is a subset of every set. The set of even numbers is a subset of the set of natural numbers. Also, $\{a, b\} \subseteq \{a, b, c\}$.

But $\{a, b, e\}$ is not a subset of $\{a, b, c\}$.

Note that a set may contain other sets, not just as subsets but as elements! In particular, a set may happen to *both* be an element and a subset of another, e.g., $\{0\} \in \{0, \{0\}\}$ and also $\{0\} \subseteq \{0, \{0\}\}$.

Extensionality gives a criterion of identity for sets: $X = Y$ iff every element of $X$ is also an element of $Y$ and vice versa. The definition of "subset" defines $X \subseteq Y$ precisely as the first half of this criterion: every element of $X$ is also an element of $Y$. Of course the definition also applies if we switch $X$ and $Y$: $Y \subseteq X$ iff every element of $Y$ is also an element of $X$. And that, in turn, is exactly the "vice versa" part of extensionality. In other words, extensionality amounts to: $X = Y$ iff $X \subseteq Y$ and $Y \subseteq X$.

**Definition 1.9** (Power Set)**.** The set consisting of all subsets of a set $X$ is called the *power set of* $X$, written $\wp(X)$.

$$\wp(X) = \{Y : Y \subseteq X\}$$

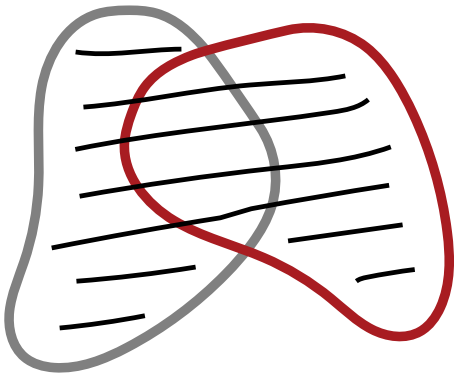


Figure 1.1: The union $X \cup Y$ of two sets is set of elements of $X$ together with those of $Y$.

**Example 1.10.** What are all the possible subsets of $\{a, b, c\}$? They are: $\emptyset$, $\{a\}$, $\{b\}$, $\{c\}$, $\{a, b\}$, $\{a, c\}$, $\{b, c\}$, $\{a, b, c\}$. The set of all these subsets is $\wp(\{a, b, c\})$:

$$\wp(\{a, b, c\}) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a, b\}, \{b, c\}, \{a, c\}, \{a, b, c\}\}$$

## 1.4 Unions and Intersections

We can define new sets by abstraction, and the property used to define the new set can mention sets we've already defined. So for instance, if $X$ and $Y$ are sets, the set $\{x : x \in X \vee x \in Y\}$ defines a set which consists of all those objects which are elements of either $X$ or $Y$, i.e., it's the set that combines the elements of $X$ and $Y$. This operation on sets—combining them—is very useful and common, and so we give it a name and a symbol.

**Definition 1.11** (Union)**.** The *union* of two sets $X$ and $Y$, written $X \cup Y$, is the set of all things which are elements of $X$, $Y$, or both.

$$X \cup Y = \{x : x \in X \vee x \in Y\}$$

**Example 1.12.** Since the multiplicity of elements doesn't matter, the union of two sets which have an element in common contains that element only once, e.g., $\{a, b, c\} \cup \{a, 0, 1\} = \{a, b, c, 0, 1\}$.

The union of a set and one of its subsets is just the bigger set: $\{a, b, c\} \cup \{a\} = \{a, b, c\}$.

The union of a set with the empty set is identical to the set: $\{a, b, c\} \cup \emptyset = \{a, b, c\}$.

The operation that forms the set of all elements that $X$ and $Y$ have in common is called their *intersection*.

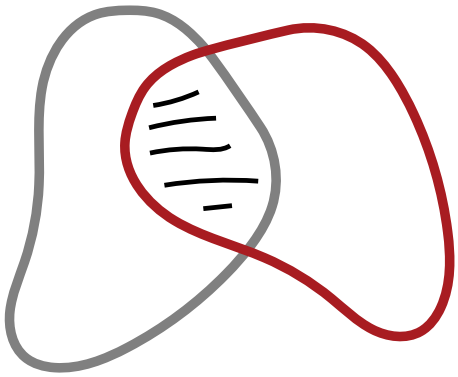


Figure 1.2: The intersection $X \cap Y$ of two sets is the set of elements they have in common.

**Definition 1.13** (Intersection)**.** The *intersection* of two sets $X$ and $Y$, written $X \cap Y$, is the set of all things which are elements of both $X$ and $Y$.

$$X \cap Y = \{x : x \in X \wedge x \in Y\}$$

Two sets are called *disjoint* if their intersection is empty. This means they have no elements in common.

**Example 1.14.** If two sets have no elements in common, their intersection is empty: $\{a, b, c\} \cap \{0, 1\} = \emptyset$.

If two sets do have elements in common, their intersection is the set of all those: $\{a, b, c\} \cap \{a, b, d\} = \{a, b\}$.

The intersection of a set with one of its subsets is just the smaller set: $\{a, b, c\} \cap \{a, b\} = \{a, b\}$.

The intersection of any set with the empty set is empty: $\{a, b, c\} \cap \emptyset = \emptyset$.

We can also form the union or intersection of more than two sets. An elegant way of dealing with this in general is the following: suppose you collect all the sets you want to form the union (or intersection) of into a single set. Then we can define the union of all our original sets as the set of all objects which belong to at least one element of the set, and the intersection as the set of all objects which belong to every element of the set.

**Definition 1.15.** If $Z$ is a set of sets, then $\bigcup Z$ is the set of elements of elements of $Z$:

$$\bigcup Z = \{x : x \text{ belongs to an element of } Z\}, \text{ i.e.,}$$
$$\bigcup Z = \{x : \text{there is a } Y \in Z \text{ so that } x \in Y\}$$

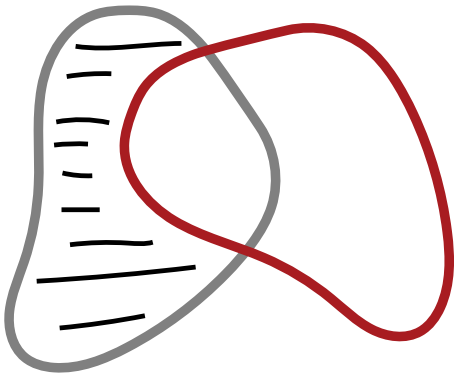


Figure 1.3: The difference $X \setminus Y$ of two sets is the set of those elements of $X$ which are not also elements of $Y$.

**Definition 1.16.** If $Z$ is a set of sets, then $\bigcap Z$ is the set of objects which all elements of $Z$ have in common:

$$\bigcap Z = \{x : x \text{ belongs to every element of } Z\}, \text{ i.e.,}$$
$$\bigcap Z = \{x : \text{for all } Y \in Z, x \in Y\}$$

**Example 1.17.** Suppose $Z = \{\{a, b\}, \{a, d, e\}, \{a, d\}\}$. Then $\bigcup Z = \{a, b, d, e\}$ and $\bigcap Z = \{a\}$.

We could also do the same for a sequence of sets $X_1, X_2, \ldots$

$$\bigcup_i X_i = \{x : x \text{ belongs to one of the } X_i\}$$
$$\bigcap_i X_i = \{x : x \text{ belongs to every } X_i\}.$$

**Definition 1.18** (Difference)**.** The *difference* $X \setminus Y$ is the set of all elements of $X$ which are not also elements of $Y$, i.e.,

$$X \setminus Y = \{x : x \in X \text{ and } x \notin Y\}.$$

## 1.5 Pairs, Tuples, Cartesian Products

Sets have no order to their elements. We just think of them as an unordered collection. So if we want to represent order, we use *ordered pairs* $\langle x, y\rangle$. In an unordered pair $\{x, y\}$, the order does not matter: $\{x, y\} = \{y, x\}$. In an ordered pair, it does: if $x \neq y$, then $\langle x, y\rangle \neq \langle y, x\rangle$.

Sometimes we also want ordered sequences of more than two objects, e.g., *triples* $\langle x, y, z\rangle$, *quadruples* $\langle x, y, z, u\rangle$, and so on. In fact, we can think of

triples as special ordered pairs, where the first element is itself an ordered pair: $\langle x, y, z\rangle$ is short for $\langle\langle x, y\rangle, z\rangle$. The same is true for quadruples: $\langle x, y, z, u\rangle$ is short for $\langle\langle\langle x, y\rangle, z\rangle, u\rangle$, and so on. In general, we talk of *ordered n-tuples* $\langle x_1, \ldots, x_n\rangle$.

**Definition 1.19** (Cartesian product)**.** Given sets $X$ and $Y$, their *Cartesian product* $X \times Y$ is $\{\langle x, y\rangle : x \in X \text{ and } y \in Y\}$.

**Example 1.20.** If $X = \{0, 1\}$, and $Y = \{1, a, b\}$, then their product is

$$X \times Y = \{\langle 0, 1\rangle, \langle 0, a\rangle, \langle 0, b\rangle, \langle 1, 1\rangle, \langle 1, a\rangle, \langle 1, b\rangle\}.$$

**Example 1.21.** If $X$ is a set, the product of $X$ with itself, $X \times X$, is also written $X^2$. It is the set of *all* pairs $\langle x, y\rangle$ with $x, y \in X$. The set of all triples $\langle x, y, z\rangle$ is $X^3$, and so on. We can give an inductive definition:

$$X^1 = X$$
$$X^{k+1} = X^k \times X$$

**Proposition 1.22.** *If $X$ has $n$ elements and $Y$ has $m$ elements, then $X \times Y$ has $n \cdot m$ elements.*

*Proof.* For every element $x$ in $X$, there are $m$ elements of the form $\langle x, y\rangle \in X \times Y$. Let $Y_x = \{\langle x, y\rangle : y \in Y\}$. Since whenever $x_1 \neq x_2$, $\langle x_1, y\rangle \neq \langle x_2, y\rangle$, $Y_{x_1} \cap Y_{x_2} = \emptyset$. But if $X = \{x_1, \ldots, x_n\}$, then $X \times Y = Y_{x_1} \cup \cdots \cup Y_{x_n}$, and so has $n \cdot m$ elements.

To visualize this, arrange the elements of $X \times Y$ in a grid:

$$\begin{array}{rccccc} Y_{x_1} = & \{\langle x_1, y_1\rangle & \langle x_1, y_2\rangle & \ldots & \langle x_1, y_m\rangle\} \\ Y_{x_2} = & \{\langle x_2, y_1\rangle & \langle x_2, y_2\rangle & \ldots & \langle x_2, y_m\rangle\} \\ \vdots & & \vdots & & \\ Y_{x_n} = & \{\langle x_n, y_1\rangle & \langle x_n, y_2\rangle & \ldots & \langle x_n, y_m\rangle\} \end{array}$$

Since the $x_i$ are all different, and the $y_j$ are all different, no two of the pairs in this grid are the same, and there are $n \cdot m$ of them. □

**Example 1.23.** If $X$ is a set, a *word* over $X$ is any sequence of elements of $X$. A sequence can be thought of as an $n$-tuple of elements of $X$. For instance, if $X = \{a, b, c\}$, then the sequence "*bac*" can be thought of as the triple $\langle b, a, c\rangle$. Words, i.e., sequences of symbols, are of crucial importance in computer science, of course. By convention, we count elements of $X$ as sequences of length 1, and $\emptyset$ as the sequence of length 0. The set of *all* words over $X$ then is

$$X^* = \{\emptyset\} \cup X \cup X^2 \cup X^3 \cup \ldots$$

## 1.6 Russell's Paradox

We said that one can define sets by specifying a property that its elements share, e.g., defining the set of Richard's siblings as

$$S = \{x : x \text{ is a sibling of Richard}\}.$$

In the very general context of mathematics one must be careful, however: not every property lends itself to *comprehension*. Some properties do not define sets. If they did, we would run into outright contradictions. One example of such a case is Russell's Paradox.

Sets may be elements of other sets—for instance, the power set of a set $X$ is made up of sets. And so it makes sense, of course, to ask or investigate whether a set is an element of another set. Can a set be a member of itself? Nothing about the idea of a set seems to rule this out. For instance, surely *all* sets form a collection of objects, so we should be able to collect them into a single set—the set of all sets. And it, being a set, would be an element of the set of all sets.

Russell's Paradox arises when we consider the property of not having itself as an element. The set of all sets does not have this property, but all sets we have encountered so far have it. $\mathbb{N}$ is not an element of $\mathbb{N}$, since it is a set, not a natural number. $\wp(X)$ is generally not an element of $\wp(X)$; e.g., $\wp(\mathbb{R}) \notin \wp(\mathbb{R})$ since it is a set of sets of real numbers, not a set of real numbers. What if we suppose that there is a set of all sets that do not have themselves as an element? Does

$$R = \{x : x \notin x\}$$

exist?

If $R$ exists, it makes sense to ask if $R \in R$ or not—it must be either $\in R$ or $\notin R$. Suppose the former is true, i.e., $R \in R$. $R$ was defined as the set of all sets that are not elements of themselves, and so if $R \in R$, then $R$ does not have this defining property of $R$. But only sets that have this property are in $R$, hence, $R$ cannot be an element of $R$, i.e., $R \notin R$. But $R$ can't both be and not be an element of $R$, so we have a contradiction.

Since the assumption that $R \in R$ leads to a contradiction, we have $R \notin R$. But this also leads to a contradiction! For if $R \notin R$, it does have the defining property of $R$, and so would be an element of $R$ just like all the other non-self-containing sets. And again, it can't both not be and be an element of $R$.

## Problems

**Problem 1.1.** Show that there is only one empty set, i.e., show that if $X$ and $Y$ are sets without members, then $X = Y$.

**Problem 1.2.** List all subsets of $\{a, b, c, d\}$.

**Problem 1.3.** Show that if $X$ has $n$ elements, then $\wp(X)$ has $2^n$ elements.

**Problem 1.4.** Prove rigorously that if $X \subseteq Y$, then $X \cup Y = Y$.

**Problem 1.5.** Prove rigorously that if $X \subseteq Y$, then $X \cap Y = X$.

**Problem 1.6.** List all elements of $\{1, 2, 3\}^3$.

**Problem 1.7.** Show, by induction on $k$, that for all $k \geq 1$, if $X$ has $n$ elements, then $X^k$ has $n^k$ elements.

# Chapter 2

# Relations

## 2.1 Relations as Sets

You will no doubt remember some interesting relations between objects of some of the sets we've mentioned. For instance, numbers come with an *order relation* $<$ and from the theory of whole numbers the relation of *divisibility without remainder* (usually written $n \mid m$) may be familar. There is also the relation *is identical with* that every object bears to itself and to no other thing. But there are many more interesting relations that we'll encounter, and even more possible relations. Before we review them, we'll just point out that we can look at relations as a special sort of set. For this, first recall what a *pair* is: if $a$ and $b$ are two objects, we can combine them into the *ordered pair* $\langle a, b\rangle$. Note that for ordered pairs the order *does* matter, e.g, $\langle a, b\rangle \neq \langle b, a\rangle$, in contrast to unordered pairs, i.e., 2-element sets, where $\{a, b\} = \{b, a\}$.

If $X$ and $Y$ are sets, then the *Cartesian product* $X \times Y$ of $X$ and $Y$ is the set of all pairs $\langle a, b\rangle$ with $a \in X$ and $b \in Y$. In particular, $X^2 = X \times X$ is the set of all pairs from $X$.

Now consider a relation on a set, e.g., the $<$-relation on the set $\mathbb{N}$ of natural numbers, and consider the set of all pairs of numbers $\langle n, m\rangle$ where $n < m$, i.e.,

$$R = \{\langle n, m\rangle : n, m \in \mathbb{N} \text{ and } n < m\}.$$

Then there is a close connection between the number $n$ being less than a number $m$ and the corresponding pair $\langle n, m\rangle$ being a member of $R$, namely, $n < m$ if and only if $\langle n, m\rangle \in R$. In a sense we can consider the set $R$ to *be* the $<$-relation on the set $\mathbb{N}$. In the same way we can construct a subset of $\mathbb{N}^2$ for any relation between numbers. Conversely, given any set of pairs of numbers $S \subseteq \mathbb{N}^2$, there is a corresponding relation between numbers, namely, the relationship $n$ bears to $m$ if and only if $\langle n, m\rangle \in S$. This justifies the following definition:

**Definition 2.1** (Binary relation)**.** A *binary relation* on a set $X$ is a subset of $X^2$. If $R \subseteq X^2$ is a binary relation on $X$ and $x, y \in X$, we write $Rxy$ (or $xRy$) for $\langle x, y\rangle \in R$.

**Example 2.2.** The set $\mathbb{N}^2$ of pairs of natural numbers can be listed in a 2-dimensional matrix like this:

$$\begin{array}{cccccc}
\langle \mathbf{0},\mathbf{0}\rangle & \langle 0,1\rangle & \langle 0,2\rangle & \langle 0,3\rangle & \dots \\
\langle 1,0\rangle & \langle \mathbf{1},\mathbf{1}\rangle & \langle 1,2\rangle & \langle 1,3\rangle & \dots \\
\langle 2,0\rangle & \langle 2,1\rangle & \langle \mathbf{2},\mathbf{2}\rangle & \langle 2,3\rangle & \dots \\
\langle 3,0\rangle & \langle 3,1\rangle & \langle 3,2\rangle & \langle \mathbf{3},\mathbf{3}\rangle & \dots \\
\vdots & \vdots & \vdots & \vdots & \ddots
\end{array}$$

The subset consisting of the pairs lying on the diagonal, i.e.,

$$\{\langle 0,0\rangle, \langle 1,1\rangle, \langle 2,2\rangle, \dots\},$$

is the *identity relation on* $\mathbb{N}$. (Since the identity relation is popular, let's define $\mathrm{Id}_X = \{\langle x, x\rangle : x \in X\}$ for any set $X$.) The subset of all pairs lying above the diagonal, i.e.,

$$L = \{\langle 0,1\rangle, \langle 0,2\rangle, \dots, \langle 1,2\rangle, \langle 1,3\rangle, \dots, \langle 2,3\rangle, \langle 2,4\rangle, \dots\},$$

is the *less than* relation, i.e., $Lnm$ iff $n < m$. The subset of pairs below the diagonal, i.e.,

$$G = \{\langle 1,0\rangle, \langle 2,0\rangle, \langle 2,1\rangle, \langle 3,0\rangle, \langle 3,1\rangle, \langle 3,2\rangle, \dots\},$$

is the *greater than* relation, i.e., $Gnm$ iff $n > m$. The union of $L$ with $I$, $K = L \cup I$, is the *less than or equal to* relation: $Knm$ iff $n \leq m$. Similarly, $H = G \cup I$ is the *greater than or equal to relation*. $L$, $G$, $K$, and $H$ are special kinds of relations called *orders*. $L$ and $G$ have the property that no number bears $L$ or $G$ to itself (i.e., for all $n$, neither $Lnn$ nor $Gnn$). Relations with this property are called *irreflexive*, and, if they also happen to be orders, they are called *strict orders.*

Although orders and identity are important and natural relations, it should be emphasized that according to our definition *any* subset of $X^2$ is a relation on $X$, regardless of how unnatural or contrived it seems. In particular, $\emptyset$ is a relation on any set (the *empty relation*, which no pair of elements bears), and $X^2$ itself is a relation on $X$ as well (one which every pair bears), called the *universal relation*. But also something like $E = \{\langle n, m\rangle : n > 5 \text{ or } m \times n \geq 34\}$ counts as a relation.

## 2.2 Special Properties of Relations

Some kinds of relations turn out to be so common that they have been given special names. For instance, $\leq$ and $\subseteq$ both relate their respective domains

(say, $\mathbb{N}$ in the case of $\leq$ and $\wp(X)$ in the case of $\subseteq$) in similar ways. To get at exactly how these relations are similar, and how they differ, we categorize them according to some special properties that relations can have. It turns out that (combinations of) some of these special properties are especially important: orders and equivalence relations.

**Definition 2.3** (Reflexivity)**.** A relation $R \subseteq X^2$ is *reflexive* iff, for every $x \in X$, $Rxx$.

**Definition 2.4** (Transitivity)**.** A relation $R \subseteq X^2$ is *transitive* iff, whenever $Rxy$ and $Ryz$, then also $Rxz$.

**Definition 2.5** (Symmetry)**.** A relation $R \subseteq X^2$ is *symmetric* iff, whenever $Rxy$, then also $Ryx$.

**Definition 2.6** (Anti-symmetry)**.** A relation $R \subseteq X^2$ is *anti-symmetric* iff, whenever both $Rxy$ and $Ryx$, then $x = y$ (or, in other words: if $x \neq y$ then either $\neg Rxy$ or $\neg Ryx$).

In a symmetric relation, $Rxy$ and $Ryx$ always hold together, or neither holds. In an anti-symmetric relation, the only way for $Rxy$ and $Ryx$ to hold together is if $x = y$. Note that this does not *require* that $Rxy$ and $Ryx$ holds when $x = y$, only that it isn't ruled out. So an anti-symmetric relation can be reflexive, but it is not the case that every anti-symmetric relation is reflexive. Also note that being anti-symmetric and merely not being symmetric are different conditions. In fact, a relation can be both symmetric and anti-symmetric at the same time (e.g., the identity relation is).

**Definition 2.7** (Connectivity)**.** A relation $R \subseteq X^2$ is *connected* if for all $x, y \in X$, if $x \neq y$, then either $Rxy$ or $Ryx$.

**Definition 2.8** (Partial order)**.** A relation $R \subseteq X^2$ that is reflexive, transitive, and anti-symmetric is called a *partial order*.

**Definition 2.9** (Linear order)**.** A partial order that is also connected is called a *linear order*.

**Definition 2.10** (Equivalence relation)**.** A relation $R \subseteq X^2$ that is reflexive, symmetric, and transitive is called an *equivalence relation*.

## 2.3 Orders

Very often we are interested in comparisons between objects, where one object may be less or equal or greater than another in a certain respect. Size is the most obvious example of such a comparative relation, or *order*. But not all such relations are alike in all their properties. For instance, some comparative relations require any two objects to be comparable, others don't. (If they do,

we call them *linear* or *total*.) Some include identity (like $\leq$) and some exclude it (like $<$). Let's get some order into all this.

**Definition 2.11** (Preorder)**.** A relation which is both reflexive and transitive is called a *preorder.*

**Definition 2.12** (Partial order)**.** A preorder which is also anti-symmetric is called a *partial order*.

**Definition 2.13** (Linear order)**.** A partial order which is also connected is called a *total order* or *linear order.*

**Example 2.14.** Every linear order is also a partial order, and every partial order is also a preorder, but the converses don't hold. The universal relation on $X$ is a preorder, since it is reflexive and transitive. But, if $X$ has more than one element, the universal relation is not anti-symmetric, and so not a partial order. For a somewhat less silly example, consider the *no longer than* relation $\preccurlyeq$ on $\mathbb{B}^*$: $x \preccurlyeq y$ iff $\mathrm{len}(x) \leq \mathrm{len}(y)$. This is a preorder (reflexive and transitive), and even connected, but not a partial order, since it is not anti-symmetric. For instance, $01 \preccurlyeq 10$ and $10 \preccurlyeq 01$, but $01 \neq 10$.

The relation of *divisibility without remainder* gives us an example of a partial order which isn't a linear order: for integers $n$, $m$, we say $n$ (evenly) divides $m$, in symbols: $n \mid m$, if there is some $k$ so that $m = kn$. On $\mathbb{N}$, this is a partial order, but not a linear order: for instance, $2 \nmid 3$ and also $3 \nmid 2$. Considered as a relation on $\mathbb{Z}$, divisibility is only a preorder since anti-symmetry fails: $1 \mid -1$ and $-1 \mid 1$ but $1 \neq -1$. Another important partial order is the relation $\subseteq$ on a set of sets.

Notice that the examples $L$ and $G$ from Example 2.2, although we said there that they were called "strict orders," are not linear orders even though they are connected (they are not reflexive). But there is a close connection, as we will see momentarily.

**Definition 2.15** (Irreflexivity)**.** A relation $R$ on $X$ is called *irreflexive* if, for all $x \in X$, $\neg Rxx$.

**Definition 2.16** (Asymmetry)**.** A relation $R$ on $X$ is called *asymmetric* if for no pair $x, y \in X$ we have $Rxy$ and $Ryx$.

**Definition 2.17** (Strict order)**.** A *strict order* is a relation which is irreflexive, asymmetric, and transitive.

**Definition 2.18** (Strict linear order)**.** A strict order which is also connected is called a *strict linear order.*

A strict order on $X$ can be turned into a partial order by adding the diagonal $\mathrm{Id}_X$, i.e., adding all the pairs $\langle x, x\rangle$. (This is called the *reflexive closure* of $R$.) Conversely, starting from a partial order, one can get a strict order by removing $\mathrm{Id}_X$.

**Proposition 2.19.** *1. If $R$ is a strict (linear) order on $X$, then $R^+ = R \cup \mathrm{Id}_X$ is a partial order (linear order).*

*2. If $R$ is a partial order (linear order) on $X$, then $R^- = R \setminus \mathrm{Id}_X$ is a strict (linear) order.*

*Proof.* 1. Suppose $R$ is a strict order, i.e., $R \subseteq X^2$ and $R$ is irreflexive, asymmetric, and transitive. Let $R^+ = R \cup \mathrm{Id}_X$. We have to show that $R^+$ is reflexive, antisymmetric, and transitive.

$R^+$ is clearly reflexive, since for all $x \in X$, $\langle x, x\rangle \in \mathrm{Id}_X \subseteq R^+$.

To show $R^+$ is antisymmetric, suppose $R^+xy$ and $R^+yx$, i.e., $\langle x, y\rangle$ and $\langle y, x\rangle \in R^+$, and $x \neq y$. Since $\langle x, y\rangle \in R \cup \mathrm{Id}_X$, but $\langle x, y\rangle \notin \mathrm{Id}_X$, we must have $\langle x, y\rangle \in R$, i.e., $Rxy$. Similarly we get that $Ryx$. But this contradicts the assumption that $R$ is asymmetric.

Now suppose that $R^+xy$ and $R^+yz$. If both $\langle x, y\rangle \in R$ and $\langle y, z\rangle \in R$, it follows that $\langle x, z\rangle \in R$ since $R$ is transitive. Otherwise, either $\langle x, y\rangle \in \mathrm{Id}_X$, i.e., $x = y$, or $\langle y, z\rangle \in \mathrm{Id}_X$, i.e., $y = z$. In the first case, we have that $R^+yz$ by assumption, $x = y$, hence $R^+xz$. Similarly in the second case. In either case, $R^+xz$, thus, $R^+$ is also transitive.

If $R$ is connected, then for all $x \neq y$, either $Rxy$ or $Ryx$, i.e., either $\langle x, y\rangle \in R$ or $\langle y, x\rangle \in R$. Since $R \subseteq R^+$, this remains true of $R^+$, so $R^+$ is connected as well.

2. Exercise.

□

**Example 2.20.** $\leq$ is the linear order corresponding to the strict linear order $<$. $\subseteq$ is the partial order corresponding to the strict order $\subsetneq$.

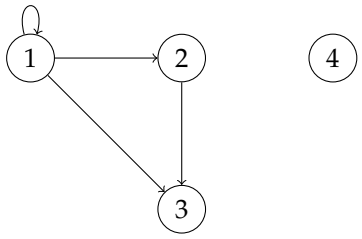
## 2.4 Graphs

A *graph* is a diagram in which points—called "nodes" or "vertices" (plural of "vertex")—are connected by edges. Graphs are a ubiquitous tool in discrete mathematics and in computer science. They are incredibly useful for representing, and visualizing, relationships and structures, from concrete things like networks of various kinds to abstract structures such as the possible outcomes of decisions. There are many different kinds of graphs in the literature which differ, e.g., according to whether the edges are directed or not, have labels or not, whether there can be edges from a node to the same node, multiple edges between the same nodes, etc. *Directed graphs* have a special connection to relations.

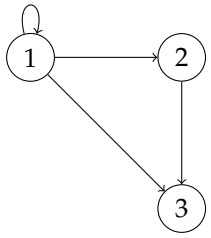
**Definition 2.21** (Directed graph)**.** A *directed graph* $G = \langle V, E\rangle$ is a set of *vertices* $V$ and a set of *edges* $E \subseteq V^2$.

According to our definition, a graph just is a set together with a relation on that set. Of course, when talking about graphs, it's only natural to expect that they are graphically represented: we can draw a graph by connecting two vertices $v_1$ and $v_2$ by an arrow iff $\langle v_1, v_2 \rangle \in E$. The only difference between a relation by itself and a graph is that a graph specifies the set of vertices, i.e., a graph may have isolated vertices. The important point, however, is that every relation $R$ on a set $X$ can be seen as a directed graph $\langle X, R \rangle$, and conversely, a directed graph $\langle V, E \rangle$ can be seen as a relation $E \subseteq V^2$ with the set $V$ explicitly specified.

**Example 2.22.** The graph $\langle V, E \rangle$ with $V = \{1, 2, 3, 4\}$ and $E = \{\langle 1, 1 \rangle, \langle 1, 2 \rangle, \langle 1, 3 \rangle, \langle 2, 3 \rangle\}$ looks like this:




This is a different graph than $\langle V', E \rangle$ with $V' = \{1, 2, 3\}$, which looks like this:




## 2.5 Operations on Relations

It is often useful to modify or combine relations. We've already used the union of relations above (which is just the union of two relations considered as sets of pairs). Here are some other ways:

**Definition 2.23.** Let $R, S \subseteq X^2$ be relations and $Y$ a set.

1. The *inverse* $R^{-1}$ of $R$ is $R^{-1} = \{\langle y, x \rangle : \langle x, y \rangle \in R\}$.
2. The *relative product* $R \mid S$ of $R$ and $S$ is

$$(R \mid S) = \{\langle x, z \rangle : \text{for some } y, Rxy \text{ and } Syz\}$$

3. The *restriction* $R \restriction Y$ of $R$ to $Y$ is $R \cap Y^2$
4. The *application* $R[Y]$ of $R$ to $Y$ is

$$R[Y] = \{y : \text{for some } x \in Y, Rxy\}$$

**Example 2.24.** Let $S \subseteq \mathbb{Z}^2$ be the successor relation on $\mathbb{Z}$, i.e., the set of pairs $\langle x, y\rangle$ where $x + 1 = y$, for $x, y \in \mathbb{Z}$. $Sxy$ holds iff $y$ is the successor of $x$.

1. The inverse $S^{-1}$ of $S$ is the predecessor relation, i.e., $S^{-1}xy$ iff $x - 1 = y$.
2. The relative product $S \mid S$ is the relation $x$ bears to $y$ if $x + 2 = y$.
3. The restriction of $S$ to $\mathbb{N}$ is the successor relation on $\mathbb{N}$.
4. The application of $S$ to a set, e.g., $S[\{1, 2, 3\}]$ is $\{2, 3, 4\}$.

**Definition 2.25** (Transitive closure)**.** The *transitive closure* $R^+$ of a relation $R \subseteq X^2$ is $R^+ = \bigcup_{i=1}^{\infty} R^i$ where $R^1 = R$ and $R^{i+1} = R^i \mid R$.

The *reflexive transitive closure* of $R$ is $R^* = R^+ \cup \mathrm{Id}_X$.

**Example 2.26.** Take the successor relation $S \subseteq \mathbb{Z}^2$. $S^2xy$ iff $x + 2 = y$, $S^3xy$ iff $x + 3 = y$, etc. So $R^*xy$ iff for some $i \geq 1$, $x + i = y$. In other words, $S^+xy$ iff $x < y$ (and $R^*xy$ iff $x \leq y$).

## Problems

**Problem 2.1.** List the elements of the relation $\subseteq$ on the set $\wp(\{a, b, c\})$.

**Problem 2.2.** Give examples of relations that are (a) reflexive and symmetric but not transitive, (b) reflexive and anti-symmetric, (c) anti-symmetric, transitive, but not reflexive, and (d) reflexive, symmetric, and transitive. Do not use relations on numbers or sets.

**Problem 2.3.** Complete the proof of Proposition 2.19, i.e., prove that if $R$ is a partial order on $X$, then $R^- = R \setminus \mathrm{Id}_X$ is a strict order.

**Problem 2.4.** Consider the less-than-or-equal-to relation $\leq$ on the set $\{1, 2, 3, 4\}$ as a graph and draw the corresponding diagram.

**Problem 2.5.** Show that the transitive closure of $R$ is in fact transitive.

# Chapter 3

# Functions

## 3.1 Basics

A *function* is a mapping which pairs each object of a given set with a single partner in another set. For instance, the operation of adding 1 defines a function: each number $n$ is paired with a unique number $n + 1$. More generally, functions may take pairs, triples, etc., of inputs and returns some kind of output. Many functions are familiar to us from basic arithmetic. For instance, addition and multiplication are functions. They take in two numbers and return a third. In this mathematical, abstract sense, a function is a *black box*: what matters is only what output is paired with what input, not the method for calculating the output.

**Definition 3.1** (Function)**.** A *function* $f\colon X \to Y$ is a mapping of each element of $X$ to an element of $Y$. We call $X$ the *domain* of $f$ and $Y$ the *codomain* of $f$. The elements of $X$ are called inputs or *arguments* of $f$, and the element of $Y$ that is paired with an argument $x$ by $f$ is called the *value of* $f$ for argument $x$, written $f(x)$.

The *range* $\operatorname{ran}(f)$ of $f$ is the subset of the codomain consisting of the values of $f$ for some argument; $\operatorname{ran}(f) = \{f(x) : x \in X\}$.

**Example 3.2.** Multiplication takes pairs of natural numbers as inputs and maps them to natural numbers as outputs, so goes from $\mathbb{N} \times \mathbb{N}$ (the domain) to $\mathbb{N}$ (the codomain). As it turns out, the range is also $\mathbb{N}$, since every $n \in \mathbb{N}$ is $n \times 1$.

Multiplication is a function because it pairs each input—each pair of natural numbers—with a single output: $\times\colon \mathbb{N}^2 \to \mathbb{N}$. By contrast, the square root operation applied to the domain $\mathbb{N}$ is not functional, since each positive integer $n$ has two square roots: $\sqrt{n}$ and $-\sqrt{n}$. We can make it functional by only returning the positive square root: $\sqrt{\ }\colon \mathbb{N} \to \mathbb{R}$. The relation that pairs each

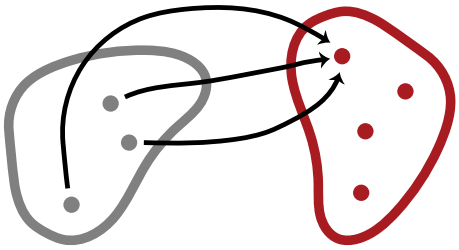


Figure 3.1: A function is a mapping of each element of one set to an element of another. An arrow points from an argument in the domain to the corresponding value in the codomain.

student in a class with their final grade is a function—no student can get two different final grades in the same class. The relation that pairs each student in a class with their parents is not a function—generally each student will have at least two parents.

We can define functions by specifying in some precise way what the value of the function is for every possible argment. Different ways of doing this are by giving a formula, describing a method for computing the value, or listing the values for each argument. However functions are defined, we must make sure that for each argment we specify one, and only one, value.

**Example 3.3.** Let $f\colon \mathbb{N} \to \mathbb{N}$ be defined such that $f(x) = x + 1$. This is a definition that specifies $f$ as a function which takes in natural numbers and outputs natural numbers. It tells us that, given a natural number $x$, $f$ will output its successor $x + 1$. In this case, the codomain $\mathbb{N}$ is not the range of $f$, since the natural number 0 is not the successor of any natural number. The range of $f$ is the set of all positive integers, $\mathbb{Z}^+$.

**Example 3.4.** Let $g\colon \mathbb{N} \to \mathbb{N}$ be defined such that $g(x) = x + 2 - 1$. This tells us that $g$ is a function which takes in natural numbers and outputs natural numbers. Given a natural number $n$, $g$ will output the predecessor of the successor of the successor of $x$, i.e., $x + 1$. Despite their different definitions, $g$ and $f$ are the same function.

Functions $f$ and $g$ defined above are the same because for any natural number $x$, $x + 2 - 1 = x + 1$. $f$ and $g$ pair each natural number with the same output. The definitions for $f$ and $g$ specify the same mapping by means of different equations, and so count as the same function.

**Example 3.5.** We can also define functions by cases. For instance, we could define $h\colon \mathbb{N} \to \mathbb{N}$ by

$$h(x) = \begin{cases} \frac{x}{2} & \text{if } x \text{ is even} \\ \frac{x+1}{2} & \text{if } x \text{ is odd.} \end{cases}$$

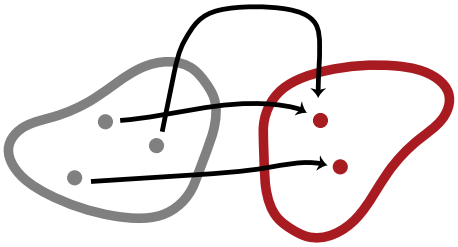


Figure 3.2: A surjective function has every element of the codomain as a value.

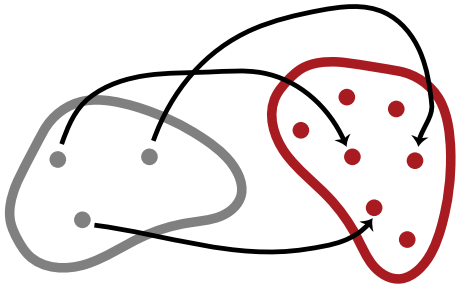


Figure 3.3: An injective function never maps two different arguments to the same value.

Since every natural number is either even or odd, the output of this function will always be a natural number. Just remember that if you define a function by cases, every possible input must fall into exactly one case. In some cases, this will require a a proof that the cases are exhaustive and exclusive.

## 3.2 Kinds of Functions

**Definition 3.6** (Surjective function)**.** A function $f\colon X \to Y$ is *surjective* iff $Y$ is also the range of $f$, i.e., for every $y \in Y$ there is at least one $x \in X$ such that $f(x) = y$.

If you want to show that a function is surjective, then you need to show that every object in the codomain is the output of the function given some input or other.

**Definition 3.7** (Injective function)**.** A function $f\colon X \to Y$ is *injective* iff for each $y \in Y$ there is at most one $x \in X$ such that $f(x) = y$.

Any function pairs each possible input with a unique output. An injective function has a unique input for each possible output. If you want to show that a function $f$ is injective, you need to show that for any elements $x$ and $x'$ of the domain, if $f(x) = f(x')$, then $x = x'$.

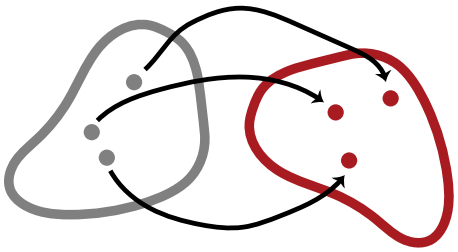


Figure 3.4: A bijective function uniquely pairs the elements of the codomain with those of the domain.

An example of a function which is neither injective, nor surjective, is the constant function $f\colon \mathbb{N} \to \mathbb{N}$ where $f(x) = 1$.

An example of a function which is both injective and surjective is the identity function $f\colon \mathbb{N} \to \mathbb{N}$ where $f(x) = x$.

The successor function $f\colon \mathbb{N} \to \mathbb{N}$ where $f(x) = x + 1$ is injective, but not surjective.

The function

$$f(x) = \begin{cases} \frac{x}{2} & \text{if } x \text{ is even} \\ \frac{x+1}{2} & \text{if } x \text{ is odd.} \end{cases}$$

is surjective, but not injective.

**Definition 3.8** (Bijection)**.** A function $f\colon X \to Y$ is *bijective* iff it is both surjective and injective. We call such a function a *bijection* from $X$ to $Y$ (or between $X$ and $Y$).

## 3.3 Inverses of Functions

One obvious question about functions is whether a given mapping can be "reversed." For instance, the successor function $f(x) = x + 1$ can be reversed in the sense that the function $g(y) = y - 1$ "undoes" what $f$ does. But we must be careful: While the definition of $g$ defines a function $\mathbb{Z} \to \mathbb{Z}$, it does not define a function $\mathbb{N} \to \mathbb{N}$ ($g(0) \notin \mathbb{N}$). So even in simple cases, it is not quite obvious if functions can be reversed, and that it may depend on the domain and codomain. Let's give a precise definition.

**Definition 3.9.** A function $g\colon Y \to X$ is an *inverse* of a function $f\colon X \to Y$ if $f(g(y)) = y$ and $g(f(x)) = x$ for all $x \in X$ and $y \in Y$.

When do functions have inverses? A good candidate for an inverse of $f\colon X \to Y$ is $g\colon Y \to X$ "defined by"

$$g(y) = \text{"the" } x \text{ such that } f(x) = y.$$

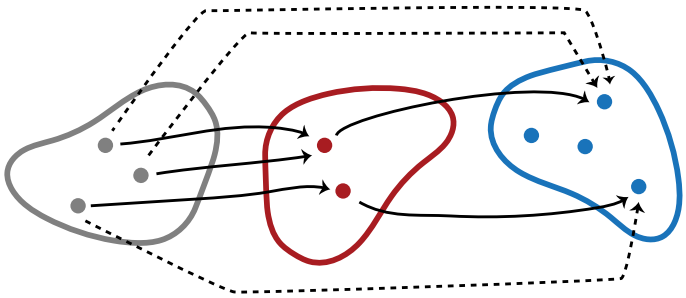


Figure 3.5: The composition $g \circ f$ of two functions $f$ and $g$.

The scare quotes around "defined by" suggest that this is not a definition. At least, it is not in general. For in order for this definition to specify a function, there has to be one and only one $x$ such that $f(x) = y$—the output of $g$ has to be uniquely specified. Moreover, it has to be specified for every $y \in Y$. If there are $x_1$ and $x_2 \in X$ with $x_1 \neq x_2$ but $f(x_1) = f(x_2)$, then $g(y)$ would not be uniquely specified for $y = f(x_1) = f(x_2)$. And if there is no $x$ at all such that $f(x) = y$, then $g(y)$ is not specified at all. In other words, for $g$ to be defined, $f$ has to be injective and surjective.

**Proposition 3.10.** *If $f\colon X \to Y$ is bijective, $f$ has a unique inverse $f^{-1}\colon Y \to X$.*

*Proof.* Exercise. □

## 3.4 Composition of Functions

We have already seen that the inverse $f^{-1}$ of a bijective function $f$ is itself a function. It is also possible to compose functions $f$ and $g$ to define a new function by first applying $f$ and then $g$. Of course, this is only possible if the ranges and domains match, i.e., the range of $f$ must be a subset of the domain of $g$.

**Definition 3.11** (Composition)**.** Let $f\colon X \to Y$ and $g\colon Y \to Z$. The *composition* of $f$ with $g$ is the function $(g \circ f)\colon X \to Z$, where $(g \circ f)(x) = g(f(x))$.

The function $(g \circ f)\colon X \to Z$ pairs each member of $X$ with a member of $Z$. We specify which member of $Z$ a member of $X$ is paired with as follows—given an input $x \in X$, first apply the function $f$ to $x$, which will output some $y \in Y$. Then apply the function $g$ to $y$, which will output some $z \in Z$.

**Example 3.12.** Consider the functions $f(x) = x + 1$, and $g(x) = 2x$. What function do you get when you compose these two? $(g \circ f)(x) = g(f(x))$. So that means for every natural number you give this function, you first add one,

and then you multiply the result by two. So their composition is $(g \circ f)(x) = 2(x+1)$.

## 3.5 Isomorphism

An *isomorphism* is a bijection that preserves the structure of the sets it relates, where structure is a matter of the relationships that obtain between the elements of the sets. Consider the following two sets $X = \{1, 2, 3\}$ and $Y = \{4, 5, 6\}$. These sets are both structured by the relations successor, less than, and greater than. An isomorphism between the two sets is a bijection that preserves those structures. So a bijective function $f\colon X \to Y$ is an isomorphism if, $i < j$ iff $f(i) < f(j)$, $i > j$ iff $f(i) > f(j)$, and $j$ is the successor of $i$ iff $f(j)$ is the successor of $f(i)$.

**Definition 3.13** (Isomorphism)**.** Let $U$ be the pair $\langle X, R\rangle$ and $V$ be the pair $\langle Y, S\rangle$ such that $X$ and $Y$ are sets and $R$ and $S$ are relations on $X$ and $Y$ respectively. A bijection $f$ from $X$ to $Y$ is an *isomorphism* from $U$ to $V$ iff it preserves the relational structure, that is, for any $x_1$ and $x_2$ in $X$, $\langle x_1, x_2\rangle \in R$ iff $\langle f(x_1), f(x_2)\rangle \in S$.

**Example 3.14.** Consider the following two sets $X = \{1, 2, 3\}$ and $Y = \{4, 5, 6\}$, and the relations less than and greater than. The function $f\colon X \to Y$ where $f(x) = 7 - x$ is an isomorphism between $\langle X, <\rangle$ and $\langle Y, >\rangle$.

## 3.6 Partial Functions

It is sometimes useful to relax the definition of function so that it is not required that the output of the function is defined for all possible inputs. Such mappings are called *partial functions*.

**Definition 3.15.** A *partial function* $f\colon X \nrightarrow Y$ is a mapping which assigns to every element of $X$ at most one element of $Y$. If $f$ assigns an element of $Y$ to $x \in X$, we say $f(x)$ is *defined*, and otherwise *undefined*. If $f(x)$ is defined, we write $f(x)\downarrow$, otherwise $f(x)\uparrow$. The *domain* of a partial function $f$ is the subset of $X$ where it is defined, i.e., $\mathrm{dom}(f) = \{x : f(x)\downarrow\}$.

**Example 3.16.** Every function $f\colon X \to Y$ is also a partial function. Partial functions that are defined everywhere on $X$—i.e., what we so far have simply called a function—are also called *total* functions.

**Example 3.17.** The partial function $f\colon \mathbb{R} \nrightarrow \mathbb{R}$ given by $f(x) = 1/x$ is undefined for $x = 0$, and defined everywhere else.

## 3.7 Functions and Relations

A function which maps elements of $X$ to elements of $Y$ obviously defines a relation between $X$ and $Y$, namely the relation which holds between $x$ and $y$ iff $f(x) = y$. In fact, we might even—if we are interested in reducing the building blocks of mathematics for instance—*identify* the function $f$ with this relation, i.e., with a set of pairs. This then raises the question: which relations define functions in this way?

**Definition 3.18** (Graph of a function)**.** Let $f : X \pfun Y$ be a partial function. The *graph* of $f$ is the relation $R_f \subseteq X \times Y$ defined by

$$R_f = \{\langle x, y\rangle : f(x) = y\}.$$

**Proposition 3.19.** *Suppose $R \subseteq X \times Y$ has the property that whenever $Rxy$ and $Rxy'$ then $y = y'$. Then $R$ is the graph of the partial function $f : X \pfun Y$ defined by: if there is a $y$ such that $Rxy$, then $f(x) = y$, otherwise $f(x) \uparrow$. If $R$ is also* serial*, i.e., for each $x \in X$ there is a $y \in Y$ such that $Rxy$, then $f$ is total.*

*Proof.* Suppose there is a $y$ such that $Rxy$. If there were another $y' \neq y$ such that $Rxy'$, the condition on $R$ would be violated. Hence, if there is a $y$ such that $Rxy$, that $y$ is unique, and so $f$ is well-defined. Obviously, $R_f = R$ and $f$ is total if $R$ is serial. □

## Problems

**Problem 3.1.** Show that if $f$ is bijective, an inverse $g$ of $f$ exists, i.e., define such a $g$, show that it is a function, and show that it is an inverse of $f$, i.e., $f(g(y)) = y$ and $g(f(x)) = x$ for all $x \in X$ and $y \in Y$.

**Problem 3.2.** Show that if $f : X \to Y$ has an inverse $g$, then $f$ is bijective.

**Problem 3.3.** Show that if $g : Y \to X$ and $g' : Y \to X$ are inverses of $f : X \to Y$, then $g = g'$, i.e., for all $y \in Y$, $g(y) = g'(y)$.

**Problem 3.4.** Show that if $f : X \to Y$ and $g : Y \to Z$ are both injective, then $g \circ f : X \to Z$ is injective.

**Problem 3.5.** Show that if $f : X \to Y$ and $g : Y \to Z$ are both surjective, then $g \circ f : X \to Z$ is surjective.

**Problem 3.6.** Given $f : X \pfun Y$, define the partial function $g : Y \pfun X$ by: for any $y \in Y$, if there is a unique $x \in X$ such that $f(x) = y$, then $g(y) = x$; otherwise $g(y) \uparrow$. Show that if $f$ is injective, then $g(f(x)) = x$ for all $x \in \mathrm{dom}(f)$, and $f(g(y)) = y$ for all $y \in \mathrm{ran}(f)$.

**Problem 3.7.** Suppose $f : X \to Y$ and $g : Y \to Z$. Show that the graph of $(g \circ f)$ is $R_f \mid R_g$.

# Chapter 4

# The Size of Sets

## 4.1 Introduction

When Georg Cantor developed set theory in the 1870s, his interest was in part to make palatable the idea of an infinite collection—an actual infinity, as the medievals would say. Key to this rehabilitation of the notion of the infinite was a way to assign sizes—"cardinalities"—to sets. The cardinality of a finite set is just a natural number, e.g., $\emptyset$ has cardinality 0, and a set containing five things has cardinality 5. But what about infinite sets? Do they all have the same cardinality, $\infty$? It turns out, they do not.

The first important idea here is that of an enumeration. We can list every finite set by listing all its elements. For some infinite sets, we can also list all their elements if we allow the list itself to be infinite. Such sets are called enumerable. Cantor's surprising result was that some infinite sets are not enumerable.

## 4.2 Enumerable Sets

One way of specifying a finite set is by listing its elements. But conversely, since there are only finitely many elements in a set, every finite set can be enumerated. By this we mean: its elements can be put into a list (a list with a beginning, where each element of the list other than the first has a unique predecessor). Some infinite sets can also be enumerated, such as the set of positive integers.

**Definition 4.1** (Enumeration)**.** Informally, an *enumeration* of a set $X$ is a list (possibly infinite) of elements of $X$ such that every element of $X$ appears on the list at some finite position. If $X$ has an enumeration, then $X$ is said to be *enumerable*. If $X$ is enumerable and infinite, we say $X$ is denumerable.

A couple of points about enumerations:

1. We count as enumerations only lists which have a beginning and in which every element other than the first has a single element immediately preceding it. In other words, there are only finitely many elements between the first element of the list and any other element. In particular, this means that every element of an enumeration has a finite position: the first element has position 1, the second position 2, etc.

2. We can have different enumerations of the same set $X$ which differ by the order in which the elements appear: 4, 1, 25, 16, 9 enumerates the (set of the) first five square numbers just as well as 1, 4, 9, 16, 25 does.

3. Redundant enumerations are still enumerations: 1, 1, 2, 2, 3, 3, ... enumerates the same set as 1, 2, 3, ... does.

4. Order and redundancy *do* matter when we specify an enumeration: we can enumerate the positive integers beginning with 1, 2, 3, 1, ..., but the pattern is easier to see when enumerated in the standard way as 1, 2, 3, 4, ...

5. Enumerations must have a beginning: ..., 3, 2, 1 is not an enumeration of the positive integers because it has no first element. To see how this follows from the informal definition, ask yourself, "at what position in the list does the number 76 appear?"

6. The following is not an enumeration of the positive integers: 1, 3, 5, ..., 2, 4, 6, ... The problem is that the even numbers occur at places $\infty + 1$, $\infty + 2$, $\infty + 3$, rather than at finite positions.

7. Lists may be gappy: 2, $-$, 4, $-$, 6, $-$, ... enumerates the even positive integers.

8. The empty set is enumerable: it is enumerated by the empty list!

**Proposition 4.2.** *If $X$ has an enumeration, it has an enumeration without gaps or repetitions.*

*Proof.* Suppose $X$ has an enumeration $x_1, x_2, \ldots$ in which each $x_i$ is an element of $X$ or a gap. We can remove repetitions from an enumeration by replacing repeated elements by gaps. For instance, we can turn the enumeration into a new one in which $x'_i$ is $x_i$ if $x_i$ is an element of $X$ that is not among $x_1, \ldots, x_{i-1}$ or is $-$ if it is. We can remove gaps by closing up the elements in the list. To make precise what "closing up" amounts to is a bit difficult to describe. Roughly, it means that we can generate a new enumeration $x''_1, x''_2, \ldots$, where each $x''_i$ is the first element in the enumeration $x'_1, x'_2, \ldots$ after $x''_{i-1}$ (if there is one). □

The last argument shows that in order to get a good handle on enumerations and enumerable sets and to prove things about them, we need a more precise definition. The following provides it.

**Definition 4.3** (Enumeration)**.** An *enumeration* of a set $X$ is any surjective function $f\colon \mathbb{Z}^+ \to X$.

Let's convince ourselves that the formal definition and the informal definition using a possibly gappy, possibly infinite list are equivalent. A surjective function (partial or total) from $\mathbb{Z}^+$ to a set $X$ enumerates $X$. Such a function determines an enumeration as defined informally above: the list $f(1)$, $f(2)$, $f(3)$, .... Since $f$ is surjective, every element of $X$ is guaranteed to be the value of $f(n)$ for some $n \in \mathbb{Z}^+$. Hence, every element of $X$ appears at some finite position in the list. Since the function may not be injective, the list may be redundant, but that is acceptable (as noted above).

On the other hand, given a list that enumerates all elements of $X$, we can define a surjective function $f\colon \mathbb{Z}^+ \to X$ by letting $f(n)$ be the $n$th element of the list that is not a gap, or the final element of the list if there is no $n$th element. There is one case in which this does not produce a surjective function: if $X$ is empty, and hence the list is empty. So, every non-empty list determines a surjective function $f\colon \mathbb{Z}^+ \to X$.

**Definition 4.4.** A set $X$ is enumerable iff it is empty or has an enumeration.

**Example 4.5.** A function enumerating the positive integers ($\mathbb{Z}^+$) is simply the identity function given by $f(n) = n$. A function enumerating the natural numbers $\mathbb{N}$ is the function $g(n) = n - 1$.

**Example 4.6.** The functions $f\colon \mathbb{Z}^+ \to \mathbb{Z}^+$ and $g\colon \mathbb{Z}^+ \to \mathbb{Z}^+$ given by

$$\begin{aligned} f(n) &= 2n \text{ and} \\ g(n) &= 2n + 1 \end{aligned}$$

enumerate the even positive integers and the odd positive integers, respectively. However, neither function is an enumeration of $\mathbb{Z}^+$, since neither is surjective.

**Example 4.7.** The function $f(n) = (-1)^n \lceil \frac{(n-1)}{2} \rceil$ (where $\lceil x \rceil$ denotes the *ceiling* function, which rounds $x$ up to the nearest integer) enumerates the set of integers $\mathbb{Z}$. Notice how $f$ generates the values of $\mathbb{Z}$ by "hopping" back and forth between positive and negative integers:

$$\begin{array}{cccccccc} f(1) & f(2) & f(3) & f(4) & f(5) & f(6) & f(7) & \dots \\ -\lceil \frac{0}{2} \rceil & \lceil \frac{1}{2} \rceil & -\lceil \frac{2}{2} \rceil & \lceil \frac{3}{2} \rceil & -\lceil \frac{4}{2} \rceil & \lceil \frac{5}{2} \rceil & -\lceil \frac{6}{2} \rceil & \dots \\ 0 & 1 & -1 & 2 & -2 & 3 & \dots & \end{array}$$

You can also think of $f$ as defined by cases as follows:

$$f(n) = \begin{cases} 0 & \text{if } n = 1 \\ n/2 & \text{if } n \text{ is even} \\ -(n-1)/2 & \text{if } n \text{ is odd and } > 1 \end{cases}$$

That is fine for "easy" sets. What about the set of, say, pairs of positive integers?

$$\mathbb{Z}^+ \times \mathbb{Z}^+ = \{\langle n, m\rangle : n, m \in \mathbb{Z}^+\}$$

We can organize the pairs of positive integers in an *array*, such as the following:

| | **1** | **2** | **3** | **4** | ... |
|---|---|---|---|---|---|
| **1** | $\langle 1,1\rangle$ | $\langle 1,2\rangle$ | $\langle 1,3\rangle$ | $\langle 1,4\rangle$ | ... |
| **2** | $\langle 2,1\rangle$ | $\langle 2,2\rangle$ | $\langle 2,3\rangle$ | $\langle 2,4\rangle$ | ... |
| **3** | $\langle 3,1\rangle$ | $\langle 3,2\rangle$ | $\langle 3,3\rangle$ | $\langle 3,4\rangle$ | ... |
| **4** | $\langle 4,1\rangle$ | $\langle 4,2\rangle$ | $\langle 4,3\rangle$ | $\langle 4,4\rangle$ | ... |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋱ |

Clearly, every ordered pair in $\mathbb{Z}^+ \times \mathbb{Z}^+$ will appear exactly once in the array. In particular, $\langle n, m\rangle$ will appear in the $n$th column and $m$th row. But how do we organize the elements of such an array into a one-way list? The pattern in the array below demonstrates one way to do this:

| | | | | | |
|---|---|---|---|---|---|
| | 1 | 2 | 4 | 7 | ... |
| | 3 | 5 | 8 | ... | ... |
| | 6 | 9 | ... | ... | ... |
| | 10 | ... | ... | ... | ... |
| | ⋮ | ⋮ | ⋮ | ⋮ | ⋱ |

This pattern is called *Cantor's zig-zag method*. Other patterns are perfectly permissible, as long as they "zig-zag" through every cell of the array. By Cantor's zig-zag method, the enumeration for $\mathbb{Z}^+ \times \mathbb{Z}^+$ according to this scheme would be:

$$\langle 1,1\rangle, \langle 1,2\rangle, \langle 2,1\rangle, \langle 1,3\rangle, \langle 2,2\rangle, \langle 3,1\rangle, \langle 1,4\rangle, \langle 2,3\rangle, \langle 3,2\rangle, \langle 4,1\rangle, \ldots$$

What ought we do about enumerating, say, the set of ordered triples of positive integers?

$$\mathbb{Z}^+ \times \mathbb{Z}^+ \times \mathbb{Z}^+ = \{\langle n, m, k\rangle : n, m, k \in \mathbb{Z}^+\}$$

We can think of $\mathbb{Z}^+ \times \mathbb{Z}^+ \times \mathbb{Z}^+$ as the Cartesian product of $\mathbb{Z}^+ \times \mathbb{Z}^+$ and $\mathbb{Z}^+$, that is,

$$(\mathbb{Z}^+)^3 = (\mathbb{Z}^+ \times \mathbb{Z}^+) \times \mathbb{Z}^+ = \{\langle\langle n, m\rangle, k\rangle : \langle n, m\rangle \in \mathbb{Z}^+ \times \mathbb{Z}^+, k \in \mathbb{Z}^+\}$$

and thus we can enumerate $(\mathbb{Z}^+)^3$ with an array by labelling one axis with the enumeration of $\mathbb{Z}^+$, and the other axis with the enumeration of $(\mathbb{Z}^+)^2$:

| | **1** | **2** | **3** | **4** | ... |
|---|---|---|---|---|---|
| $\langle \mathbf{1}, \mathbf{1} \rangle$ | $\langle 1,1,1 \rangle$ | $\langle 1,1,2 \rangle$ | $\langle 1,1,3 \rangle$ | $\langle 1,1,4 \rangle$ | ... |
| $\langle \mathbf{1}, \mathbf{2} \rangle$ | $\langle 1,2,1 \rangle$ | $\langle 1,2,2 \rangle$ | $\langle 1,2,3 \rangle$ | $\langle 1,2,4 \rangle$ | ... |
| $\langle \mathbf{2}, \mathbf{1} \rangle$ | $\langle 2,1,1 \rangle$ | $\langle 2,1,2 \rangle$ | $\langle 2,1,3 \rangle$ | $\langle 2,1,4 \rangle$ | ... |
| $\langle \mathbf{1}, \mathbf{3} \rangle$ | $\langle 1,3,1 \rangle$ | $\langle 1,3,2 \rangle$ | $\langle 1,3,3 \rangle$ | $\langle 1,3,4 \rangle$ | ... |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ |

Thus, by using a method like Cantor's zig-zag method, we may similarly obtain an enumeration of $(\mathbb{Z}^+)^3$.

Cantor's zig-zag method makes the enumerability of $(\mathbb{Z}^+)^2$ (and analogously, $(\mathbb{Z}^+)^3$, etc.) visually evident. Following the zig-zag line in the array and counting the places, we can tell that $\langle 2,3 \rangle$ is at place 8, but specifying the inverse $g\colon (\mathbb{Z}^+)^2 \to \mathbb{Z}^+$ of the zig-zag enumeration such that

$$g(\langle 1,1 \rangle) = 1,\ \ g(\langle 1,2 \rangle) = 2,\ \ g(\langle 2,1 \rangle) = 3,\ \ \ldots\ \ g(\langle 2,3 \rangle) = 8,\ \ \ldots$$

would be helpful. To calculate the position of each pair in the enumeration, we can use the function below. (The exact derivation of the function is somewhat messy, so we are skipping it here.)

$$g(n,m) = \frac{(n+m-2)(n+m-1)}{2} + n$$

Accordingly, the pair $\langle 2,3 \rangle$ is in position $((2+3-2)(2+3-1)/2) + 2 = (3 \cdot 4/2) + 2 = (12/2) + 2 = 8$; pair $\langle 3,7 \rangle$ is in position $((3+7-2)(3+7-1)/2) + 3 = 39$.

Functions like $g$ above, i.e., inverses of enumerations of sets of pairs, are called *pairing functions*.

**Definition 4.8** (Pairing function)**.** A function $f\colon X \times Y \to \mathbb{Z}^+$ is an arithmetical *pairing function* if $f$ is total and injective. We also say that $f$ *encodes* $X \times Y$, and that for $f(\langle x,y \rangle) = n$, $n$ is the *code* for $\langle x,y \rangle$.

The idea is that we can use such functions to encode, e.g., pairs of positive integers in $\mathbb{Z}^+$, or, in other words, represent pairs of positive integers as positive integers. Using the inverse of the pairing function, we can *decode* the integer, i.e., find out which pair of positive integers is represented.

There are other enumerations of $(\mathbb{Z}^+)^2$ that make it easier to figure out what their inverses are. Here is one. Instead of visualizing the enumeration in an array, start with the list of positive integers associated with (initially) empty spaces. Imagine filling these spaces successively with pairs $\langle n,m \rangle$ as follow. Starting with the pairs that have 1 in the first place (i.e., pairs $\langle 1,m \rangle$), put the first (i.e., $\langle 1,1 \rangle$) in the first empty place, then skip an empty space, put

the second (i.e., $\langle 1, 2\rangle$) in the next empty place, skip one again, and so forth. The (incomplete) beginning of our enumeration now looks like this

| $f(1)$ | $f(2)$ | $f(3)$ | $f(4)$ | $f(5)$ | $f(6)$ | $f(7)$ | $f(8)$ | $f(9)$ | $f(10)$ | $\dots$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $\langle 1,1\rangle$ | | $\langle 1,2\rangle$ | | $\langle 1,3\rangle$ | | $\langle 1,4\rangle$ | | $\langle 1,5\rangle$ | | $\dots$ |

Repeat this with pairs $\langle 2, m\rangle$ for the place that still remain empty, again skipping every other empty place:

| $f(1)$ | $f(2)$ | $f(3)$ | $f(4)$ | $f(5)$ | $f(6)$ | $f(7)$ | $f(8)$ | $f(9)$ | $f(10)$ | $\dots$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $\langle 1,1\rangle$ | $\langle 2,1\rangle$ | $\langle 1,2\rangle$ | | $\langle 1,3\rangle$ | $\langle 2,2\rangle$ | $\langle 1,4\rangle$ | | $\langle 1,5\rangle$ | $\langle 2,3\rangle$ | $\dots$ |

Enter pairs $\langle 3, m\rangle$, $\langle 4, m\rangle$, etc., in the same way. Our completed enumeration thus starts like this:

| $f(1)$ | $f(2)$ | $f(3)$ | $f(4)$ | $f(5)$ | $f(6)$ | $f(7)$ | $f(8)$ | $f(9)$ | $f(10)$ | $\dots$ |
|---|---|---|---|---|---|---|---|---|---|---|
| $\langle 1,1\rangle$ | $\langle 2,1\rangle$ | $\langle 1,2\rangle$ | $\langle 3,1\rangle$ | $\langle 1,3\rangle$ | $\langle 2,2\rangle$ | $\langle 1,4\rangle$ | $\langle 4,1\rangle$ | $\langle 1,5\rangle$ | $\langle 2,3\rangle$ | $\dots$ |

If we number the cells in the array above according to this enumeration, we will not find a neat zig-zag line, but this arrangement:

| | **1** | **2** | **3** | **4** | **5** | **6** | $\dots$ |
|---|---|---|---|---|---|---|---|
| **1** | 1 | 3 | 5 | 7 | 9 | 11 | $\dots$ |
| **2** | 2 | 6 | 10 | 14 | 18 | $\dots$ | $\dots$ |
| **3** | 4 | 12 | 20 | 28 | $\dots$ | $\dots$ | $\dots$ |
| **4** | 8 | 24 | 40 | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| **5** | 16 | 48 | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| **6** | 32 | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ | $\dots$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\ddots$ |

We can see that the pairs in the first row are in the odd numbered places of our enumeration, i.e., pair $\langle 1, m\rangle$ is in place $2m - 1$; pairs in the second row, $\langle 1, m\rangle$, are in places whose number is the double of an odd number, specifically, $2 \cdot (2m - 1)$; pairs in the third row, $\langle 1, m\rangle$, are in places whose number is four times an odd number, $4 \cdot (2m - 1)$; and so on. The factors of $(2m - 1)$ for each row, $1, 2, 4, 8, \dots$, are powers of 2: $2^0, 2^1, 2^2, 2^3, \dots$ In fact, the relevant exponent is one less than the first member of the pair in question. Thus, for pair $\langle n, m\rangle$ the factor is $n - 1$. This gives us the general formula: $2^{n-1} \cdot (2m - 1)$, and hence:

**Example 4.9.** The function $f\colon (\mathbb{Z}^+)^2 \to \mathbb{Z}^+$ given by

$$h(n, m) = 2^{n-1}(2m - 1)$$

is a pairing function for the set of pairs of positive integers $(\mathbb{Z}^+)^2$.

Accordingly, in our second enumeration of $(\mathbb{Z}^+)^2$, the pair $\langle 2,3\rangle$ is in position $2^{2-1}\cdot(2\cdot 3-1) = 2\cdot 5 = 10$; pair $\langle 3,7\rangle$ is in position $2^{3-1}\cdot(2\cdot 7-1) = 52$.

Another common pairing function that encodes $(\mathbb{Z}^+)^2$ is the following:

**Example 4.10.** The function $f\colon (\mathbb{Z}^+)^2 \to \mathbb{Z}^+$ given by

$$j(n,m) = 2^n 3^m$$

is a pairing function for the set of pairs of positive integers $(\mathbb{Z}^+)^2$.

$j$ is injective, but nor surjective. That means the inverse of $j$ is a partial, surjective function, and hence an enumeration of $(\mathbb{Z}^+)^2$. (Exercise.)

## 4.3 Non-enumerable Sets

Some sets, such as the set $\mathbb{Z}^+$ of positive integers, are infinite. So far we've seen examples of infinite sets which were all enumerable. However, there are also infinite sets which do not have this property. Such sets are called *non-enumerable*.

First of all, it is perhaps already surprising that there are non-enumerable sets. For any enumerable set $X$ there is a surjective function $f\colon \mathbb{Z}^+ \to X$. If a set is non-enumerable there is no such function. That is, no function mapping the infinitely many elements of $\mathbb{Z}^+$ to $X$ can exhaust all of $X$. So there are "more" elements of $X$ than the infinitely many positive integers.

How would one prove that a set is non-enumerable? You have to show that no such surjective function can exist. Equivalently, you have to show that the elements of $X$ cannot be enumerated in a one way infinite list. The best way to do this is to show that every list of elements of $X$ must leave at least one element out; or that no function $f\colon \mathbb{Z}^+ \to X$ can be surjective. We can do this using Cantor's *diagonal method*. Given a list of elements of $X$, say, $x_1$, $x_2$, ..., we construct another element of $X$ which, by its construction, cannot possibly be on that list.

Our first example is the set $\mathbb{B}^\omega$ of all infinite, non-gappy sequences of 0's and 1's.

**Theorem 4.11.** *$\mathbb{B}^\omega$ is non-enumerable.*

*Proof.* Suppose, by way of contradiction, that $\mathbb{B}^\omega$ is enumerable, i.e., suppose that there is a list $s_1$, $s_2$, $s_3$, $s_4$, ... of all elements of $\mathbb{B}^\omega$. Each of these $s_i$ is itself an infinite sequence of 0's and 1's. Let's call the $j$-th element of the $i$-th sequence in this list $s_i(j)$. Then the $i$-th sequence $s_i$ is

$$s_i(1), s_i(2), s_i(3), \ldots$$

We may arrange this list, and the elements of each sequence $s_i$ in it, in an array:

| | 1 | 2 | 3 | 4 | ... |
|---|---|---|---|---|---|
| 1 | $\mathbf{s_1(1)}$ | $s_1(2)$ | $s_1(3)$ | $s_1(4)$ | ... |
| 2 | $s_2(1)$ | $\mathbf{s_2(2)}$ | $s_2(3)$ | $s_2(4)$ | ... |
| 3 | $s_3(1)$ | $s_3(2)$ | $\mathbf{s_3(3)}$ | $s_3(4)$ | ... |
| 4 | $s_4(1)$ | $s_4(2)$ | $s_4(3)$ | $\mathbf{s_4(4)}$ | ... |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋱ |

The labels down the side give the number of the sequence in the list $s_1, s_2, \ldots$; the numbers across the top label the elements of the individual sequences. For instance, $s_1(1)$ is a name for whatever number, a 0 or a 1, is the first element in the sequence $s_1$, and so on.

Now we construct an infinite sequence, $\bar{s}$, of 0's and 1's which cannot possibly be on this list. The definition of $\bar{s}$ will depend on the list $s_1$, $s_2$, .... Any infinite list of infinite sequences of 0's and 1's gives rise to an infinite sequence $\bar{s}$ which is guaranteed to not appear on the list.

To define $\bar{s}$, we specify what all its elements are, i.e., we specify $\bar{s}(n)$ for all $n \in \mathbb{Z}^+$. We do this by reading down the diagonal of the array above (hence the name "diagonal method") and then changing every 1 to a 0 and every 1 to a 0. More abstractly, we define $\bar{s}(n)$ to be 0 or 1 according to whether the $n$-th element of the diagonal, $s_n(n)$, is 1 or 0.

$$\bar{s}(n) = \begin{cases} 1 & \text{if } s_n(n) = 0 \\ 0 & \text{if } s_n(n) = 1. \end{cases}$$

If you like formulas better than definitions by cases, you could also define $\bar{s}(n) = 1 - s_n(n)$.

Clearly $\bar{s}$ is a non-gappy infinite sequence of 0's and 1's, since it is just the mirror sequence to the sequence of 0's and 1's that appear on the diagonal of our array. So $\bar{s}$ is an element of $\mathbb{B}^\omega$. But it cannot be on the list $s_1, s_2, \ldots$ Why not?

It can't be the first sequence in the list, $s_1$, because it differs from $s_1$ in the first element. Whatever $s_1(1)$ is, we defined $\bar{s}(1)$ to be the opposite. It can't be the second sequence in the list, because $\bar{s}$ differs from $s_2$ in the second element: if $s_2(2)$ is 0, $\bar{s}(2)$ is 1, and vice versa. And so on.

More precisely: if $\bar{s}$ were on the list, there would be some $k$ so that $\bar{s} = s_k$. Two sequences are identical iff they agree at every place, i.e., for any $n$, $\bar{s}(n) = s_k(n)$. So in particular, taking $n = k$ as a special case, $\bar{s}(k) = s_k(k)$ would have to hold. $s_k(k)$ is either 0 or 1. If it is 0 then $\bar{s}(k)$ must be 1—that's how we defined $\bar{s}$. But if $s_k(k) = 1$ then, again because of the way we defined $\bar{s}$, $\bar{s}(k) = 0$. In either case $\bar{s}(k) \neq s_k(k)$.

We started by assuming that there is a list of elements of $\mathbb{B}^\omega$, $s_1$, $s_2$, ... From this list we constructed a sequence $\bar{s}$ which we proved cannot be on the

list. But it definitely is a sequence of 0's and 1's if all the $s_i$ are sequences of 0's and 1's, i.e., $\bar{s} \in \mathbb{B}^\omega$. This shows in particular that there can be no list of *all* elements of $\mathbb{B}^\omega$, since for any such list we could also construct a sequence $\bar{s}$ guaranteed to not be on the list, so the assumption that there is a list of all sequences in $\mathbb{B}^\omega$ leads to a contradiction. □

This proof method is called "diagonalization" because it uses the diagonal of the array to define $\bar{s}$. Diagonalization need not involve the presence of an array: we can show that sets are not enumerable by using a similar idea even when no array and no actual diagonal is involved.

**Theorem 4.12.** *$\wp(\mathbb{Z}^+)$ is not enumerable.*

*Proof.* We proceed in the same way, by showing that for every list of subsets of $\mathbb{Z}^+$ there is a subset of $\mathbb{Z}^+$ which cannot be on the list. Suppose the following is a given list of subsets of $\mathbb{Z}^+$:

$$Z_1, Z_2, Z_3, \ldots$$

We now define a set $\overline{Z}$ such that for any $n \in \mathbb{Z}^+$, $n \in \overline{Z}$ iff $n \notin Z_n$:

$$\overline{Z} = \{n \in \mathbb{Z}^+ : n \notin Z_n\}$$

$\overline{Z}$ is clearly a set of positive integers, since by assumption each $Z_n$ is, and thus $\overline{Z} \in \wp(\mathbb{Z}^+)$. But $\overline{Z}$ cannot be on the list. To show this, we'll establish that for each $k \in \mathbb{Z}^+$, $\overline{Z} \neq Z_k$.

So let $k \in \mathbb{Z}^+$ be arbitrary. We've defined $\overline{Z}$ so that for any $n \in \mathbb{Z}^+$, $n \in \overline{Z}$ iff $n \notin Z_n$. In particular, taking $n = k$, $k \in \overline{Z}$ iff $k \notin Z_k$. But this shows that $\overline{Z} \neq Z_k$, since $k$ is an element of one but not the other, and so $\overline{Z}$ and $Z_k$ have different elements. Since $k$ was arbitrary, $\overline{Z}$ is not on the list $Z_1, Z_2, \ldots$ □

The preceding proof did not mention a diagonal, but you can think of it as involving a diagonal if you picture it this way: Imagine the sets $Z_1, Z_2, \ldots$, written in an array, where each element $j \in Z_i$ is listed in the $j$-th column. Say the first four sets on that list are $\{1, 2, 3, \ldots\}$, $\{2, 4, 6, \ldots\}$, $\{1, 2, 5\}$, and $\{3, 4, 5, \ldots\}$. Then the array would begin with

$$\begin{array}{lccccccc}
Z_1 = \{ & \mathbf{1}, & 2, & 3, & 4, & 5, & 6, & \ldots\} \\
Z_2 = \{ & & \mathbf{2}, & & 4, & & 6, & \ldots\} \\
Z_3 = \{ & 1, & 2, & & & 5 & & \} \\
Z_4 = \{ & & & 3, & \mathbf{4}, & 5, & 6, & \ldots\} \\
\vdots & & & & & \ddots & &
\end{array}$$

Then $\overline{Z}$ is the set obtained by going down the diagonal, leaving out any numbers that appear along the diagonal and include those $j$ where the array has a gap in the $j$-th row/column. In the above case, we would leave out 1 and 2, include 3, leave out 4, etc.

## 4.4 Reduction

We showed $\wp(\mathbb{Z}^+)$ to be non-enumerable by a diagonalization argument. We already had a proof that $\mathbb{B}^\omega$, the set of all infinite sequences of 0s and 1s, is non-enumerable. Here's another way we can prove that $\wp(\mathbb{Z}^+)$ is non-enumerable: Show that *if* $\wp(\mathbb{Z}^+)$ *is enumerable then* $\mathbb{B}^\omega$ *is also enumerable*. Since we know $\mathbb{B}^\omega$ is not enumerable, $\wp(\mathbb{Z}^+)$ can't be either. This is called *reducing* one problem to another—in this case, we reduce the problem of enumerating $\mathbb{B}^\omega$ to the problem of enumerating $\wp(\mathbb{Z}^+)$. A solution to the latter—an enumeration of $\wp(\mathbb{Z}^+)$—would yield a solution to the former—an enumeration of $\mathbb{B}^\omega$.

How do we reduce the problem of enumerating a set $Y$ to that of enumerating a set $X$? We provide a way of turning an enumeration of $X$ into an enumeration of $Y$. The easiest way to do that is to define a surjective function $f\colon X \to Y$. If $x_1, x_2, \ldots$ enumerates $X$, then $f(x_1), f(x_2), \ldots$ would enumerate $Y$. In our case, we are looking for a surjective function $f\colon \wp(\mathbb{Z}^+) \to \mathbb{B}^\omega$.

*Proof of Theorem 4.12 by reduction.* Suppose that $\wp(\mathbb{Z}^+)$ were enumerable, and thus that there is an enumeration of it, $Z_1, Z_2, Z_3, \ldots$

Define the function $f\colon \wp(\mathbb{Z}^+) \to \mathbb{B}^\omega$ by letting $f(Z)$ be the sequence $s_k$ such that $s_k(n) = 1$ iff $n \in Z$, and $s_k(n) = 0$ otherwise. This clearly defines a function, since whenever $Z \subseteq \mathbb{Z}^+$, any $n \in \mathbb{Z}^+$ either is an element of $Z$ or isn't. For instance, the set $2\mathbb{Z}^+ = \{2, 4, 6, \ldots\}$ of positive even numbers gets mapped to the sequence $010101\ldots$, the empty set gets mapped to $0000\ldots$ and the set $\mathbb{Z}^+$ itself to $1111\ldots$.

It also is surjective: Every sequence of 0s and 1s corresponds to some set of positive integers, namely the one which has as its members those integers corresponding to the places where the sequence has 1s. More precisely, suppose $s \in \mathbb{B}^\omega$. Define $Z \subseteq \mathbb{Z}^+$ by:

$$Z = \{n \in \mathbb{Z}^+ : s(n) = 1\}$$

Then $f(Z) = s$, as can be verified by consulting the definition of $f$.

Now consider the list

$$f(Z_1), f(Z_2), f(Z_3), \ldots$$

Since $f$ is surjective, every member of $\mathbb{B}^\omega$ must appear as a value of $f$ for some argument, and so must appear on the list. This list must therefore enumerate all of $\mathbb{B}^\omega$.

So if $\wp(\mathbb{Z}^+)$ were enumerable, $\mathbb{B}^\omega$ would be enumerable. But $\mathbb{B}^\omega$ is non-enumerable (Theorem 4.11). Hence $\wp(\mathbb{Z}^+)$ is non-enumerable. □

It is easy to be confused about the direction the reduction goes in. For instance, a surjective function $g\colon \mathbb{B}^\omega \to X$ does *not* establish that $X$ is non-enumerable. (Consider $g\colon \mathbb{B}^\omega \to \mathbb{B}$ defined by $g(s) = s(1)$, the function that

maps a sequence of 0's and 1's to its first element. It is surjective, because some sequences start with 0 and some start with 1. But $\mathbb{B}$ is finite.) Note also that the function $f$ must be surjective, or otherwise the argument does not go through: $f(x_1)$, $f(x_2)$, ... would then not be guaranteed to include all the elements of $Y$. For instance, $h\colon \mathbb{Z}^+ \to \mathbb{B}^\omega$ defined by

$$h(n) = \underbrace{000\ldots0}_{n\ 0\text{'s}}$$

is a function, but $\mathbb{Z}^+$ is enumerable.

## 4.5 Equinumerous Sets

We have an intuitive notion of "size" of sets, which works fine for finite sets. But what about infinite sets? If we want to come up with a formal way of comparing the sizes of two sets of *any* size, it is a good idea to start with defining when sets are the same size. Let's say sets of the same size are *equinumerous*. We want the formal notion of equinumerosity to correspond with our intuitive notion of "same size," hence the formal notion ought to satisfy the following properties:

**Reflexivity:** Every set is equinumerous with itself.

**Symmetry:** For any sets $X$ and $Y$, if $X$ is equinumerous with $Y$, then $Y$ is equinumerous with $X$.

**Transitivity:** For any sets $X, Y$, and $Z$, if $X$ is equinumerous with $Y$ and $Y$ is equinumerous with $Z$, then $X$ is equinumerous with $Z$.

In other words, we want equinumerosity to be an *equivalence relation*.

**Definition 4.13.** A set $X$ is *equinumerous* with a set $Y$, $X \approx Y$, if and only if there is a bijective $f\colon X \to Y$.

**Proposition 4.14.** *Equinumerosity defines an equivalence relation.*

*Proof.* Let $X, Y$, and $Z$ be sets.

**Reflexivity:** Using the identity map $1_X\colon X \to X$, where $1_X(x) = x$ for all $x \in X$, we see that $X$ is equinumerous with itself (clearly, $1_X$ is bijective).

**Symmetry:** Suppose that $X$ is equinumerous with $Y$. Then there is a bijective $f\colon X \to Y$. Since $f$ is bijective, its inverse $f^{-1}$ exists and also bijective. Hence, $f^{-1}\colon Y \to X$ is a bijective function from $Y$ to $X$, so $Y$ is also equinumerous with $X$.

**Transitivity:** Suppose that $X$ is equinumerous with $Y$ via the bijective function $f\colon X \to Y$ and that $Y$ is equinumerous with $Z$ via the bijective function $g\colon Y \to Z$. Then the composition of $g \circ f\colon X \to Z$ is bijective, and $X$ is thus equinumerous with $Z$.

Therefore, equinumerosity is an equivalence relation. □

**Theorem 4.15.** *Suppose $X$ and $Y$ are equinumerous. Then $X$ is enumerable if and only if $Y$ is.*

*Proof.* Let $X$ and $Y$ be equinumerous. Suppose that $X$ is enumerable. Then either $X = \emptyset$ or there is a surjective function $f\colon \mathbb{Z}^+ \to X$. Since $X$ and $Y$ are equinumerous, there is a bijective $g\colon X \to Y$. If $X = \emptyset$, then $Y = \emptyset$ also (otherwise there would be an element $y \in Y$ but no $x \in X$ with $g(x) = y$). If, on the other hand, $f\colon \mathbb{Z}^+ \to X$ is surjective, then $g \circ f\colon \mathbb{Z}^+ \to Y$ is surjective. To see this, let $y \in Y$. Since $g$ is surjective, there is an $x \in X$ such that $g(x) = y$. Since $f$ is surjective, there is an $n \in \mathbb{Z}^+$ such that $f(n) = x$. Hence,

$$(g \circ f)(n) = g(f(n)) = g(x) = y$$

and thus $g \circ f$ is surjective. We have that $g \circ f$ is an enumeration of $Y$, and so $Y$ is enumerable. □

## 4.6 Comparing Sizes of Sets

Just like we were able to make precise when two sets have the same size in a way that also accounts for the size of infinite sets, we can also compare the sizes of sets in a precise way. Our definition of "is smaller than (or equinumerous)" will require, instead of a bijection between the sets, a total injective function from the first set to the second. If such a function exists, the size of the first set is less than or equal to the size of the second. Intuitively, an injective function from one set to another guarantees that the range of the function has at least as many elements as the domain, since no two elements of the domain map to the same element of the range.

**Definition 4.16.** $X$ is *no larger than* $Y$, $X \preceq Y$, if and only if there is an injective function $f\colon X \to Y$.

**Theorem 4.17** (Schröder-Bernstein)**.** *Let $X$ and $Y$ be sets. If $X \preceq Y$ and $Y \preceq X$, then $X \approx Y$.*

In other words, if there is a total injective function from $X$ to $Y$, and if there is a total injective function from $Y$ back to $X$, then there is a total bijection from $X$ to $Y$. Sometimes, it can be difficult to think of a bijection between two equinumerous sets, so the Schröder-Bernstein theorem allows us to break the comparison down into cases so we only have to think of an injection from

the first to the second, and vice-versa. The Schröder-Bernstein theorem, apart from being convenient, justifies the act of discussing the "sizes" of sets, for it tells us that set cardinalities have the familiar anti-symmetric property that numbers have.

**Definition 4.18.** $X$ is *smaller than* $Y$, $X \prec Y$, if and only if there is an injective function $f\colon X \to Y$ but no bijective $g\colon X \to Y$.

**Theorem 4.19** (Cantor)**.** *For all* $X$, $X \prec \wp(X)$.

*Proof.* The function $f\colon X \to \wp(X)$ that maps any $x \in X$ to its singleton $\{x\}$ is injective, since if $x \neq y$ then also $f(x) = \{x\} \neq \{y\} = f(y)$.

There cannot be a surjective function $g\colon X \to \wp(X)$, let alone a bijective one. For suppose that $g\colon X \to \wp(X)$. Since $g$ is total, every $x \in X$ is mapped to a subset $g(x) \subseteq X$. We show that $g$ cannot be surjective. To do this, we define a subset $Y \subseteq X$ which by definition cannot be in the range of $g$. Let

$$\overline{Y} = \{x \in X : x \notin g(x)\}.$$

Since $g(x)$ is defined for all $x \in X$, $\overline{Y}$ is clearly a well-defined subset of $X$. But, it cannot be in the range of $g$. Let $x \in X$ be arbitrary, we show that $\overline{Y} \neq g(x)$. If $x \in g(x)$, then it does not satisfy $x \notin g(x)$, and so by the definition of $\overline{Y}$, we have $x \notin \overline{Y}$. If $x \in \overline{Y}$, it must satisfy the defining property of $\overline{Y}$, i.e., $x \notin g(x)$. Since $x$ was arbitrary this shows that for each $x \in X$, $x \in g(x)$ iff $x \notin \overline{Y}$, and so $g(x) \neq \overline{Y}$. So $\overline{Y}$ cannot be in the range of $g$, contradicting the assumption that $g$ is surjective. □

It's instructive to compare the proof of Theorem 4.19 to that of Theorem 4.12. There we showed that for any list $Z_1, Z_2, \ldots$, of subsets of $\mathbb{Z}^+$ one can construct a set $\overline{Z}$ of numbers guaranteed not to be on the list. It was guaranteed not to be on the list because, for every $n \in \mathbb{Z}^+$, $n \in Z_n$ iff $n \notin \overline{Z}$. This way, there is always some number that is an element of one of $Z_n$ and $\overline{Z}$ but not the other. We follow the same idea here, except the indices $n$ are now elements of $X$ instead of $\mathbb{Z}^+$. The set $\overline{Y}$ is defined so that it is different from $g(x)$ for each $x \in X$, because $x \in g(x)$ iff $x \notin \overline{Y}$. Again, there is always an element of $X$ which is an element of one of $g(x)$ and $\overline{Y}$ but not the other. And just as $\overline{Z}$ therefore cannot be on the list $Z_1, Z_2, \ldots$, $\overline{Y}$ cannot be in the range of $g$.

## Problems

**Problem 4.1.** According to Definition 4.4, a set $X$ is enumerable iff $X = \emptyset$ or there is a surjective $f\colon \mathbb{Z}^+ \to X$. It is also possible to define "enumerable set" precisely by: a set is enumerable iff there is an injective function $g\colon X \to \mathbb{Z}^+$. Show that the definitions are equivalent, i.e., show that there is an injective function $g\colon X \to \mathbb{Z}^+$ iff either $X = \emptyset$ or there is a surjective $f\colon \mathbb{Z}^+ \to X$.

**Problem 4.2.** Define an enumeration of the positive squares 4, 9, 16, ...

**Problem 4.3.** Show that if $X$ and $Y$ are enumerable, so is $X \cup Y$.

**Problem 4.4.** Show by induction on $n$ that if $X_1, X_2, \ldots, X_n$ are all enumerable, so is $X_1 \cup \cdots \cup X_n$.

**Problem 4.5.** Give an enumeration of the set of all positive rational numbers. (A positive rational number is one that can be written as a fraction $n/m$ with $n, m \in \mathbb{Z}^+$).

**Problem 4.6.** Show that $\mathbb{Q}$ is enumerable. (A rational number is one that can be written as a fraction $z/m$ with $z \in \mathbb{Z}$, $m \in \mathbb{Z}^+$).

**Problem 4.7.** Define an enumeration of $\mathbb{B}^*$.

**Problem 4.8.** Recall from your introductory logic course that each possible truth table expresses a truth function. In other words, the truth functions are all functions from $\mathbb{B}^k \to \mathbb{B}$ for some $k$. Prove that the set of all truth functions is enumerable.

**Problem 4.9.** Show that the set of all finite subsets of an arbitrary infinite enumerable set is enumerable.

**Problem 4.10.** A set of positive integers is said to be *cofinite* iff it is the complement of a finite set of positive integers. Let $I$ be the set that contains all the finite and cofinite sets of positive integers. Show that $I$ is enumerable.

**Problem 4.11.** Show that the enumerable union of enumerable sets is enumerable. That is, whenever $X_1, X_2, \ldots$ are sets, and each $X_i$ is enumerable, then the union $\bigcup_{i=1}^{\infty} X_i$ of all of them is also enumerable.

**Problem 4.12.** Let $f : X \times Y \to \mathbb{Z}^+$ be an arbitrary pairing function. Show that the inverse of $f$ is an enumeration of $X \times Y$.

**Problem 4.13.** Specify a function that encodes $\mathbb{N}^3$.

**Problem 4.14.** Show that $\wp(\mathbb{N})$ is non-enumerable by a diagonal argument.

**Problem 4.15.** Show that the set of functions $f\colon \mathbb{Z}^+ \to \mathbb{Z}^+$ is non-enumerable by an explicit diagonal argument. That is, show that if $f_1, f_2, \ldots$, is a list of functions and each $f_i\colon \mathbb{Z}^+ \to \mathbb{Z}^+$, then there is some $\overline{f}\colon \mathbb{Z}^+ \to \mathbb{Z}^+$ not on this list.

**Problem 4.16.** Show that if there is an injective function $g\colon Y \to X$, and $Y$ is non-enumerable, then so is $X$. Do this by showing how you can use $g$ to turn an enumeration of $X$ into one of $Y$.

**Problem 4.17.** Show that the set of all *sets of* pairs of positive integers is non-enumerable by a reduction argument.

**Problem 4.18.** Show that $\mathbb{N}^\omega$, the set of infinite sequences of natural numbers, is non-enumerable by a reduction argument.

**Problem 4.19.** Let $P$ be the set of functions from the set of positive integers to the set $\{0\}$, and let $Q$ be the set of *partial* functions from the set of positive integers to the set $\{0\}$. Show that $P$ is enumerable and $Q$ is not. (Hint: reduce the problem of enumerating $\mathbb{B}^\omega$ to enumerating $Q$).

**Problem 4.20.** Let $S$ be the set of all surjective functions from the set of positive integers to the set $\{0,1\}$, i.e., $S$ consists of all surjective $f\colon \mathbb{Z}^+ \to \mathbb{B}$. Show that $S$ is non-enumerable.

**Problem 4.21.** Show that the set $\mathbb{R}$ of all real numbers is non-enumerable.

**Problem 4.22.** Show that if $X$ is equinumerous with $U$ and and $Y$ is equinumerous with $V$, and the intersections $X \cap Y$ and $U \cap V$ are empty, then the unions $X \cup Y$ and $U \cup V$ are equinumerous.

**Problem 4.23.** Show that if $X$ is infinite and enumerable, then it is equinumerous with the positive integers $\mathbb{Z}^+$.

**Problem 4.24.** Show that there cannot be an injective function $g\colon \wp(X) \to X$, for any set $X$. Hint: Suppose $g\colon \wp(X) \to X$ is injective. Then for each $x \in X$ there is at most one $Y \subseteq X$ such that $g(Y) = x$. Define a set $\overline{Y}$ such that for every $x \in X$, $g(\overline{Y}) \neq x$.

# Part II

# Propositional Logic

This part contains material on classical propositional logic. The first chapter is relatively rudimenatry and just lists definitions and results, many proofs are not carried out but are left as exercises. The material on proof systems and the completeness theorem is included from the part on first-order logic, with the "FOL" tag set to false. This leaves out everything related to predicates, terms, and quantifiers, and replaces talk of structures $\mathfrak{M}$ with talk about valuations $\mathfrak{v}$.

It is planned to expand this part to include more detail, and to add further topics and results, such as truth-functional completeness.

# Chapter 5

# Syntax and Semantics

This is a very quick summary of definitions only. It should be expanded to provide a gentle intro to proofs by induction on formulas, with lots more examples.

## 5.1 Introduction

Propositional logic deals with formulas that are built from propositional variables using the propositional connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$. Intuitively, a propositional variable $p$ stands for a sentence or proposition that is be true or false. Whenever the "truth value" of the propositional variable in a formula are determined, so is the truth value of any formulas formed from them using propositional connectives. We say that propositional logic is *truth functional*, because its semantics is given by functions of truth values. In particular, in propositional logic we leave out of consideration any further determination of truth and falsity, e.g., whether something is necessarily true rather than just contingently true, or whether something is known to be true, or whether something is true now rather than was true or will be true. We only consider two truth values true ($\mathbb{T}$) and false ($\mathbb{F}$), and so exclude from discussion the possibility that a statement may be neither true nor false, or only half true. We also concentrate only on connectives where the truth value of a formula built from them is completely determined by the truth values of its parts (and not, say, on its meaning). In particular, whether the truth value of conditionals in English is truth functional in this sense is contentious. The material conditional $\rightarrow$ is; other logics deal with conditionals that are not truth functional.

In order to develop the theory and metatheory of truth-functional propositional logic, we must first define the syntax and semantics of its expressions. We will describe one way of constructing formulas from propositional variables using the connectives. Alternative definitions are possible. Other sys-

tems will chose different symbols, will select different sets of connectives as primitive, will use parentheses differently (or even not at all, as in the case of so-called Polish notation). What all approaches have in common, though, is that the formation rules define the set of formulas *inductively*. If done properly, every expression can result essentially in only one way according to the formation rules. The inductive definition resulting in expressions that are *uniquely readable* means we can give meanings to these expressions using the same method—inductive definition.

Giving the meaning of expressions is the domain of semantics. The central concept in semantics for propositonal logic is that of satisfaction in a valuation. A valuation $\mathfrak{v}$ assigns truth values $\mathbb{T}$, $\mathbb{F}$ to the propositional variables. Any valuation determines a truth value $\overline{\mathfrak{v}}(\varphi)$ for any formula $\varphi$. A formula is satisfied in a valuation $\mathfrak{v}$ iff $\overline{\mathfrak{v}}(\varphi) = \mathbb{T}$—we write this as $\mathfrak{v} \vDash \varphi$. This relation can also be defined by induction on the structure of $\varphi$, using the truth functions for the logical connectives to define, say, satisfaction of $\varphi \wedge \psi$ in terms of satisfaction (or not) of $\varphi$ and $\psi$.

On the basis of the satisfaction relation $\mathfrak{v} \vDash \varphi$ for sentences we can then define the basic semantic notions of tautology, entailment, and satisfiability. A formula is a tautology, $\vDash \varphi$, if every valuation satisfies it, i.e., $\overline{\mathfrak{v}}(\varphi) = \mathbb{T}$ for any $\mathfrak{v}$. It is entailed by a set of formulas, $\Gamma \vDash \varphi$, if every valuation that satisfies all the formulas in $\Gamma$ also satisfies $\varphi$. And a set of formulas is satisfiable if some valuation satisfies all formulas in it at the same time. Because formulas are inductively defined, and satisfaction is in turn defined by induction on the structure of formulas, we can use induction to prove properties of our semantics and to relate the semantic notions defined.

## 5.2 Propositional Formulas

Formulas of propositional logic are built up from *propositional variables* and the propositional constant $\bot$ using *logical connectives*.

1. A denumerable set $\mathrm{At}_0$ of propositional variables $p_0$, $p_1$, ...
2. The propositional constant for falsity $\bot$.
3. The logical connectives: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (conditional)
4. Punctuation marks: (, ), and the comma.

In addition to the primitive connectives introduced above, we also use the following *defined* symbols: $\leftrightarrow$ (biconditional), truth $\top$

A defined symbol is not officially part of the language, but is introduced as an informal abbreviation: it allows us to abbreviate formulas which would,

if we only used primitive symbols, get quite long. This is obviously an advantage. The bigger advantage, however, is that proofs become shorter. If a symbol is primitive, it has to be treated separately in proofs. The more primitive symbols, therefore, the longer our proofs.

You may be familiar with different terminology and symbols than the ones we use above. Logic texts (and teachers) commonly use either $\sim$, $\neg$, and ! for "negation", $\wedge$, $\cdot$, and & for "conjunction". Commonly used symbols for the "conditional" or "implication" are $\rightarrow$, $\Rightarrow$, and $\supset$. Symbols for "biconditional," "bi-implication," or "(material) equivalence" are $\leftrightarrow$, $\Leftrightarrow$, and $\equiv$. The $\bot$ symbol is variously called "falsity," "falsum,", "absurdity,", or "bottom." The $\top$ symbol is variously called "truth," "verum,", or "top."

**Definition 5.1** (Formula)**.** The set $\mathrm{Frm}(\mathcal{L}_0)$ of *formulas* of propositional logic is defined inductively as follows:

1. $\bot$ is an atomic formula.
2. Every propositional variable $p_i$ is an atomic formula.
3. If $\varphi$ is a formula, then $\neg\varphi$ is formula.
4. If $\varphi$ and $\psi$ are formulas, then $(\varphi \wedge \psi)$ is a formula.
5. If $\varphi$ and $\psi$ are formulas, then $(\varphi \vee \psi)$ is a formula.
6. If $\varphi$ and $\psi$ are formulas, then $(\varphi \rightarrow \psi)$ is a formula.
7. If $\varphi$ is a formula and $x$ is a variable, then $\forall x\, \varphi$ is a formula.
8. If $\varphi$ is a formula and $x$ is a variable, then $\exists x\, \varphi$ is a formula.
9. Nothing else is a formula.

The definitions of the set of terms and that of formulas are *inductive definitions*. Essentially, we construct the set of formulas in infinitely many stages. In the initial stage, we pronounce all atomic formulas to be formulas; this corresponds to the first few cases of the definition, i.e., the cases for $\bot$, $p_i$. "Atomic formula" thus means any formula of this form.

The other cases of the definition give rules for constructing new formulas out of formulas already constructed. At the second stage, we can use them to construct formulas out of atomic formulas. At the third stage, we construct new formulas from the atomic formulas and those obtained in the second stage, and so on. A formula is anything that is eventually constructed at such a stage, and nothing else.

**Definition 5.2.** Formulas constructed using the defined operators are to be understood as follows:

1. $\top$ abbreviates $\neg\bot$.

2. $\varphi \leftrightarrow \psi$ abbreviates $(\varphi \to \psi) \wedge (\psi \to \varphi)$.

**Definition 5.3** (Syntactic identity)**.** The symbol $\equiv$ expresses syntactic identity between strings of symbols, i.e., $\varphi \equiv \psi$ iff $\varphi$ and $\psi$ are strings of symbols of the same length and which contain the same symbol in each place.

The $\equiv$ symbol may be flanked by strings obtained by concatenation, e.g., $\varphi \equiv (\psi \vee \chi)$ means: the string of symbols $\varphi$ is the same string as the one obtained by concatenating an opening parenthesis, the string $\psi$, the $\vee$ symbol, the string $\chi$, and a closing parenthesis, in this order. If this is the case, then we know that the first symbol of $\varphi$ is an opening parenthesis, $\varphi$ contains $\psi$ as a substring (starting at the second symbol), that substring is followed by $\vee$, etc.

## 5.3 Preliminaries

**Theorem 5.4** (*Principle of induction on formulas*)**.** *If some property $P$ holds for all the atomic formulas and is such that*

1. *it holds for $\neg\varphi$ whenever it holds for $\varphi$;*
2. *it holds for $(\varphi \wedge \psi)$ whenever it holds for $\varphi$ and $\psi$;*
3. *it holds for $(\varphi \vee \psi)$ whenever it holds for $\varphi$ and $\psi$;*
4. *it holds for $(\varphi \to \psi)$ whenever it holds for $\varphi$ and $\psi$;*

*then $P$ holds for all formulas.*

*Proof.* Let $S$ be the collection of all formulas with property $P$. Clearly $S \subseteq \mathrm{Frm}(\mathcal{L}_0)$. $S$ satisfies all the conditions of Definition 5.1: it contains all atomic formulas and is closed under the logical operators. $\mathrm{Frm}(\mathcal{L}_0)$ is the smallest such class, so $\mathrm{Frm}(\mathcal{L}_0) \subseteq S$. So $\mathrm{Frm}(\mathcal{L}_0) = S$, and every formula has property $P$. □

**Proposition 5.5.** *Any formula in* $\mathrm{Frm}(\mathcal{L}_0)$ *is* balanced*, in that it has as many left parentheses as right ones.*

**Proposition 5.6.** *No proper initial segment of a formula is a formula.*

**Proposition 5.7** (Unique Readability)**.** *Any formula $\varphi$ in* $\mathrm{Frm}(\mathcal{L}_0)$ *has exactly one parsing as one of the following*

1. $\bot$.
2. $p_n$ *for some* $p_n \in \mathrm{At}_0$.
3. $\neg\psi$ *for some formula* $\psi$.
4. $(\psi \wedge \chi)$ *for some formulas* $\psi$ *and* $\chi$.

5. $(\psi \vee \chi)$ *for some formulas* $\psi$ *and* $\chi$.

6. $(\psi \to \chi)$ *for some formulas* $\psi$ *and* $\chi$.

*Moreover, this parsing is* unique.

*Proof.* By induction on $\varphi$. For instance, suppose that $\varphi$ has two distinct readings as $(\psi \to \chi)$ and $(\psi' \to \chi')$. Then $\psi$ and $\psi'$ must be the same (or else one would be a proper initial segment of the other); so if the two readings of $\varphi$ are distinct it must be because $\chi$ and $\chi'$ are distinct readings of the same sequence of symbols, which is impossible by the inductive hypothesis. □

**Definition 5.8** (Uniform Substitution)**.** If $\varphi$ and $\psi$ are formulas, and $p_i$ is a propositional variable, then $\varphi[\psi/p_i]$ denotes the result of replacing each occurrence of $p_i$ by an occurrence of $\psi$ in $\varphi$; similarly, the simultaneous substitution of $p_1, \ldots, p_n$ by formulas $\psi_1, \ldots, B_n$ is denoted by $\varphi[\psi_1/p_1, \ldots, \psi_n/p_n]$.

## 5.4 Valuations and Satisfaction

**Definition 5.9** (Valuations)**.** Let $\{\mathbb{T}, \mathbb{F}\}$ be the set of the two truth values, "true" and "false." A *valuation* for $\mathcal{L}_0$ is a function $\mathfrak{v}$ assigning either $\mathbb{T}$ or $\mathbb{F}$ to the propositional variables of the language, i.e., $\mathfrak{v}\colon \mathrm{At}_0 \to \{\mathbb{T}, \mathbb{F}\}$.

**Definition 5.10.** Given a valuation $v$, define the evaluation function $\overline{\mathfrak{v}}(\,:\,)\mathrm{Frm}(\mathcal{L}_0) \to \{\mathbb{T}, \mathbb{F}\}$ inductively by:

$$
\begin{aligned}
\overline{\mathfrak{v}}(\bot) &= \mathbb{F};\\
\overline{\mathfrak{v}}(p_n) &= \mathfrak{v}(p_n);\\
\overline{\mathfrak{v}}(\neg\varphi) &= \begin{cases} \mathbb{T} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{F};\\ \mathbb{F} & \text{otherwise.}\end{cases}\\
\overline{\mathfrak{v}}(\varphi \wedge \psi) &= \begin{cases} \mathbb{T} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{T} \text{ and } \overline{\mathfrak{v}}(\psi) = \mathbb{T};\\ \mathbb{F} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{F} \text{ or } \overline{\mathfrak{v}}(\psi) = \mathbb{F}.\end{cases}\\
\overline{\mathfrak{v}}(\varphi \vee \psi) &= \begin{cases} \mathbb{T} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{T} \text{ or } \overline{\mathfrak{v}}(\psi) = \mathbb{T};\\ \mathbb{F} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{F} \text{ and } \overline{\mathfrak{v}}(\psi) = \mathbb{F}.\end{cases}\\
\overline{\mathfrak{v}}(\varphi \to \psi) &= \begin{cases} \mathbb{T} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{F} \text{ or } \overline{\mathfrak{v}}(\psi) = \mathbb{T};\\ \mathbb{F} & \text{if } \overline{\mathfrak{v}}(\varphi) = \mathbb{T} \text{ and } \overline{\mathfrak{v}}(\psi) = \mathbb{F}.\end{cases}
\end{aligned}
$$

The valuation clauses correspond to the following truth tables:

| $\varphi$ | $\psi$ | $\varphi \wedge \psi$ | $\varphi \vee \psi$ | $\varphi \to \psi$ | $\varphi \leftrightarrow \psi$ |
|---|---|---|---|---|---|
| $\mathbb{T}$ | $\mathbb{T}$ | $\mathbb{T}$ | $\mathbb{T}$ | $\mathbb{T}$ | $\mathbb{T}$ |
| $\mathbb{T}$ | $\mathbb{F}$ | $\mathbb{F}$ | $\mathbb{T}$ | $\mathbb{F}$ | $\mathbb{F}$ |
| $\mathbb{F}$ | $\mathbb{T}$ | $\mathbb{F}$ | $\mathbb{T}$ | $\mathbb{T}$ | $\mathbb{F}$ |
| $\mathbb{F}$ | $\mathbb{F}$ | $\mathbb{F}$ | $\mathbb{F}$ | $\mathbb{T}$ | $\mathbb{T}$ |

**Theorem 5.11** (Local Determination)**.** *Suppose that* $\mathfrak{v}_1$ *and* $\mathfrak{v}_2$ *are valuations that agree on the propositional letters occurring in* $\varphi$*, i.e.,* $\overline{\mathfrak{v}_1}(p_n) = \overline{\mathfrak{v}_2}(p_n)$ *whenever* $p_n$ *occurs in* $\varphi$*. Then they also agree on any* $\varphi$*, i.e.,* $\overline{\mathfrak{v}_1}(\varphi) = \overline{\mathfrak{v}_2}(\varphi)$*.*

*Proof.* By induction on $\varphi$. □

**Definition 5.12** (Satisfaction)**.** Using the evaluation function, we can define the notion of *satisfaction of a formula* $\varphi$ *by a valuation* $\mathfrak{v}$, $\mathfrak{v} \vDash \varphi$, inductively as follows. (We write $\mathfrak{v} \nvDash \varphi$ to mean "not $\mathfrak{v} \vDash \varphi$.")

1. $\varphi \equiv \bot$: $\mathfrak{v} \nvDash \varphi$.
2. $\varphi \equiv p_i$: $\mathfrak{M} \vDash \varphi$ iff $\overline{\mathfrak{v}}(p_i) = \mathbb{T}$.
3. $\varphi \equiv \neg\psi$: $\mathfrak{v} \vDash \varphi$ iff $\mathfrak{v} \nvDash \psi$.
4. $\varphi \equiv (\psi \wedge \chi)$: $\mathfrak{v} \vDash \varphi$ iff $\mathfrak{v} \vDash \psi$ and $\mathfrak{v} \vDash \chi$.
5. $\varphi \equiv (\psi \vee \chi)$: $\mathfrak{v} \vDash \varphi$ iff $\mathfrak{v} \vDash \varphi$ or $\mathfrak{v} \vDash \psi$ (or both).
6. $\varphi \equiv (\psi \to \chi)$: $\mathfrak{v} \vDash \varphi$ iff $\mathfrak{v} \nvDash \psi$ or $\mathfrak{v} \vDash \chi$ (or both).

If $\Gamma$ is a set of formulas, $\mathfrak{v} \vDash \Gamma$ iff $\mathfrak{v} \vDash \varphi$ for every $\varphi \in \Gamma$.

**Proposition 5.13.** $\mathfrak{v} \vDash \varphi$ *iff* $\overline{\mathfrak{v}}(\varphi) = \mathbb{T}$.

*Proof.* By induction on $\varphi$. □

## 5.5 Semantic Notions

We define the following semantic notions:

**Definition 5.14.**
1. A formula $\varphi$ is *satisfiable* if for some $\mathfrak{v}$, $\mathfrak{v} \vDash \varphi$; it is *unsatisfiable* if for no $\mathfrak{v}$, $\mathfrak{v} \vDash \varphi$;
2. A formula $\varphi$ is a *tautology* if $\mathfrak{v} \vDash \varphi$ for all valuations $v$;
3. A formula $\varphi$ is *contingent* if it is satisfiable but not a tautology;
4. If $\Gamma$ is a set of formulas, $\Gamma \vDash \varphi$ ("$\Gamma$ entails $\varphi$") if and only if $\mathfrak{v} \vDash \varphi$ for every valuation $\mathfrak{v}$ for which $\mathfrak{v} \vDash \Gamma$.
5. If $\Gamma$ is a set of formulas, $\Gamma$ is *satisfiable* if there is a valuation $\mathfrak{v}$ for which $\mathfrak{v} \vDash \Gamma$, and $\Gamma$ is *unsatisfiable* otherwise.

**Proposition 5.15.** *1. $\varphi$ is a tautology if and only if $\emptyset \vDash \varphi$;*

*2. If $\Gamma \vDash \varphi$ and $\Gamma \vDash \varphi \to \psi$ then $\Gamma \vDash \psi$;*

*3. If $\Gamma$ is satisfiable then every finite subset of $\Gamma$ is also satisfiable;*

*4. Monotony: if $\Gamma \subseteq \Delta$ and $\Gamma \vDash \varphi$ then also $\Delta \vDash \varphi$;*

*5. Transitivity: if $\Gamma \vDash \varphi$ and $\Delta \cup \{\varphi\} \vDash \psi$ then $\Gamma \cup \Delta \vDash \psi$;*

*Proof.* Exercise. □

**Proposition 5.16.** *$\Gamma \vDash \varphi$ if and only if $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable;*

*Proof.* Exercise. □

**Theorem 5.17** (Semantic Deduction Theorem)**.** *$\Gamma \vDash \varphi \to \psi$ if and only if $\Gamma \cup \{\varphi\} \vDash \psi$.*

*Proof.* Exercise. □

## Problems

**Problem 5.1.** Prove Proposition 5.5

**Problem 5.2.** Prove Proposition 5.6

**Problem 5.3.** Give a mathematically rigorous definition of $\varphi[\psi/p]$ by induction.

**Problem 5.4.** Prove Proposition 5.13

**Problem 5.5.** Prove Proposition 5.15

**Problem 5.6.** Prove Proposition 5.16

**Problem 5.7.** Prove Theorem 5.17

# Chapter 6

# Derivation Systems

This chapter collects general material on derivation systems. A textbook using a specific system can insert the introduction section plus the relevant survey section at the beginning of the chapter introducing that system.

## 6.1 Introduction

Logics commonly have both a semantics and a derivation system. The semantics concerns concepts such as truth, satisfiability, validity, and entailment. The purpose of derivation systems is to provide a purely syntactic method of establishing entailment and validity. They are purely syntactic in the sense that a derivation in such a system is a finite syntactic object, usually a sequence (or other finite arrangement) of sentences or formulas. Good derivation systems have the property that any given sequence or arrangement of sentences or formulas can be verified mechanically to be "correct."

The simplest (and historically first) derivation systems for first-order logic were *axiomatic*. A sequence of formulas counts as a derivation in such a system if each individual formula in it is either among a fixed set of "axioms" or follows from formulas coming before it in the sequence by one of a fixed number of "inference rules"—and it can be mechanically verified if a formula is an axiom and whether it follows correctly from other formulas by one of the inference rules. Axiomatic proof systems are easy to describe—and also easy to handle meta-theoretically—but derivations in them are hard to read and understand, and are also hard to produce.

Other derivation systems have been developed with the aim of making it easier to construct derivations or easier to understand derivations once they are complete. Examples are natural deduction, truth trees, also known as tableaux proofs, and the sequent calculus. Some derivation systems are de-

signed especially with mechanization in mind, e.g., the resolution method is easy to implement in software (but its derivations are essentially impossible to understand). Most of these other proof systems represent derivations as trees of formulas rather than sequences. This makes it easier to see which parts of a derivation depend on which other parts.

So for a given logic, such as first-order logic, the different derivation systems will give different explications of what it is for a sentence to be a *theorem* and what it means for a sentence to be derivable from some others. However that is done (via axiomatic derivations, natural deductions, sequent derivations, truth trees, resolution refutations), we want these relations to match the semantic notions of validity and entailment. Let's write $\vdash \varphi$ for "$\varphi$ is a theorem" and "$\Gamma \vdash \varphi$" for "$\varphi$ is derivable from $\Gamma$." However $\vdash$ is defined, we want it to match up with $\models$, that is:

1. $\vdash \varphi$ if and only if $\models \varphi$
2. $\Gamma \vdash \varphi$ if and only if $\Gamma \models \varphi$

The "only if" direction of the above is called *soundness*. A derivation system is sound if derivability guarantees entailment (or validity). Every decent derivation system has to be sound; unsound derivation systems are not useful at all. After all, the entire purpose of a derivation is to provide a syntactic guarantee of validity or entailment. We'll prove soundness for the derivation systems we present.

The converse "if" direction is also important: it is called *completeness*. A complete derivation system is strong enough to show that $\varphi$ is a theorem whenever $\varphi$ is valid, and that there $\Gamma \vdash \varphi$ whenever $\Gamma \models \varphi$. Completeness is harder to establish, and some logics have no complete derivation systems. First-order logic does. Kurt Gödel was the first one to prove completeness for a derivation system of first-order logic in his 1929 dissertation.

Another concept that is connected to derivation systems is that of *consistency*. A set of sentences is called inconsistent if anything whatsoever can be derived from it, and consistent otherwise. Inconsistency is the syntactic counterpart to unsatisfiablity: like unsatisfiable sets, inconsistent sets of sentences do not make good theories, they are defective in a fundamental way. Consistent sets of sentences may not be true or useful, but at least they pass that minimal threshold of logical usefulness. For different derivation systems the specific definition of consistency of sets of sentences might differ, but like $\vdash$, we want consistency to coincide with its semantic counterpart, satisfiability. We want it to always be the case that $\Gamma$ is consistent if and only if it is satisfiable. Here, the "if" direction amounts to completeness (consistency guarantees satisfiability), and the "only if" direction amounts to soundness (satisfiability guarantees consistency). In fact, for classical first-order logic, the two versions of soundness and completeness are equivalent.

## 6.2 The Sequent Calculus

While many derivation systems operate with arrangements of sentences, the sequent calculus operates with *sequents*. A sequent is an expression of the form

$$\varphi_1, \ldots, \varphi_m \Rightarrow \psi_1, \ldots, \psi_m,$$

that is a pair of sequences of sentences, separated by the sequent symbol $\Rightarrow$. Either sequence may be empty. A derivation in the sequent calculus is a tree of sequents, where the topmost sequents are of a special form (they are called "initial sequents" or "axioms") and every other sequent follows from the sequents immediately above it by one of the rules of inference. The rules of inference either manipulate the sentences in the sequents (adding, removing, or rearranging them on either the left or the right), or they introduce a complex formula in the conclusion of the rule. For instance, the $\wedge$L rule allows the inference from $\varphi, \Gamma \Rightarrow \Delta$ to $A \wedge \psi, \Gamma \Rightarrow \Delta$, and the $\rightarrow$R allows the inference from $\varphi, \Gamma \Rightarrow \Delta, \psi$ to $\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi$, for any $\Gamma$, $\Delta$, $\varphi$, and $\psi$. (In particular, $\Gamma$ and $\Delta$ may be empty.)

The $\vdash$ relation based on the sequent calculus is defined as follows: $\Gamma \vdash \varphi$ iff there is some sequence $\Gamma_0$ such that every $\varphi$ in $\Gamma_0$ is in $\Gamma$ and there is a derivation with the sequent $\Gamma_0 \Rightarrow \varphi$ at its root. $\varphi$ is a theorem in the sequent calculus if the sequent $\Rightarrow \varphi$ has a derivation. For instance, here is a derivation that shows that $\vdash (\varphi \wedge \psi) \rightarrow \varphi$:

$$\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi}\,\wedge\text{L}}{\Rightarrow (\varphi \wedge \psi) \rightarrow \varphi}\,\rightarrow\text{R}$$

A set $\Gamma$ is inconsistent in the sequent calculus if there is a derivation of $\Gamma_0 \Rightarrow$ (where every $\varphi \in \Gamma_0$ is in $\Gamma$ and the right side of the sequent is empty). Using the rule WR, any sentence can be derived from an inconsistent set.

The sequent calculus was invented in the 1930s by Gerhard Gentzen. Because of its systematic and symmetric design, it is a very useful formalism for developing a theory of derivations. It is relatively easy to find derivations in the sequent calculus, but these derivations are often hard to read and their connection to proofs are sometimes not easy to see. It has proved to be a very elegant approach to derivation systems, however, and many logics have sequent calculus systems.

## 6.3 Natural Deduction

Natural deduction is a derivation system intended to mirror actual reasoning (especially the kind of regimented reasoning employed by mathematicians). Actual reasoning proceeds by a number of "natural" patterns. For instance,

proof by cases allows us to establish a conclusion on the basis of a disjunctive premise, by establishing that the conclusion follows from either of the disjuncts. Indirect proof allows us to establish a conclusion by showing that its negation leads to a contradiction. Conditional proof establishes a conditional claim "if ... then ..." by showing that the consequent follows from the antecedent. Natural deduction is a formalization of some of these natural inferences. Each of the logical connectives and quantifiers comes with two rules, an introduction and an elimination rule, and they each correspond to one such natural inference pattern. For instance, $\rightarrow$Intro corresponds to conditional proof, and $\vee$Elim to proof by cases. A particularly simple rule is $\wedge$Elim which allows the inference from $\varphi \wedge \psi$ to $\varphi$ (or $\psi$).

One feature that distinguishes natural deduction from other derivation systems is its use of assumptions. A derivation in natural deduction is a tree of formulas. A single formula stands at the root of the tree of formulas, and the "leaves" of the tree are formulas from which the conclusion is derived. In natural deduction, some leaf formulas play a role inside the derivation but are "used up" by the time the derivation reaches the conclusion. This corresponds to the practice, in actual reasoning, of introducing hypotheses which only remain in effect for a short while. For instance, in a proof by cases, we assume the truth of each of the disjuncts; in conditional proof, we assume the truth of the antecedent; in indirect proof, we assume the truth of the negation of the conclusion. This way of introducing hypothetical assumptions and then doing away with them in the service of establishing an intermediate step is a hallmark of natural deduction. The formulas at the leaves of a natural deduction derivation are called assumptions, and some of the rules of inference may "discharge" them. For instance, if we have a derivation of $\psi$ from some assumptions which include $\varphi$, then the $\rightarrow$Intro rule allows us to infer $\varphi \rightarrow \psi$ and discharge any assumption of the form $\varphi$. (To keep track of which assumptions are discharged at which inferences, we label the inference and the assumptions it discharges with a number.) The assumptions that remain undischarged at the end of the derivation are together sufficient for the truth of the conclusion, and so a derivation establishes that its undischarged assumptions entail its conclusion.

The relation $\Gamma \vdash \varphi$ based on natural deduction holds iff there is a derivation in which $\varphi$ is the last sentence in the tree, and every leaf which is undischarged is in $\Gamma$. $\varphi$ is a theorem in natural deduction iff there is a derivation in which $\varphi$ is the last sentence and all assumptions are discharged. For instance, here is a derivation that shows that $\vdash (\varphi \wedge \psi) \rightarrow \varphi$:

$$\cfrac{\cfrac{[\varphi \wedge \psi]^1}{\varphi}\,\wedge\text{Elim}}{(\varphi \wedge \psi) \rightarrow \varphi}\,{}^{1}\rightarrow\text{Intro}$$

The label 1 indicates that the assumption $\varphi \wedge \psi$ is discharged at the $\rightarrow$Intro inference.

A set $\Gamma$ is inconsistent iff $\Gamma \vdash \bot$ in natural deduction. The rule $\bot_I$ makes it so that from an inconsistent set, any sentence can be derived.

Natural deduction systems were developed by Gerhard Gentzen and Stanisław Jaśkowski in the 1930s, and later developed by Dag Prawitz and Frederic Fitch. Because its inferences mirror natural methods of proof, it is favored by philosophers. The versions developed by Fitch are often used in introductory logic textbooks. In the philosophy of logic, the rules of natural deduction have sometimes been taken to give the meanings of the logical operators ("proof-theoretic semantics").

## 6.4 Tableaux

While many derivation systems operate with arrangements of sentences, tableaux operate with signed formulas. A signed formula is a pair consisting of a truth value sign ($\mathbb{T}$ or $\mathbb{F}$) and a sentence

$$\mathbb{T}\varphi \text{ or } \mathbb{F}\varphi.$$

A tableau consists of signed formulas arranged in a downward-branching tree. It begins with a number of *assumptions* and continues with signed formulas which result from one of the signed formulas above it by applying one of the rules of inference. Each rule allows us to add one or more signed formulas to the end of a branch, or two signed formulas side by side—in this case a branch splits into two, with the two added signed formulas forming the ends of the two branches.

A rule applied to a complex signed formula results in the addition of signed formulas which are immediate sub-formulas. They come in pairs, one rule for each of the two signs. For instance, the $\wedge\mathbb{T}$ rule applies to $\mathbb{T}\varphi \wedge \psi$, and allows the addition of both the two signed formulas $\mathbb{T}\varphi$ and $\mathbb{T}\psi$ to the end of any branch containing $\mathbb{T}\varphi \wedge \psi$, and the rule $\varphi \wedge \psi\mathbb{F}$ allows a branch to be split by adding $\mathbb{F}\varphi$ and $\mathbb{F}\psi$ side-by-side. A tableau is closed if every one of its branches contains a matching pair of signed formulas $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$.

The $\vdash$ relation based on tableaux is defined as follows: $\Gamma \vdash \varphi$ iff there is some finite set $\Gamma_0 = \{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ such that there is a closed tableau for the assumptions

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

For instance, here is a closed tableau that shows that $\vdash (\varphi \wedge \psi) \rightarrow \varphi$:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\varphi$ | $\to\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\psi$ | $\to\mathbb{T}\,2$ |
| | $\otimes$ | |

A set $\Gamma$ is inconsistent in the tableau calculus if there is a closed tableau for assumptions

$$\{\mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

for some $\psi_i \in \Gamma$.

The sequent calculus was invented in the 1950s independently by Evert Beth and Jaakko Hintikka, and simplified and popularized by Raymond Smullyan. It is very easy to use, since constructing a tableau is a very systematic procedure. Because of the systematic nature of tableaux, they also lend themselves to implementation by computer. However, tableau is often hard to read and their connection to proofs are sometimes not easy to see. The approach is also quite general, and many different logics have tableau systems. Tableaux also help us to find structures that satisfy given (sets of) sentences: if the set is satisfiable, it won't have a closed tableau, i.e., any tableau will have an open branch. The satisfying structure can be "read off" an open branch, provided all rules it is possible to apply have been applied on that branch. There is also a very close connection to the sequent calculus: essentially, a closed tableau is a condensed derivation in the sequent calculus, written upside-down.

## 6.5 Axiomatic Derivations

Axiomatic derivations are the oldest and simplest logical derivation systems. Its derivations are simply sequences of sentences. A sequence of sentences conunts as a correct derivation if every sentence $\varphi$ in it satisfies one of the following conditions:

1. $\varphi$ is an axiom, or
2. $\varphi$ is an element of a given set $\Gamma$ of sentences, or
3. $\varphi$ is justified by a rule of inference.

To be an axiom, $\varphi$ has to have the form of on of a number of fixed sentence schemas. There are many sets of axiom schemas that provide a satisfactory (sound and complete) derivation system for first-order logic. Some are organized according to the connectives they govern, e.g., the schemas

$$\varphi \to (\psi \to \varphi) \qquad \psi \to (\psi \vee \chi) \qquad (\psi \wedge \chi) \to \psi$$

are common axioms that govern $\rightarrow$, $\vee$ and $\wedge$. Some axiom systems aim at a minimal number of axioms. Depending on the connectives that are taken as primitives, it is even possible to find axiom systems that consist of a single axiom.

A rule of inference is a conditional statement that gives a sufficient condition for a sentence in a derivation to be justified. Modus ponens is one very common such rule: it says that if $\varphi$ and $\varphi \rightarrow \psi$ are already justified, then $\psi$ is justified. This means that a line in a derivation containing the sentence $\psi$ is justified, provided that both $\varphi$ and $\varphi \rightarrow \psi$ (for some sentence $\varphi$) appear in the derivation before $\psi$.

The $\vdash$ relation based on axiomatic derivations is defined as follows: $\Gamma \vdash \varphi$ iff there is a derivation with the sentence $\varphi$ as its last formula (and $\Gamma$ is taken as the set of sentences in that derivation which are justified by (2) above). $\varphi$ is a theorem if $\varphi$ has a derivation where $\Gamma$ is empty, i.e., every sentence in the derivation is justfied either by (1) or (3). For instance, here is a derivation that shows that $\vdash \varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi))$:

1. $\psi \rightarrow (\psi \vee \varphi)$
2. $(\psi \rightarrow (\psi \vee \varphi)) \rightarrow (\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi)))$
3. $\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi))$

The sentence on line 1 is of the form of the axiom $\varphi \rightarrow (\varphi \vee \psi)$ (with the roles of $\varphi$ and $\psi$ reversed). The sentence on line 2 is of the form of the axiom $\varphi \rightarrow (\psi \rightarrow \varphi)$. Thus, both lines are justified. Line 3 is justified by modus ponens: if we abbreviate it as $\theta$, then line 2 has the form $\chi \rightarrow \theta$, where $\chi$ is $\psi \rightarrow (\psi \vee \varphi)$, i.e., line 1.

A set $\Gamma$ is inconsistent if $\Gamma \vdash \bot$. A complete axiom system will also prove that $\bot \rightarrow \varphi$ for any $\varphi$, and so if $\Gamma$ is inconsistent, then $\Gamma \vdash \varphi$ for any $\varphi$.

Systems of axiomatic derivations for logic were first given by Gottlob Frege in his 1879 *Begriffsschrift*, which for this reason is often considered the first work of modern logic. They were perfected in Alfred North Whitehead and Bertrand Russell's *Principia Mathematica* and by David Hilbert and his students in the 1920s. They are thus often called "Frege systems" or "Hilbert systems." They are very versatile in that it is often easy to find an axiomatic system for a logic. Because derivations have a very simple structure and only one or two inference rules, it is also relatively easy to prove things *about* them. However, they are very hard to use in practice, i.e., it is difficult to find and write proofs.

# Chapter 7

# The Sequent Calculus

This chapter presents Gentzen's standard sequent calculus LK for classical first-order logic. It could use more examples and exercises. To include or exclude material relevant to the sequent calculus as a proof system, use the "prfLK" tag.

## 7.1 Rules and Derivations

For the following, let $\Gamma, \Delta, \Pi, \Lambda$ represent finite sequences of sentences.

**Definition 7.1** (Sequent)**.** A *sequent* is an expression of the form

$$\Gamma \Rightarrow \Delta$$

where $\Gamma$ and $\Delta$ are finite (possibly empty) sequences of sentences of the language $\mathcal{L}$. $\Gamma$ is called the *antecedent*, while $\Delta$ is the *succedent*.

The intuitive idea behind a sequent is: if all of the sentences in the antecedent hold, then at least one of the sentences in the succedent holds. That is, if $\Gamma = \langle \varphi_1, \ldots, \varphi_m \rangle$ and $\Delta = \langle \psi_1, \ldots, \psi_n \rangle$, then $\Gamma \Rightarrow \Delta$ holds iff

$$(\varphi_1 \wedge \cdots \wedge \varphi_m) \rightarrow (\psi_1 \vee \cdots \vee \psi_n)$$

holds. There are two special cases: where $\Gamma$ is empty and when $\Delta$ is empty. When $\Gamma$ is empty, i.e., $m = 0$, $\Rightarrow \Delta$ holds iff $\psi_1 \vee \cdots \vee \psi_n$ holds. When $\Delta$ is empty, i.e., $n = 0$, $\Gamma \Rightarrow$ holds iff $\neg(\varphi_1 \wedge \cdots \wedge \varphi_m)$ does. We say a sequent is valid iff the corresponding sentence is valid.

If $\Gamma$ is a sequence of sentences, we write $\Gamma, \varphi$ for the result of appending $\varphi$ to the right end of $\Gamma$ (and $\varphi, \Gamma$ for the result of appending $\varphi$ to the left end of $\Gamma$). If $\Delta$ is a sequence of sentences also, then $\Gamma, \Delta$ is the concatenation of the two sequences.

**Definition 7.2** (Initial Sequent)**.** An *initial sequent* is a sequent of one of the following forms:

1. $\varphi \Rightarrow \varphi$
2. $\bot \Rightarrow$

for any sentence $\varphi$ in the language.

Derivations in the sequent calculus are certain trees of sequents, where the topmost sequents are initial sequents, and if a sequent stands below one or two other sequents, it must follow correctly by a rule of inference. The rules for **LK** are divided into two main types: *logical* rules and *structural* rules. The logical rules are named for the main operator of the sentence containing $\varphi$ and/or $\psi$ in the lower sequent. Each one comes in two versions, one for inferring a sequent with the sentence containg the logical operator on the left, and one with the sentence on the right.

## 7.2 Propositional Rules

**Rules for $\neg$**

$$\frac{\Gamma \Rightarrow \Delta, \varphi}{\neg\varphi, \Gamma \Rightarrow \Delta}\,\neg\mathrm{L} \qquad \frac{\varphi, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\varphi}\,\neg\mathrm{R}$$

**Rules for $\wedge$**

$$\frac{\varphi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\,\wedge\mathrm{L} \qquad \frac{\psi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\,\wedge\mathrm{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}\,\wedge\mathrm{R}$$

**Rules for $\vee$**

$$\frac{\varphi, \Gamma \Rightarrow \Delta \qquad \psi, \Gamma \Rightarrow \Delta}{\varphi \vee \psi, \Gamma \Rightarrow \Delta}\,\vee\mathrm{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\,\vee\mathrm{R} \qquad \frac{\Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\,\vee\mathrm{R}$$

**Rules for $\rightarrow$**

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \psi, \Pi \Rightarrow \Lambda}{\varphi \rightarrow \psi, \Gamma, \Pi \Rightarrow \Delta, \Lambda} \rightarrow\text{L} \qquad \frac{\varphi, \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi} \rightarrow\text{R}$$

## 7.3 Structural Rules

We also need a few rules that allow us to rearrange sentences in the left and right side of a sequent. Since the logical rules require that the sentences in the premise which the rule acts upon stand either to the far left or to the far right, we need an "exchange" rule that allows us to move sentences to the right position. It's also important sometimes to be able to combine two identical sentences into one, and to add a sentence on either side.

**Weakening**

$$\frac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta} \text{WL} \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \varphi} \text{WR}$$

**Contraction**

$$\frac{\varphi, \varphi, \Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta} \text{CL} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi} \text{CR}$$

**Exchange**

$$\frac{\Gamma, \varphi, \psi, \Pi \Rightarrow \Delta}{\Gamma, \psi, \varphi, \Pi \Rightarrow \Delta} \text{XL} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \psi, \Lambda}{\Gamma \Rightarrow \Delta, \psi, \varphi, \Lambda} \text{XR}$$

A series of weakening, contraction, and exchange inferences will often be indicated by double inference lines.

The following rule, called "cut," is not strictly speaking necessary, but makes it a lot easier to reuse and combine derivations.

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \varphi, \Pi \Rightarrow \Lambda}{\Gamma, \Pi \Rightarrow \Delta, \Lambda} \text{Cut}$$

## 7.4 Derivations

We've said what an initial sequent looks like, and we've given the rules of inference. Derivations in the sequent calculus are inductively generated from these: each derivation either is an initial sequent on its own, or consists of one or two derivations followed by an inference.

**Definition 7.3** (**LK** derivation)**.** An **LK**-*derivation* of a sequent $S$ is a tree of sequents satisfying the following conditions:

1. The topmost sequents of the tree are initial sequents.
2. The bottommost sequent of the tree is $S$.
3. Every sequent in the tree except $S$ is a premise of a correct application of an inference rule whose conclusion stands directly below that sequent in the tree.

We then say that $S$ is the *end-sequent* of the derivation and that $S$ is *derivable in* **LK** (or **LK**-derivable).

**Example 7.4.** Every initial sequent, e.g., $\chi \Rightarrow \chi$ is a derivation. We can obtain a new derivation from this by applying, say, the WL rule,

$$\frac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\,\mathrm{WL}$$

The rule, however, is meant to be general: we can replace the $\varphi$ in the rule with any sentence, e.g., also with $\theta$. If the premise matches our initial sequent $\chi \Rightarrow \chi$, that means that both $\Gamma$ and $\Delta$ are just $\chi$, and the conclusion would then be $\theta, \chi \Rightarrow \chi$. So, the following is a derivation:

$$\frac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\mathrm{WL}$$

We can now apply another rule, say XL, which allows us to switch two sentences on the left. So, the following is also a correct derivation:

$$\dfrac{\dfrac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\mathrm{WL}}{\chi, \theta \Rightarrow \chi}\,\mathrm{XL}$$

In this application of the rule, which was given as

$$\frac{\Gamma, \varphi, \psi, \Pi \Rightarrow \Delta}{\Gamma, \psi, \varphi, \Pi \Rightarrow \Delta,}\,\mathrm{XL}$$

both $\Gamma$ and $\Pi$ were empty, $\Delta$ is $\chi$, and the roles of $\varphi$ and $\psi$ are played by $\theta$ and $\chi$, respectively. In much the same way, we also see that

$$\dfrac{\theta \Rightarrow \theta}{\chi, \theta \Rightarrow \theta}\,\text{WL}$$

is a derivation. Now we can take these two derivations, and combine them using $\wedge$R. That rule was

$$\dfrac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}\,\wedge\text{R}$$

In our case, the premises must match the last sequents of the derivations ending in the premises. That means that $\Gamma$ is $\chi, \theta$, $\Delta$ is empty, $\varphi$ is $\chi$ and $\psi$ is $\theta$. So the conclusion, if the inference should be correct, is $\chi, \theta \Rightarrow \chi \wedge \theta$. Of course, we can also reverse the premises, then $\varphi$ would be $\theta$ and $\psi$ would be $\chi$. So both of the following are correct derivations.

$$\dfrac{\dfrac{\dfrac{\dfrac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\text{WL}}{\chi, \theta \Rightarrow \chi}\,\text{XL} \qquad \dfrac{\theta \Rightarrow \theta}{\chi, \theta \Rightarrow \theta}\,\text{WL}}{\chi, \theta \Rightarrow \chi \wedge \theta}\,\wedge\text{R} \qquad \dfrac{\dfrac{\dfrac{\theta \Rightarrow \theta}{\chi, \theta \Rightarrow \theta}\,\text{WL} \qquad \dfrac{\dfrac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\text{WL}}{\chi, \theta \Rightarrow \chi}\,\text{XL}}{\chi, \theta \Rightarrow \theta \wedge \chi}\,\wedge\text{R}$$

## 7.5 Examples of Derivations

**Example 7.5.** Give an **LK**-derivation for the sequent $\varphi \wedge \psi \Rightarrow \varphi$.

We begin by writing the desired end-sequent at the bottom of the derivation.

$$\dfrac{}{\varphi \wedge \psi \Rightarrow \varphi}$$

Next, we need to figure out what kind of inference could have a lower sequent of this form. This could be a structural rule, but it is a good idea to start by looking for a logical rule. The only logical connective occurring in the lower sequent is $\wedge$, so we're looking for an $\wedge$ rule, and since the $\wedge$ symbol occurs in the antecedent, we're looking at the $\wedge$L rule.

$$\dfrac{}{\varphi \wedge \psi \Rightarrow \varphi}\,\wedge\text{L}$$

There are two options for what could have been the upper sequent of the $\wedge$L inference: we could have an upper sequent of $\varphi \Rightarrow \varphi$, or of $\psi \Rightarrow \varphi$. Clearly, $\varphi \Rightarrow \varphi$ is an initial sequent (which is a good thing), while $\psi \Rightarrow \varphi$ is not derivable in general. We fill in the upper sequent:

$$\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi}\,\wedge\text{L}$$

We now have a correct **LK**-derivation of the sequent $\varphi \wedge \psi \Rightarrow \varphi$.

**Example 7.6.** Give an **LK**-derivation for the sequent $\neg\varphi \vee \psi \Rightarrow \varphi \rightarrow \psi$.

Begin by writing the desired end-sequent at the bottom of the derivation.

$$\frac{}{\neg\varphi \vee \psi \;\Rightarrow\; \varphi \to \psi}$$

To find a logical rule that could give us this end-sequent, we look at the logical connectives in the end-sequent: $\neg$, $\vee$, and $\to$. We only care at the moment about $\vee$ and $\to$ because they are main operators of sentences in the end-sequent, while $\neg$ is inside the scope of another connective, so we will take care of it later. Our options for logical rules for the final inference are therefore the $\vee$L rule and the $\to$R rule. We could pick either rule, really, but let's pick the $\to$R rule (if for no reason other than it allows us to put off splitting into two branches). According to the form of $\to$R inferences which can yield the lower sequent, this must look like:

$$\dfrac{\dfrac{}{\varphi, \neg\varphi \vee \psi \;\Rightarrow\; \psi}}{\neg\varphi \vee \psi \;\Rightarrow\; \varphi \to \psi}\,{\to}\mathrm{R}$$

If we move $\neg\varphi \vee \psi$ to the outside of the antecedent, we can apply the $\vee$L rule. According to the schema, this must split into two upper sequents as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\varphi, \varphi \;\Rightarrow\; \psi} \qquad \dfrac{}{\psi, \varphi \;\Rightarrow\; \psi}}{\neg\varphi \vee \psi, \varphi \;\Rightarrow\; \psi}\,{\vee}\mathrm{L}}{\varphi, \neg\varphi \vee \psi \;\Rightarrow\; \psi}\,\mathrm{XR}}{\neg\varphi \vee \psi \;\Rightarrow\; \varphi \to \psi}\,{\to}\mathrm{R}$$

Remember that we are trying to wind our way up to initial sequents; we seem to be pretty close! The right branch is just one weakening and one exchange away from an initial sequent and then it is done:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\varphi, \varphi \;\Rightarrow\; \psi} \qquad \dfrac{\dfrac{\dfrac{}{\psi \;\Rightarrow\; \psi}}{\varphi, \psi \;\Rightarrow\; \psi}\,\mathrm{WL}}{\psi, \varphi \;\Rightarrow\; \psi}\,\mathrm{XL}}{\neg\varphi \vee \psi, \varphi \;\Rightarrow\; \psi}\,{\vee}\mathrm{L}}{\varphi, \neg\varphi \vee \psi \;\Rightarrow\; \psi}\,\mathrm{XR}}{\neg\varphi \vee \psi \;\Rightarrow\; \varphi \to \psi}\,{\to}\mathrm{R}$$

Now looking at the left branch, the only logical connective in any sentence is the $\neg$ symbol in the antecedent sentences, so we're looking at an instance of the $\neg$L rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\varphi \;\Rightarrow\; \psi, \varphi}}{\neg\varphi, \varphi \;\Rightarrow\; \psi}\,\neg\mathrm{L} \qquad \dfrac{\dfrac{\dfrac{}{\psi \;\Rightarrow\; \psi}}{\varphi, \psi \;\Rightarrow\; \psi}\,\mathrm{WL}}{\psi, \varphi \;\Rightarrow\; \psi}\,\mathrm{XL}}{\neg\varphi \vee \psi, \varphi \;\Rightarrow\; \psi}\,{\vee}\mathrm{L}}{\varphi, \neg\varphi \vee \psi \;\Rightarrow\; \psi}\,\mathrm{XR}}{\neg\varphi \vee \psi \;\Rightarrow\; \varphi \to \psi}\,{\to}\mathrm{R}$$

Similarly to how we finished off the right branch, we are just one weakening and one exchange away from finishing off this left branch as well.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\varphi \Rightarrow \varphi, \psi}\,\mathrm{WR}}{\varphi \Rightarrow \psi, \varphi}\,\mathrm{XR}}{\neg\varphi, \varphi \Rightarrow \psi}\,\neg\mathrm{L} \qquad \dfrac{\dfrac{\psi \Rightarrow \psi}{\varphi, \psi \Rightarrow \psi}\,\mathrm{WL}}{\psi, \varphi \Rightarrow \psi}\,\mathrm{XL}}{\neg\varphi \vee \psi, \varphi \Rightarrow \psi}\,\vee\mathrm{L}}{\varphi, \neg\varphi \vee \psi \Rightarrow \psi}\,\mathrm{XR}}{\neg\varphi \vee \psi \Rightarrow \varphi \to \psi}\,\to\mathrm{R}$$

**Example 7.7.** Give an **LK**-derivation of the sequent $\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)$

Using the techniques from above, we start by writing the desired end-sequent at the bottom.

$$\dfrac{}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}$$

The available main connectives of sentences in the end-sequent are the $\vee$ symbol and the $\neg$ symbol. It would work to apply either the $\vee$L or the $\neg$R rule here, but we start with the $\neg$R rule because it avoids splitting up into two branches for a moment:

$$\dfrac{\dfrac{}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}\,\neg\mathrm{R}$$

Now we have a choice of whether to look at the $\wedge$L or the $\vee$L rule. Let's see what happens when we apply the $\wedge$L rule: we have a choice to start with either the sequent $\varphi, \neg\varphi \vee \psi \Rightarrow$ or the sequent $\psi, \neg\varphi \vee \psi \Rightarrow$. Since the proof is symmetric with regards to $\varphi$ and $\psi$, let's go with the former:

$$\dfrac{\dfrac{\dfrac{}{\varphi, \neg\varphi \vee \neg\psi \Rightarrow}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}\,\wedge\mathrm{L}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}\,\neg\mathrm{R}$$

Continuing to fill in the derivation, we see that we run into a problem:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\,\neg\mathrm{L} \qquad \dfrac{\dfrac{}{\varphi \Rightarrow \psi}\,?}{\neg\psi, \varphi \Rightarrow}\,\neg\mathrm{L}}{\neg\varphi \vee \neg\psi, \varphi \Rightarrow}\,\vee\mathrm{L}}{\varphi, \neg\varphi \vee \neg\psi \Rightarrow}\,\mathrm{XL}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}\,\wedge\mathrm{L}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}\,\neg\mathrm{R}$$

The top of the right branch cannot be reduced any further, and it cannot be brought by way of structural inferences to an initial sequent, so this is not the right path to take. So clearly, it was a mistake to apply the $\wedge$L rule above. Going back to what we had before and carrying out the $\vee$L rule instead, we get

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\varphi, \varphi \wedge \psi \Rightarrow} \quad \dfrac{}{\neg\psi, \varphi \wedge \psi \Rightarrow}}{\neg\varphi \vee \neg\psi, \varphi \wedge \psi \Rightarrow} \vee\text{L}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow} \text{XL}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)} \neg\text{R}$$

Completing each branch as we've done before, we get

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow \varphi}}{\varphi \wedge \psi \Rightarrow \varphi} \wedge\text{L}}{\neg\varphi, \varphi \wedge \psi \Rightarrow} \neg\text{L} \quad \dfrac{\dfrac{\dfrac{}{\psi \Rightarrow \psi}}{\varphi \wedge \psi \Rightarrow \psi} \wedge\text{L}}{\neg\psi, \varphi \wedge \psi \Rightarrow} \neg\text{L}}{\neg\varphi \vee \neg\psi, \varphi \wedge \psi \Rightarrow} \vee\text{L}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow} \text{XL}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)} \neg\text{R}$$

(We could have carried out the $\wedge$ rules lower than the $\neg$ rules in these steps and still obtained a correct derivation).

**Example 7.8.** So far we haven't used the contraction rule, but it is sometimes required. Here's an example where that happens. Suppose we want to prove $\Rightarrow A \vee \neg\varphi$. Applying $\vee$R backwards would give us one of these two derivations:

$$\dfrac{\dfrac{}{\Rightarrow \varphi}}{\Rightarrow \varphi \vee \neg\varphi} \vee\text{R} \qquad \dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow}}{\Rightarrow \neg\varphi} \neg\text{R}}{\Rightarrow \varphi \vee \neg\varphi} \vee\text{R}$$

Neither of these of course ends in an initial sequent. The trick is to realize that the contraction rule allows us to combine two copies of a sentence into one—and when we're searching for a proof, i.e., going from bottom to top, we can keep a copy of $\varphi \vee \neg\varphi$ in the premise, e.g.,

$$\dfrac{\dfrac{\dfrac{}{\Rightarrow \varphi \vee \neg\varphi, \varphi}}{\Rightarrow \varphi \vee \neg\varphi, \varphi \vee \neg\varphi} \vee\text{R}}{\Rightarrow \varphi \vee \neg\varphi} \text{CR}$$

Now we can apply $\vee$R a second time, and also get $\neg\varphi$, which leads to a complete derivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\Rightarrow \varphi, \neg\varphi}\neg\mathrm{R}}{\Rightarrow \varphi, \varphi \vee \neg\varphi}\vee\mathrm{R}}{\Rightarrow \varphi \vee \neg\varphi, \varphi}\mathrm{XR}}{\Rightarrow \varphi \vee \neg\varphi, \varphi \vee \neg\varphi}\vee\mathrm{R}}{\Rightarrow \varphi \vee \neg\varphi}\mathrm{CR}$$

This section collects the definitions of the provability relation and consistency for natural deduction.

## 7.6 Proof-Theoretic Notions

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the derivability or non-derivability of certain sequents. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorem*.

**Definition 7.9** (Theorems)**.** A sentence $\varphi$ is a *theorem* if there is a derivation in **LK** of the sequent $\Rightarrow \varphi$. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 7.10** (Derivability)**.** A sentence $\varphi$ is *derivable from* a set of sentences $\Gamma$, $\Gamma \vdash \varphi$, iff there is a finite subset $\Gamma_0 \subseteq \Gamma$ and a sequence $\Gamma_0'$ of the sentences in $\Gamma_0$ such that **LK** derives $\Gamma_0' \Rightarrow \varphi$. If $\varphi$ is not derivable from $\Gamma$ we write $\Gamma \nvdash \varphi$.

Because of the contraction, weakening, and exchange rules, the order and number of sentences in $\Gamma_0'$ does not matter: if a sequent $\Gamma_0' \Rightarrow \varphi$ is derivable, then so is $\Gamma_0'' \Rightarrow \varphi$ for any $\Gamma_0''$ that contains the same sentences as $\Gamma_0'$. For instance, if $\Gamma_0 = \{\psi, \chi\}$ then both $\Gamma_0' = \langle \psi, \psi, \chi \rangle$ and $\Gamma_0'' = \langle \chi, \chi, \psi \rangle$ are sequences containing just the sentences in $\Gamma_0$. If a sequent containing one is derivable, so is the other, e.g.:

$$\dfrac{\dfrac{\dfrac{\vdots}{\psi, \psi, \chi \Rightarrow \varphi}}{\psi, \chi \Rightarrow \varphi}\mathrm{CL}}{\chi, \psi \Rightarrow \varphi}\mathrm{XL}}{\chi, \chi, \psi \Rightarrow \varphi}\mathrm{WL}$$

From now on we'll say that if $\Gamma_0$ is a finite set of sentences then $\Gamma_0 \Rightarrow \varphi$ is any sequent where the antecedent is a sequence of sentences in $\Gamma_0$ and tacitly include contractions, exchanges, and weakenings if necessary.

**Definition 7.11** (Consistency)**.** A set of sentences $\Gamma$ is *inconsistent* iff there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that **LK** derives $\Gamma_0 \Rightarrow$ . If $\Gamma$ is not inconsistent, i.e., if for every finite $\Gamma_0 \subseteq \Gamma$, **LK** does not derive $\Gamma_0 \Rightarrow$ , we say it is *consistent*.

**Proposition 7.12** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* The initial sequent $\varphi \Rightarrow \varphi$ is derivable, and $\{\varphi\} \subseteq \Gamma$. □

**Proposition 7.13** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Suppose $\Gamma \vdash \varphi$, i.e., there is a finite $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \Rightarrow \varphi$ is derivable. Since $\Gamma \subseteq \Delta$, then $\Gamma_0$ is also a finite subset of $\Delta$. The derivation of $\Gamma_0 \Rightarrow \varphi$ thus also shows $\Delta \vdash \varphi$. □

**Proposition 7.14** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* If $\Gamma \vdash \varphi$, there is a finite $\Gamma_0 \subseteq \Gamma$ and a derivation $\pi_0$ of $\Gamma_0 \Rightarrow \varphi$. If $\{\varphi\} \cup \Delta \vdash \psi$, then for some finite subset $\Delta_0 \subseteq \Delta$, there is a derivation $\pi_1$ of $\varphi, \Delta_0 \Rightarrow \psi$. Consider the following derivation:

$$\dfrac{\begin{array}{cc}\vdots\,\pi_0 & \vdots\,\pi_1\\ \Gamma_0 \Rightarrow \varphi & \varphi, \Delta_0 \Rightarrow \psi\end{array}}{\Gamma_0, \Delta_0 \Rightarrow \psi}\,\text{Cut}$$

Since $\Gamma_0 \cup \Delta_0 \subseteq \Gamma \cup \Delta$, this shows $\Gamma \cup \Delta \vdash \psi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 7.15.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every sentence $\varphi$.*

*Proof.* Exercise. □

**Proposition 7.16** (Compactness)**.** 1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*

2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that the sequent $\Gamma_0 \Rightarrow \varphi$ has a derivation. Consequently, $\Gamma_0 \vdash \varphi$.

2. If $\Gamma$ is inconsistent, there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that **LK** derives $\Gamma_0 \Rightarrow$ . But then $\Gamma_0$ is a finite subset of $\Gamma$ that is inconsistent.

□

## 7.7 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 7.17.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are finite $\Gamma_0$ and $\Gamma_1 \subseteq \Gamma$ such that **LK** derives $\Gamma_0 \Rightarrow \varphi$ and $\varphi, \Gamma_1 \Rightarrow$ . Let the **LK**-derivation of $\Gamma_0 \Rightarrow \varphi$ be $\pi_0$ and the **LK**-derivation of $\Gamma_1, \varphi \Rightarrow$ be $\pi_1$. We can then derive

$$\dfrac{\begin{array}{cc}\vdots\,\pi_0 & \vdots\,\pi_1\\ \Gamma_0 \Rightarrow \varphi & \varphi, \Gamma_1 \Rightarrow\end{array}}{\Gamma_0, \Gamma_1 \Rightarrow}\ \text{Cut}$$

Since $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$, hence $\Gamma$ is inconsistent. □

**Proposition 7.18.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$, i.e., there is a derivation $\pi_0$ of $\Gamma \Rightarrow \varphi$. By adding a $\neg$L rule, we obtain a derivation of $\neg\varphi, \Gamma \Rightarrow$ , i.e., $\Gamma \cup \{\neg\varphi\}$ is inconsistent.

If $\Gamma \cup \{\neg A\}$ is inconsistent, there is a derivation $\pi_1$ of $\neg\varphi, \Gamma \Rightarrow$ . The following is a derivation of $\Gamma \Rightarrow \varphi$:

$$\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\Rightarrow \varphi, \neg\varphi}\,\neg\text{R} \qquad \begin{array}{c}\vdots\,\pi_1\\ \neg\varphi, \Gamma \Rightarrow\end{array}}{\Gamma \Rightarrow \varphi}\ \text{Cut}$$

□

**Proposition 7.19.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* Suppose $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$. Then there is a derivation $\pi$ of a sequent $\Gamma_0 \Rightarrow \varphi$. The sequent $\neg\varphi, \Gamma_0 \Rightarrow$ is also derivable:

$$\dfrac{\begin{array}{cc}\vdots\,\pi & \dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\,\neg\text{L}}{\varphi, \neg\varphi \Rightarrow}\,\text{XL}\\ \Gamma_0 \Rightarrow \varphi & \end{array}}{\Gamma, \neg\varphi \Rightarrow}\ \text{Cut}$$

Since $\neg\varphi \in \Gamma$ and $\Gamma_0 \subseteq \Gamma$, this shows that $\Gamma$ is inconsistent. □

**Proposition 7.20.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are finite sets $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$ and **LK**-derivations $\pi_0$ and $\pi_1$ of $\varphi, \Gamma_0 \Rightarrow$ and $\neg\varphi, \Gamma_1 \Rightarrow$, respectively. We can then derive

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \pi_0\\ \vdots\\ \varphi, \Gamma_0 \Rightarrow\end{array}}{\Gamma_0 \Rightarrow \neg\varphi}\neg\mathrm{R} \qquad \begin{array}{c}\vdots\\ \pi_1\\ \vdots\\ \neg\varphi, \Gamma_1 \Rightarrow\end{array}}{\Gamma_0, \Gamma_1 \Rightarrow}\,\mathrm{Cut}$$

Since $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$. Hence $\Gamma$ is inconsistent. □

## 7.8 Derivability and the Propositional Connectives

**Proposition 7.21.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$.*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. Both sequents $\varphi \wedge \psi \Rightarrow \varphi$ and $\varphi \wedge \psi \Rightarrow \psi$ are derivable:

$$\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi}\wedge\mathrm{L} \qquad \dfrac{\psi \Rightarrow \psi}{\varphi \wedge \psi \Rightarrow \psi}\wedge\mathrm{L}$$

2. Here is a derivation of the sequent $\varphi, \psi \Rightarrow \varphi \wedge \psi$:

$$\dfrac{\varphi \Rightarrow \varphi \qquad \psi \Rightarrow \psi}{\varphi, \psi \Rightarrow \varphi \wedge \psi}\wedge\mathrm{R}$$

□

**Proposition 7.22.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. We give a derivation of the sequent $\varphi \vee \psi, \neg\varphi, \neg\psi \Rightarrow$:

$$\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\neg\mathrm{L}}{\varphi, \neg\varphi, \neg\psi \Rightarrow} \qquad \dfrac{\dfrac{\psi \Rightarrow \psi}{\neg\psi, \psi \Rightarrow}\neg\mathrm{L}}{\psi, \neg\varphi, \neg\psi \Rightarrow}}{\varphi \vee \psi, \neg\varphi, \neg\psi \Rightarrow}\vee\mathrm{L}$$

(Recall that double inference lines indicate several weakening, contraction, and exchange inferences.)

2. Both sequents $\varphi \Rightarrow \varphi \vee \psi$ and $\psi \Rightarrow \varphi \vee \psi$ have derivations:

$$\frac{\varphi \Rightarrow \varphi}{\varphi \Rightarrow \varphi \vee \psi} \vee\text{R} \qquad \frac{\psi \Rightarrow \psi}{\psi \Rightarrow \varphi \vee \psi} \vee\text{R}$$

□

**Proposition 7.23.** *1.* $\varphi, \varphi \to \psi \vdash \psi$.

*2. Both* $\neg\varphi \vdash \varphi \to \psi$ *and* $\psi \vdash \varphi \to \psi$.

*Proof.* 1. The sequent $\varphi \to \psi, \varphi \Rightarrow \psi$ is derivable:

$$\frac{\varphi \Rightarrow \varphi \qquad \psi \Rightarrow \psi}{\varphi \to \psi, \varphi \Rightarrow \psi} \to\text{L}$$

2. Both sequents $\neg\varphi \Rightarrow \varphi \to \psi$ and $\psi \Rightarrow \varphi \to \psi$ are derivable:

$$\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\neg\text{L}}{\varphi, \neg\varphi \Rightarrow}\text{XL}}{\varphi, \neg\varphi \Rightarrow \psi}\text{WR}}{\neg\varphi \Rightarrow \varphi \to \psi} \to\text{R} \qquad \dfrac{\dfrac{\psi \Rightarrow \psi}{\varphi, \psi \Rightarrow \psi}\text{WL}}{\psi \Rightarrow \varphi \to \psi} \to\text{R}$$

□

## 7.9 Soundness

A derivation system, such as the sequent calculus, is *sound* if it cannot derive things that do not actually hold. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable $\varphi$ is a tautology;
2. if a sentence is derivable from some others, it is also a consequence of them;
3. if a set of sentences is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

Because all these proof-theoretic properties are defined via derivability in the sequent calculus of certain sequents, proving (1)–(3) above requires proving something about the semantic properties of derivable sequents. We will first define what it means for a sequent to be *valid*, and then show that every derivable sequent is valid. (1)–(3) then follow as corollaries from this result.

**Definition 7.24.** A valuation $\mathfrak{v}$ *satisfies* a sequent $\Gamma \Rightarrow \Delta$ iff either $\mathfrak{v} \nvDash \varphi$ for some $\varphi \in \Gamma$ or $\mathfrak{v} \vDash \varphi$ for some $\varphi \in \Delta$.

A sequent is *valid* iff every valuation $\mathfrak{v}$ satisfies it.

**Theorem 7.25** (Soundness)**.** *If* **LK** *derives $\Theta \Rightarrow \Xi$, then $\Theta \Rightarrow \Xi$ is valid.*

*Proof.* Let $\pi$ be a derivation of $\Theta \Rightarrow \Xi$. We proceed by induction on the number of inferences $n$ in $\pi$.

If the number of inferences is 0, then $\pi$ consists only of an initial sequent. Every initial sequent $\varphi \Rightarrow \varphi$ is obviously valid, since for every $\mathfrak{v}$, either $\mathfrak{v} \nvDash \varphi$ or $\mathfrak{v} \vDash \varphi$.

If the number of inferences is greater than 0, we distinguish cases according to the type of the lowermost inference. By induction hypothesis, we can assume that the premises of that inference are valid, since the number of inferences in the proof of any premise is smaller than $n$.

First, we consider the possible inferences with only one premise.

1. The last inference is a weakening. Then $\Theta \Rightarrow \Xi$ is either $A, \Gamma \Rightarrow \Delta$ (if the last inference is WL) or $\Gamma \Rightarrow \Delta, \varphi$ (if it's WR), and the derivation ends in one of

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta\end{array}}{\varphi, \Gamma \Rightarrow \Delta}\,\mathrm{WL} \qquad \dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta\end{array}}{\Gamma \Rightarrow \Delta, \varphi}\,\mathrm{WR}$$

   By induction hypothesis, $\Gamma \Rightarrow \Delta$ is valid, i.e., for every valuation $\mathfrak{v}$, either there is some $\chi \in \Gamma$ such that $\mathfrak{v} \nvDash \chi$ or there is some $\chi \in \Delta$ such that $\mathfrak{v} \vDash \chi$.

   If $\mathfrak{v} \nvDash \chi$ for some $\chi \in \Gamma$, then $\chi \in \Theta$ as well since $\Theta = \varphi, \Gamma$, and so $\mathfrak{v} \nvDash \chi$ for some $\chi \in \Theta$. Similarly, if $\mathfrak{v} \vDash \chi$ for some $\chi \in \Delta$, as $\chi \in \Xi$, $\mathfrak{v} \vDash \chi$ for some $\chi \in \Xi$. Consequently, $\Theta \Rightarrow \Xi$ is valid.

2. The last inference is ¬L: Then the premise of the last inference is $\Gamma \Rightarrow \Delta, \varphi$ and the conclusion is $\neg\varphi, \Gamma \Rightarrow \Delta$, i.e., the derivation ends in

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi\end{array}}{\neg\varphi, \Gamma \Rightarrow \Delta}\,\neg\mathrm{L}$$

   and $\Theta = \neg\varphi, \Gamma$ while $\Xi = \Delta$.

   The induction hypothesis tells us that $\Gamma \Rightarrow \Delta, \varphi$ is valid, i.e., for every $\mathfrak{v}$, either (a) for some $\chi \in \Gamma$, $\mathfrak{v} \nvDash \chi$, or (b) for some $\chi \in \Delta$, $\mathfrak{v} \vDash \chi$, or (c) $\mathfrak{v} \vDash \varphi$.

We want to show that $\Theta \Rightarrow \Xi$ is also valid. Let $\mathfrak{v}$ be a valuation. If (a) holds, then there is $\chi \in \Gamma$ so that $\mathfrak{v} \nvDash \varphi$, but $\varphi \in \Theta$ as well. If (b) holds, there is $\chi \in \Delta$ such that $\mathfrak{v} \vDash \chi$, but $\chi \in \Xi$ as well. Finally, if $\mathfrak{v} \vDash \varphi$, then $\mathfrak{v} \nvDash \neg\varphi$. Since $\neg\varphi \in \Theta$, there is $\chi \in \Theta$ such that $\mathfrak{v} \nvDash \chi$. Consequently, $\Theta \Rightarrow \Xi$ is valid.

3. The last inference is $\neg$R: Exercise.

4. The last inference is $\wedge$L: There are two variants: $\varphi \wedge \psi$ may be inferred on the left from $\varphi$ or from $\psi$ on the left side of the premise. In the first case, the $\pi$ ends in

$$\frac{\begin{array}{c}\vdots\\ \varphi, \Gamma \Rightarrow \Delta\end{array}}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\ \wedge\mathrm{L}$$

and $\Theta = \varphi \wedge \psi, \Gamma$ while $\Xi = \Delta$. Consider a valuation $\mathfrak{v}$. Since by induction hypothesis, $\varphi, \Gamma \Rightarrow \Delta$ is valid, (a) $\mathfrak{v} \nvDash \varphi$, (b) $\mathfrak{v} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{v} \vDash \chi$ for some $\chi \in \Delta$. In case (a), $\mathfrak{v} \nvDash \varphi \wedge \psi$, so there is $\chi \in \Theta$ (namely, $\varphi \wedge \psi$) such that $\mathfrak{v} \nvDash \chi$. In case (b), there is $\chi \in \Gamma$ such that $\mathfrak{v} \nvDash \chi$, and $\chi \in \Theta$ as well. In case (c), there is $\chi \in \Delta$ such that $\mathfrak{v} \vDash \chi$, and $\chi \in \Xi$ as well since $\Xi = \Delta$. So in each case, $\mathfrak{v}$ satisfies $\varphi \wedge \psi, \Gamma \Rightarrow \Delta$. Since $\mathfrak{v}$ was arbitrary, $\Gamma \Rightarrow \Delta$ is valid. The case where $\varphi \wedge \psi$ is inferred from $\psi$ is handled the same, changing $\varphi$ to $\psi$.

5. The last inference is $\vee$R: There are two variants: $\varphi \vee \psi$ may be inferred on the right from $\varphi$ or from $\psi$ on the right side of the premise. In the first case, $\pi$ ends in

$$\frac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi\end{array}}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\ \vee\mathrm{R}$$

Now $\Theta = \Gamma$ and $\Xi = \Delta, \varphi \vee \psi$. Consider a valuation $\mathfrak{v}$. Since $\Gamma \Rightarrow \Delta, \varphi$ is valid, (a) $\mathfrak{v} \vDash \varphi$, (b) $\mathfrak{v} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{v} \vDash \chi$ for some $\chi \in \Delta$. In case (a), $\mathfrak{v} \vDash \varphi \vee \psi$. In case (b), there is $\chi \in \Gamma$ such that $\mathfrak{v} \nvDash \chi$. In case (c), there is $\chi \in \Delta$ such that $\mathfrak{v} \vDash \chi$. So in each case, $\mathfrak{v}$ satisfies $\Gamma \Rightarrow \Delta, \varphi \vee \psi$, i.e., $\Theta \Rightarrow \Xi$. Since $\mathfrak{v}$ was arbitrary, $\Theta \Rightarrow \Xi$ is valid. The case where $\varphi \vee \psi$ is inferred from $\psi$ is handled the same, changing $\varphi$ to $\psi$.

6. The last inference is $\rightarrow$R: Then $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \varphi, \Gamma \Rightarrow \Delta, \varphi\end{array}}{\Gamma \Rightarrow \Delta, \varphi \to \psi} \to\text{R}$$

Again, the induction hypothesis says that the premise is valid; we want to show that the conclusion is valid as well. Let $\mathfrak{v}$ be arbitrary. Since $\varphi, \Gamma \Rightarrow \Delta, \psi$ is valid, at least one of the following cases obtains: (a) $\mathfrak{v} \nvDash \varphi$, (b) $\mathfrak{v} \vDash \psi$, (c) $\mathfrak{v} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{v} \vDash \chi$ for some $\chi \in \Delta$. In cases (a) and (b), $\mathfrak{v} \vDash \varphi \to \psi$ and so there is a $\chi \in \Delta, \varphi \to \psi$ such that $\mathfrak{v} \vDash \chi$. In case (c), for some $\chi \in \Gamma$, $\mathfrak{v} \nvDash \chi$. In case (d), for some $\chi \in \Delta$, $\mathfrak{v} \vDash \chi$. In each case, $\mathfrak{v}$ satisfies $\Gamma \Rightarrow \Delta, \varphi \to \psi$. Since $\mathfrak{v}$ was arbitrary, $\Gamma \Rightarrow \Delta, \varphi \to \psi$ is valid.

Now let's consider the possible inferences with two premises.

1. The last inference is a cut: then $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi\end{array} \qquad \begin{array}{c}\vdots\\ \varphi, \Pi \Rightarrow \Lambda\end{array}}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\ \text{Cut}$$

Let $\mathfrak{v}$ be a valuation. By induction hypothesis, the premises are valid, so $\mathfrak{v}$ satisfies both premises. We distinguish two cases: (a) $\mathfrak{v} \nvDash \varphi$ and (b) $\mathfrak{v} \vDash \varphi$. In case (a), in order for $\mathfrak{v}$ to satisfy the left premise, it must satisfy $\Gamma \Rightarrow \Delta$. But then it also satisfies the conclusion. In case (b), in order for $\mathfrak{v}$ to satisfy the right premise, it must satisfy $\Pi \setminus \Lambda$. Again, $\mathfrak{v}$ satisfies the conclusion.

2. The last inference is $\wedge$R. Then $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi\end{array} \qquad \begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \psi\end{array}}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi} \wedge\text{R}$$

Consider a valuation $\mathfrak{v}$. If $\mathfrak{v}$ satisfies $\Gamma \Rightarrow \Delta$, we are done. So suppose it doesn't. Since $\Gamma \Rightarrow \Delta, \varphi$ is valid by induction hypothesis, $\mathfrak{v} \vDash \varphi$. Similarly, since $\Gamma \Rightarrow \Delta, \psi$ is valid, $\mathfrak{v} \vDash \psi$. But then $\mathfrak{v} \vDash \varphi \wedge \psi$.

3. The last inference is $\vee$L: Exercise.

4. The last inference is $\to$L. Then $\pi$ ends in

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \Rightarrow \Delta, \varphi & \psi, \Pi \Rightarrow \Lambda\end{array}}{\varphi \to \psi, \Gamma, \Pi \Rightarrow \Delta, \Lambda} \to\text{L}$$

Again, consider a valuation $\mathfrak{v}$ and suppose $\mathfrak{v}$ doesn't satisfy $\Gamma, \Pi \Rightarrow \Lambda, \Pi$. We have to show that $\mathfrak{v} \nvDash \varphi \to \psi$. If $\mathfrak{v}$ doesn't satisfy $\Gamma, \Pi \Rightarrow \Lambda, \Pi$, it satisfies neither $\Gamma \Rightarrow \Delta$ nor $\Pi \Rightarrow \Lambda$. Since, $\Gamma \Rightarrow \Delta, \varphi$ is valid, we have $\mathfrak{v} \vDash \varphi$. Since $\psi, \Pi \Rightarrow \Lambda$ is valid, we have $\mathfrak{v} \nvDash \psi$. But then $\mathfrak{v} \nvDash \varphi \to \psi$, which is what we wanted to show.

□

**Corollary 7.26.** *If $\vdash \varphi$ then $\varphi$ is a tautology.*

**Corollary 7.27.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* If $\Gamma \vdash \varphi$ then for some finite subset $\Gamma_0 \subseteq \Gamma$, there is a derivation of $\Gamma_0 \Rightarrow \varphi$. By Theorem 7.25, every valuation $\mathfrak{v}$ either makes some $\psi \in \Gamma_0$ false or makes $\varphi$ true. Hence, if $\mathfrak{v} \vDash \Gamma$ then also $\mathfrak{v} \vDash \varphi$. □

**Corollary 7.28.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then there is a finite $\Gamma_0 \subseteq \Gamma$ and a derivation of $\Gamma_0 \Rightarrow$ . By Theorem 7.25, $\Gamma_0 \Rightarrow$ is valid. In other words, for every valuation $\mathfrak{v}$, there is $\chi \in \Gamma_0$ so that $\mathfrak{v} \nvDash \chi$, and since $\Gamma_0 \subseteq \Gamma$, that $\chi$ is also in $\Gamma$. Thus, no $\mathfrak{v}$ satisfies $\Gamma$, and $\Gamma$ is not satisfiable. □

## Problems

**Problem 7.1.** Give derivations of the following sequents:

1. $\Rightarrow \neg(\varphi \to \psi) \to (\varphi \wedge \neg\psi)$
2. $(\varphi \wedge \psi) \to \chi \Rightarrow (\varphi \to \chi) \vee (\psi \to \chi)$

**Problem 7.2.** Prove Proposition 15.16

**Problem 7.3.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 7.4.** Complete the proof of Theorem 7.25.

# Chapter 8

# Natural Deduction

This chapter presents a natural deduction system in the style of Gentzen/Prawitz.

To include or exclude material relevant to natural deduction as a proof system, use the "prfND" tag.

## 8.1 Rules and Derivations

Natural deduction systems are meant to closely parallel the informal reasoning used in mathematical proof (hence it is somewhat "natural"). Natural deduction proofs begin with assumptions. Inference rules are then applied. Assumptions are "discharged" by the $\neg$Intro, $\rightarrow$Intro, and $\vee$Elim inference rules, and the label of the discharged assumption is placed beside the inference for clarity.

**Definition 8.1** (Initial Formula)**.** An *initial formula* or *assumption* is any formula in the topmost position of any branch.

Derivations in natural deduction are certain trees of sentences, where the topmost sentences are assumptions, and if a sentence stands below one, two, or three other sequents, it must follow correctly by a rule of inference. The sentences at the top of the inference are called the *premises* and the sentence below the *conclusion* of the inference. The rules come in pairs, an introduction and an elimination rule for each logical operator. They introduce a logical operator in the conclusion or remove a logical operator from a premise of the rule. Some of the rules allow an assumption of a certain type to be *discharged*. To indicate which assumption is discharged by which inference, we also assign labels to both the assumption and the inference. This is indicated by writing the assumption as "$[\varphi]^n$".

It is customary to consider rules for all logical operators, even for those (if any) that we consider as defined.

## 8.2 Propositional Rules

**Rules for $\wedge$**

$$\frac{\varphi \qquad \psi}{\varphi \wedge \psi}\ \wedge\text{Intro} \qquad \frac{\varphi \wedge \psi}{\varphi}\ \wedge\text{Elim} \qquad \frac{\varphi \wedge \psi}{\psi}\ \wedge\text{Elim}$$

**Rules for $\vee$**

$$\frac{\varphi}{\varphi \vee \psi}\ \vee\text{Intro} \qquad \frac{\psi}{\varphi \vee \psi}\ \vee\text{Intro} \qquad {}^{n}\frac{\varphi \vee \psi \qquad \begin{array}{c}[\varphi]^n \\ \vdots \\ \chi\end{array} \qquad \begin{array}{c}[\psi]^n \\ \vdots \\ \chi\end{array}}{\chi}\ \vee\text{Elim}$$

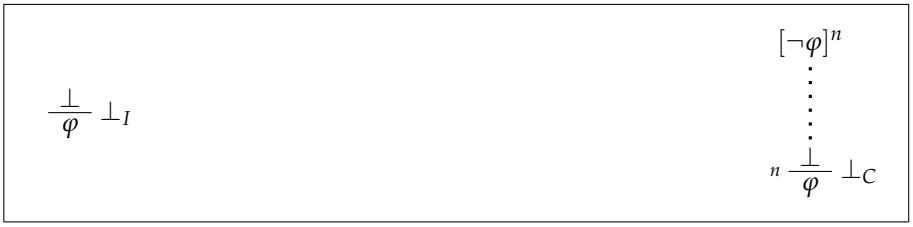
**Rules for $\rightarrow$**

$${}^{n}\frac{\begin{array}{c}[\varphi]^n \\ \vdots \\ \psi\end{array}}{\varphi \rightarrow \psi}\ \rightarrow\text{Intro} \qquad \frac{\varphi \rightarrow \psi \qquad \varphi}{\psi}\ \rightarrow\text{Elim}$$

**Rules for $\neg$**

$${}^{n}\frac{\begin{array}{c}[\varphi]^n \\ \vdots \\ \bot\end{array}}{\neg\varphi}\ \neg\text{Intro} \qquad \frac{\neg\varphi \qquad \varphi}{\bot}\ \neg\text{Elim}$$

### Rules for $\bot$




Note that ¬Intro and $\bot_C$ are very similar: The difference is that ¬Intro derives a negated sentence $\neg\varphi$ but $\bot_C$ a positive sentence $\varphi$.

## 8.3 Derivations

We've said what an assumption is, and we've given the rules of inference. Derivations in natural deduction are inductively generated from these: each derivation either is an assumption on its own, or consists of one, two, or three derivations followed by a correct inference.

**Definition 8.2** (Derivation)**.** A *derivation* of a sentence $\varphi$ from assumptions $\Gamma$ is a tree of sentences satisfying the following conditions:

1. The topmost sentences of the tree are either in $\Gamma$ or are discharged by an inference in the tree.
2. The bottommost sentence of the tree is $\varphi$.
3. Every sentence in the tree except $\varphi$ is a premise of a correct application of an inference rule whose conclusion stands directly below that sentence in the tree.

We then say that $\varphi$ is the *conclusion* of the derivation and that $\varphi$ is *derivable* from $\Gamma$.

**Example 8.3.** Every assumption on its own is a derivation. So, e.g., $\chi$ by itself is a derivation, and so is $\theta$ by itself. We can obtain a new derivation from these by applying, say, the ∧Intro rule,

$$\frac{\varphi \qquad \psi}{\varphi \wedge \psi}\,\wedge\text{Intro}$$

These rules are meant to be general: we can replace the $\varphi$ and $\psi$ in it with any sentences, e.g., by $\chi$ and $\theta$. Then the conclusion would be $\chi \wedge \theta$, and so

$$\frac{\chi \qquad \theta}{\chi \wedge \theta}\,\wedge\text{Intro}$$

is a correct derivation. Of course, we can also switch the assumptions, so that $\theta$ plays the role of $\varphi$ and $\chi$ that of $\psi$. Thus,

$$\dfrac{\theta \qquad \chi}{\theta \wedge \chi}\,\wedge\text{Intro}$$

is also a correct derivation.

We can now apply another rule, say, $\rightarrow$Intro, which allows us to conclude a conditional and allows us to discharge any assumption that is identical to the conclusion of that conditional. So both of the following would be correct derivations:

$$1\,\dfrac{\dfrac{[\chi]^1 \qquad \theta}{\chi \wedge \theta}\,\wedge\text{Intro}}{\chi \rightarrow (\chi \wedge \theta)}\,\rightarrow\text{Intro} \qquad 1\,\dfrac{\dfrac{\chi \qquad [\theta]^1}{\chi \wedge \theta}\,\wedge\text{Intro}}{\theta \rightarrow (\chi \wedge \theta)}\,\rightarrow\text{Intro}$$

## 8.4 Examples of Derivations

**Example 8.4.** Let's give a derivation of the sentence $(\varphi \wedge \psi) \rightarrow \varphi$.

We begin by writing the desired conclusion at the bottom of the derivation.

$$\dfrac{}{(\varphi \wedge \psi) \rightarrow \varphi}$$

Next, we need to figure out what kind of inference could result in a sentence of this form. The main operator of the conclusion is $\rightarrow$, so we'll try to arrive at the conclusion using the $\rightarrow$Intro rule. It is best to write down the assumptions involved and label the inference rules as you progress, so it is easy to see whether all assumptions have been discharged at the end of the proof.

$$1\,\dfrac{\begin{array}{c}[\varphi \wedge \psi]^1 \\ \vdots \\ \varphi\end{array}}{(\varphi \wedge \psi) \rightarrow \varphi}\,\rightarrow\text{Intro}$$

We now need to fill in the steps from the assumption $\varphi \wedge \psi$ to $\varphi$. Since we only have one connective to deal with, $\wedge$, we must use the $\wedge$ elim rule. This gives us the following proof:

$$1\,\dfrac{\dfrac{[\varphi \wedge \psi]^1}{\varphi}\,\wedge\text{Elim}}{(\varphi \wedge \psi) \rightarrow \varphi}\,\rightarrow\text{Intro}$$

We now have a correct derivation of $(\varphi \wedge \psi) \rightarrow \varphi$.

**Example 8.5.** Now let's give a derivation of $(\neg\varphi \vee \psi) \to (\varphi \to \psi)$.

We begin by writing the desired conclusion at the bottom of the derivation.

$$\frac{}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}$$

To find a logical rule that could give us this conclusion, we look at the logical connectives in the conclusion: $\neg$, $\vee$, and $\to$. We only care at the moment about the first occurence of $\to$ because it is the main operator of the sentence in the end-sequent, while $\neg$, $\vee$ and the second occurence of $\to$ are inside the scope of another connective, so we will take care of those later. We therefore start with the $\to$Intro rule. A correct application must look like this:

$$\dfrac{\begin{array}{c}[\neg\varphi \vee \psi]^1 \\ \vdots \\ \varphi \to \psi\end{array}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,{\scriptstyle 1}\ \to\text{Intro}$$

This leaves us with two possibilities to continue. Either we can keep working from the bottom up and look for another application of the $\to$Intro rule, or we can work from the top down and apply a $\vee$Elim rule. Let us apply the latter. We will use the assumption $\neg\varphi \vee \psi$ as the leftmost premise of $\vee$Elim. For a valid application of $\vee$Elim, the other two premises must be identical to the conclusion $\varphi \to \psi$, but each may be derived in turn from another assumption, namely the two disjuncts of $\neg\varphi \vee \psi$. So our derivation will look like this:

$$\dfrac{\dfrac{\begin{array}{ccc}[\neg\varphi \vee \psi]^1 & \begin{array}{c}[\neg\varphi]^2 \\ \vdots \\ \varphi \to \psi\end{array} & \begin{array}{c}[\psi]^2 \\ \vdots \\ \varphi \to \psi\end{array}\end{array}}{\varphi \to \psi}\,{\scriptstyle 2}\ \vee\text{Elim}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,{\scriptstyle 1}\ \to\text{Intro}$$

In each of the two branches on the right, we want to derive $\varphi \to \psi$, which is best done using $\to$Intro.

$$\dfrac{\dfrac{\begin{array}{ccc}[\neg\varphi \vee \psi]^1 & \dfrac{\begin{array}{c}[\neg\varphi]^2, [\varphi]^3 \\ \vdots \\ \psi\end{array}}{\varphi \to \psi}\,{\scriptstyle 3}\ \to\text{Intro} & \dfrac{\begin{array}{c}[\psi]^2, [\varphi]^4 \\ \vdots \\ \psi\end{array}}{\varphi \to \psi}\,{\scriptstyle 4}\ \to\text{Intro}\end{array}}{\varphi \to \psi}\,{\scriptstyle 2}\ \vee\text{Elim}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,{\scriptstyle 1}\ \to\text{Intro}$$

For the two missing parts of the derivation, we need derivations of $\psi$ from $\neg\varphi$ and $\varphi$ in the middle, and from $\varphi$ and $\psi$ on the left. Let's take the former first. $\neg\varphi$ and $\varphi$ are the two premises of $\neg$Elim:

$$\dfrac{\dfrac{[\neg\varphi]^2 \qquad [\varphi]^3}{\bot}\,\neg\text{Elim}}{\begin{array}{c}\vdots\\ \psi\end{array}}$$

By using $\bot_I$, we can obtain $\psi$ as a conclusion and complete the branch.

$$\dfrac{\dfrac{\dfrac{[\neg\varphi]^2 \qquad [\varphi]^3}{\bot}\,\bot\text{Intro}}{\dfrac{\bot}{\psi}\,\bot_I}}{\dfrac{[\neg\varphi \vee \psi]^1 \qquad \overset{3}{\dfrac{\psi}{\varphi \to \psi}}\,\to\text{Intro} \qquad \overset{4}{\dfrac{\begin{array}{c}[\psi]^2, [\varphi]^4\\ \vdots\\ \psi\end{array}}{\varphi \to \psi}}\,\to\text{Intro}}{\varphi \to \psi}\,\vee\text{Elim}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,\to\text{Intro}$$

Let's now look at the rightmost branch. Here it's important to realize that the definition of derivation *allows assumptions to be discharged* but *does not require* them to be. In other words, if we can derive $\psi$ from one of the assumptions $\varphi$ and $\psi$ without using the other, that's ok. And to derive $\psi$ from $\psi$ is trivial: $\psi$ by itself is such a derivation, and no inferences are needed. So we can simply delete the assumtion $\varphi$.

$$\dfrac{\dfrac{\dfrac{[\neg\varphi]^2 \qquad [\varphi]^3}{\bot}\,\neg\text{Elim}}{\dfrac{\bot}{\psi}\,\bot_I}}{\dfrac{[\neg\varphi \vee \psi]^1 \qquad \overset{3}{\dfrac{\psi}{\varphi \to \psi}}\,\to\text{Intro} \qquad \dfrac{[\psi]^2}{\varphi \to \psi}\,\to\text{Intro}}{\varphi \to \psi}\,\vee\text{Elim}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,\to\text{Intro}$$

Note that in the finished derivation, the rightmost $\to$Intro inference does not actually discharge any assumptions.

**Example 8.6.** So far we have not needed the $\bot_C$ rule. It is special in that it allows us to discharge an assumption that isn't a sub-formula of the conclusion of the rule. It is closely related to the $\bot_I$ rule. In fact, the $\bot_I$ rule is a special case of the $\bot_C$ rule—there is a logic called "intuitionistic logic" in which only $\bot_I$ is allowed. The $\bot_C$ rule is a last resort when nothing else works. For instance, suppose we want to derive $\varphi \vee \neg\varphi$. Our usual strategy would be to attempt to derive $\varphi \vee \neg\varphi$ using $\vee$Intro. But this would require us to derive either $\varphi$ or $\neg\varphi$ from no assumptions, and this can't be done. $\bot_C$ to the rescue!

$$\dfrac{\dfrac{}{\bot}}{\varphi \vee \neg\varphi}\,{\scriptstyle 1}\,\bot_C \qquad [\neg(\varphi \vee \neg\varphi)]^1$$

Now we're looking for a derivation of $\bot$ from $\neg(\varphi \vee \neg\varphi)$. Since $\bot$ is the conclusion of $\neg$Elim we might try that:

$$\dfrac{\dfrac{}{\bot}}{\varphi \vee \neg\varphi}\,{\scriptstyle 1}\,\bot_C \qquad \dfrac{}{\neg\varphi} \qquad \dfrac{}{\varphi} \qquad \neg\text{Elim}$$

Our strategy for finding a derivation of $\neg\varphi$ calls for an application of $\neg$Intro:

$$\dfrac{\dfrac{}{\bot}}{\varphi \vee \neg\varphi}\,{\scriptstyle 1}\,\bot_C \qquad \dfrac{\dfrac{}{\bot}}{\neg\varphi}\,{\scriptstyle 2}\,\neg\text{Intro} \qquad \dfrac{}{\varphi} \qquad \neg\text{Elim}$$

Here, we can get $\bot$ easily by applying $\neg$Elim to the assumption $\neg(\varphi \vee \neg\varphi)$ and $\varphi \vee \neg\varphi$ which follows from our new assumption $\varphi$ by $\vee$Intro:

$$\dfrac{\dfrac{}{\bot}}{\varphi \vee \neg\varphi}\,{\scriptstyle 1}\,\bot_C \qquad \dfrac{\dfrac{}{\bot}}{\neg\varphi}\,{\scriptstyle 2}\,\neg\text{Intro} \qquad \dfrac{[\varphi]^2}{\varphi \vee \neg\varphi}\,\vee\text{Intro} \qquad \dfrac{}{\neg\varphi}\,\neg\text{Elim} \qquad \dfrac{}{\varphi} \qquad \neg\text{Elim}$$

On the right side we use the same strategy, except we get $\varphi$ by $\bot_C$:

$$\dfrac{\dfrac{}{\bot}}{\varphi \vee \neg\varphi}\,{\scriptstyle 1}\,\bot_C \qquad \dfrac{\dfrac{}{\bot}}{\neg\varphi}\,{\scriptstyle 2}\,\neg\text{Intro} \qquad \dfrac{[\varphi]^2}{\varphi \vee \neg\varphi}\,\vee\text{Intro} \qquad \dfrac{}{\neg\varphi}\,\neg\text{Elim} \qquad \dfrac{\dfrac{}{\bot}}{\varphi}\,{\scriptstyle 3}\,\bot_C \qquad \dfrac{[\neg\varphi]^3}{\varphi \vee \neg\varphi}\,\vee\text{Intro} \qquad \dfrac{}{\neg\varphi}\,\neg\text{Elim} \qquad \neg\text{Elim}$$

## 8.5 Proof-Theoretic Notions

This section collects the definitions the provability relation and consistency for natural deduction.

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the derivability or non-derivability of certain sentences from others. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorems*.

**Definition 8.7** (Theorems)**.** A sentence $\varphi$ is a *theorem* if there is a derivation of $\varphi$ in natural deduction in which all assumptions are discharged. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 8.8** (Derivability)**.** A sentence $\varphi$ is *derivable from* a set of sentences $\Gamma$, $\Gamma \vdash \varphi$, if there is a derivation with conclusion $\varphi$ and in which every assumption is either discharged or is in $\Gamma$. If $\varphi$ is not derivable from $\Gamma$ we write $\Gamma \nvdash \varphi$.

**Definition 8.9** (Consistency)**.** A set of sentences $\Gamma$ is *inconsistent* iff $\Gamma \vdash \bot$. If $\Gamma$ is not inconsistent, i.e., if $\Gamma \nvdash \bot$, we say it is *consistent*.

**Proposition 8.10** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* The assumption $\varphi$ by itself is a derivation of $\varphi$ where every undischarged assumption (i.e., $\varphi$) is in $\Gamma$. □

**Proposition 8.11** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Any derivation of $\varphi$ from $\Gamma$ is also a derivation of $\varphi$ from $\Delta$. □

**Proposition 8.12** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* If $\Gamma \vdash \varphi$, there is a derivation $\delta_0$ of $\varphi$ with all undischarged assumptions in $\Gamma$. If $\{\varphi\} \cup \Delta \vdash \psi$, then there is a derivation $\delta_1$ of $\psi$ with all undischarged assumptions in $\{\varphi\} \cup \Delta$. Now consider:

$$\dfrac{\dfrac{\begin{array}{c}\Delta, [\varphi]^1 \\ \vdots\,\delta_1 \\ \psi\end{array}}{\varphi \to \psi}\,{\scriptstyle 1}\,\to\text{Intro} \qquad \begin{array}{c}\Gamma \\ \vdots\,\delta_0 \\ \varphi\end{array}}{\psi}\,\to\text{Elim}$$

The undischarged assumptions are now all among $\Gamma \cup \Delta$, so this shows $\Gamma \cup \Delta \vdash \psi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 8.13.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every sentence $\varphi$.*

*Proof.* Exercise. □

**Proposition 8.14** (Compactness)**.** 1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*

2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a derivation $\delta$ of $\varphi$ from $\Gamma$. Let $\Gamma_0$ be the set of undischarged assumptions of $\delta$. Since any derivation is finite, $\Gamma_0$ can only contain finitely many sentences. So, $\delta$ is a derivation of $\varphi$ from a finite $\Gamma_0 \subseteq \Gamma$.

2. This is the contrapositive of (1) for the special case $\varphi \equiv \bot$.

□

## 8.6 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 8.15.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* Let the derivation of $\varphi$ from $\Gamma$ be $\delta_1$ and the derivation of $\bot$ from $\Gamma \cup \{\varphi\}$ be $\delta_2$. We can then derive:

$$\dfrac{\dfrac{\begin{array}{c}\Gamma, [\varphi]^1 \\ \vdots\,\delta_2 \\ \bot\end{array}}{\neg\varphi}\,{}^1\neg\text{Intro} \qquad \begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ \varphi\end{array}}{\bot}\,\neg\text{Elim}$$

In the new derivation, the assumption $\varphi$ is discharged, so it is a derivation from $\Gamma$. □

**Proposition 8.16.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$, i.e., there is a derivation $\delta_0$ of $\varphi$ from undischarged assumptions $\Gamma$. We obtain a derivation of $\bot$ from $\Gamma \cup \{\neg\varphi\}$ as follows:

$$\dfrac{\neg\varphi \qquad \begin{array}{c}\Gamma\\ \vdots\\ \delta_0\\ \vdots\\ \varphi\end{array}}{\bot}\ \neg\text{Elim}$$

Now assume $\Gamma \cup \{\neg\varphi\}$ is inconsistent, and let $\delta_1$ be the corresponding derivation of $\bot$ from undischarged assumptions in $\Gamma \cup \{\neg\varphi\}$. We obtain a derivation of $\varphi$ from $\Gamma$ alone by using $\bot_C$:

$$\dfrac{\begin{array}{c}\Gamma, [\neg\varphi]^1\\ \vdots\\ \delta_1\\ \vdots\\ \bot\end{array}}{\varphi}\ \bot_C$$

□

**Proposition 8.17.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* Suppose $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$. Then there is a derivation $\delta$ of $\varphi$ from $\Gamma$. Consider this simple application of the ¬Elim rule:

$$\dfrac{\neg\varphi \qquad \begin{array}{c}\Gamma\\ \vdots\\ \delta\\ \vdots\\ \varphi\end{array}}{\bot}\ \neg\text{Elim}$$

Since $\neg\varphi \in \Gamma$, all undischarged assumptions are in $\Gamma$, this shows that $\Gamma \vdash \bot$. □

**Proposition 8.18.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are derivations $\delta_1$ and $\delta_2$ of $\bot$ from $\Gamma \cup \{\varphi\}$ and $\bot$ from $\Gamma \cup \{\neg\varphi\}$, respectively. We can then derive

$$\dfrac{\dfrac{\begin{array}{c}\Gamma, [\neg\varphi]^2\\ \vdots\\ \delta_2\\ \vdots\\ \bot\end{array}}{\neg\neg\varphi}\ 2\ \neg\text{Intro} \qquad \dfrac{\begin{array}{c}\Gamma, [\varphi]^1\\ \vdots\\ \delta_1\\ \vdots\\ \bot\end{array}}{\neg\varphi}\ 1\ \neg\text{Intro}}{\bot}\ \neg\text{Elim}$$

Since the assumptions $\varphi$ and $\neg\varphi$ are discharged, this is a derivation of $\bot$ from $\Gamma$ alone. Hence $\Gamma$ is inconsistent. □

## 8.7 Derivability and the Propositional Connectives

**Proposition 8.19.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. We can derive both

$$\dfrac{\varphi \wedge \psi}{\varphi}\,\wedge\text{Elim} \qquad \dfrac{\varphi \wedge \psi}{\psi}\,\wedge\text{Elim}$$

2. We can derive:

$$\dfrac{\varphi \qquad \psi}{\varphi \wedge \psi}\,\wedge\text{Intro}$$

□

**Proposition 8.20.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. Consider the following derivation:

$$\dfrac{\varphi \vee \psi \qquad \dfrac{\neg\varphi \qquad [\varphi]^1}{\bot}\,\neg\text{Elim} \qquad \dfrac{\neg\psi \qquad [\psi]^1}{\bot}\,\neg\text{Elim}}{\bot}\,\vee\text{Elim}\ 1$$

This is a derivation of $\bot$ from undischarged assumptions $\varphi \vee \psi$, $\neg\varphi$, and $\neg\psi$.

2. We can derive both

$$\dfrac{\varphi}{\varphi \vee \psi}\,\vee\text{Intro} \qquad \dfrac{\psi}{\varphi \vee \psi}\,\vee\text{Intro}$$

□

**Proposition 8.21.** *1. $\varphi, \varphi \to \psi \vdash \psi$.*

*2. Both $\neg\varphi \vdash \varphi \to \psi$ and $\psi \vdash \varphi \to \psi$.*

*Proof.* 1. We can derive:

$$\dfrac{\varphi \to \psi \qquad \psi}{\psi}\,\to\text{Elim}$$

2. This is shown by the following two derivations:

$$\dfrac{\dfrac{\dfrac{\neg\varphi \qquad [\varphi]^1}{\bot}\,\neg\text{Elim}}{\psi}\,\bot_I}{1\ \dfrac{}{\varphi\to\psi}\,\to\text{Intro}} \qquad\qquad \dfrac{\psi}{\varphi\to\psi}\,\to\text{Intro}$$

Note that $\to$Intro may, but does not have to, discharge the assumption $\varphi$. □

## 8.8 Soundness

A derivation system, such as natural deduction, is *sound* if it cannot derive things that do not actually follow. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable sentence is a tautology;
2. if a sentence is derivable from some others, it is also a consequence of them;
3. if a set of sentences is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

**Theorem 8.22** (Soundness)**.** *If $\varphi$ is derivable from the undischarged assumptions $\Gamma$, then $\Gamma \vDash \varphi$.*

*Proof.* Let $\delta$ be a derivation of $\varphi$. We proceed by induction on the number of inferences in $\delta$.

For the induction basis we show the claim if the number of inferences is 0. In this case, $\delta$ consists only of an initial formula. Every initial formula $\varphi$ is an undischarged assumption, and as such, any valuation $\mathfrak{v}$ that satisfies all of the undischarged assumptions of the proof also satisfies $\varphi$.

Now for the inductive step. Suppose that $\delta$ contains $n$ inferences. The premise(s) of the lowermost inference are derived using sub-derivations, each of which contains fewer than $n$ inferences. We assume the induction hypothesis: The premises of the last inference follow from the undischarged assumptions of the sub-derivations ending in those premises. We have to show that $\varphi$ follows from the undischarged assumptions of the entire proof.

We distinguish cases according to the type of the lowermost inference. First, we consider the possible inferences with only one premise.

1. Suppose that the last inference is $\neg$Intro: The derivation has the form

$$\dfrac{\begin{array}{c}\Gamma, [\varphi]^n \\ \vdots \\ \delta_1 \\ \vdots \\ \bot\end{array}}{\neg\varphi}\, {}^{n}\neg\text{Intro}$$

By inductive hypothesis, $\bot$ follows from the undischarged assumptions $\Gamma \cup \{\varphi\}$ of $\delta_1$. Consider a valuation $\mathfrak{v}$. We need to show that, if $\mathfrak{v} \vDash \Gamma$, then $\mathfrak{v} \vDash \neg\varphi$. Suppose for reductio that $\mathfrak{v} \vDash \Gamma$, but $\mathfrak{v} \nvDash \neg\varphi$, i.e., $\mathfrak{v} \vDash \varphi$. This would mean that $\mathfrak{v} \vDash \Gamma \cup \{\varphi\}$. This is contrary to our inductive hypothesis. So, $\mathfrak{v} \vDash \neg\varphi$.

2. The last inference is $\wedge$Elim: There are two variants: $\varphi$ or $\psi$ may be inferred from the premise $\varphi \wedge \psi$. Consider the first case. The derivation $\delta$ looks like this:

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \delta_1 \\ \vdots \\ \varphi \wedge \psi\end{array}}{\varphi}\,\wedge\text{Elim}$$

By inductive hypothesis, $\varphi \wedge \psi$ follows from the undischarged assumptions $\Gamma$ of $\delta_1$. Consider a structure $\mathfrak{v}$. We need to show that, if $\mathfrak{v} \vDash \Gamma$, then $\mathfrak{v} \vDash \varphi$. Suppose $\mathfrak{v} \vDash \Gamma$. By our inductive hypothesis ($\Gamma \vDash \varphi \vee \psi$), we know that $\mathfrak{v} \vDash \varphi \wedge \psi$. By definition, $\mathfrak{v} \vDash \varphi \wedge \psi$ iff $\mathfrak{v} \vDash \varphi$ and $\mathfrak{v} \vDash \psi$. (The case where $\psi$ is inferred from $\varphi \wedge \psi$ is handled similarly.)

3. The last inference is $\vee$Intro: There are two variants: $\varphi \vee \psi$ may be inferred from the premise $\varphi$ or the premise $\psi$. Consider the first case. The derivation has the form

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \delta_1 \\ \vdots \\ \varphi\end{array}}{\varphi \vee \psi}\,\vee\text{Intro}$$

By inductive hypothesis, $\varphi$ follows from the undischarged assumptions $\Gamma$ of $\delta_1$. Consider a valuation $\mathfrak{v}$. We need to show that, if $\mathfrak{v} \vDash \Gamma$, then $\mathfrak{v} \vDash \varphi \vee \psi$. Suppose $\mathfrak{v} \vDash \Gamma$; then $\mathfrak{v} \vDash \varphi$ since $\Gamma \vDash \varphi$ (the inductive hypothesis). So it must also be the case that $\mathfrak{v} \vDash \varphi \vee \psi$. (The case where $\varphi \vee \psi$ is inferred from $\psi$ is handled similarly.)

4. The last inference is $\rightarrow$Intro: $\varphi \rightarrow \psi$ is inferred from a subproof with assumption $\varphi$ and conclusion $\psi$, i.e.,

$$\dfrac{\begin{array}{c}\Gamma, [\varphi]^n \\ \vdots\,\delta_1 \\ \psi\end{array}}{\varphi \to \psi}\, n \to\text{Intro}$$

By inductive hypothesis, $\psi$ follows from the undischarged assumptions of $\delta_1$, i.e., $\Gamma \cup \{\varphi\} \vDash \psi$. Consider a valuation $\mathfrak{v}$. The undischarged assumptions of $\delta$ are just $\Gamma$, since $\varphi$ is discharged at the last inference. So we need to show that $\Gamma \vDash \varphi \to \psi$. For reductio, suppose that for some valuation $\mathfrak{v}$, $\mathfrak{v} \vDash \Gamma$ but $\mathfrak{v} \nvDash \varphi \to \psi$. So, $\mathfrak{v} \vDash \varphi$ and $\mathfrak{v} \nvDash \psi$. But by hypothesis, $\psi$ is a consequence of $\Gamma \cup \{\varphi\}$, i.e., $\mathfrak{v} \vDash \psi$, which is a contradiction. So, $\Gamma \vDash \varphi \to \psi$.

5. The last inference is $\bot_I$: Here, $\delta$ ends in

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ \bot\end{array}}{\varphi}\, \bot_I$$

By induction hypothesis, $\Gamma \vDash \bot$. We have to show that $\Gamma \vDash \varphi$. Suppose not; then for some $\mathfrak{v}$ we have $\mathfrak{v} \vDash \Gamma$ and $\mathfrak{v} \nvDash \varphi$. But we always have $\mathfrak{v} \nvDash \bot$, so this would mean that $\Gamma \nvDash \bot$, contrary to the induction hypothesis.

6. The last inference is $\bot_C$: Exercise.

Now let's consider the possible inferences with several premises: $\vee$Elim, $\wedge$Intro, and $\to$Elim.

1. The last inference is $\wedge$Intro. $\varphi \wedge \psi$ is inferred from the premises $\varphi$ and $\psi$ and $\delta$ has the form

$$\dfrac{\begin{array}{cc}\Gamma_1 & \Gamma_2 \\ \vdots\,\delta_1 & \vdots\,\delta_2 \\ \varphi & \psi\end{array}}{\varphi \wedge \psi}\, \wedge\text{Intro}$$

By induction hypothesis, $\varphi$ follows from the undischarged assumptions $\Gamma_1$ of $\delta_1$ and $\psi$ follows from the undischarged assumptions $\Gamma_2$ of $\delta_2$. The undischarged assumptions of $\delta$ are $\Gamma_1 \cup \gamma_2$, so we have to show that $\Gamma_1 \cup \Gamma_2 \vDash \varphi \wedge \psi$. Consider a valuation $\mathfrak{v}$ with $\mathfrak{v} \vDash \Gamma_1 \cup \Gamma_2$. Since $\mathfrak{v} \vDash \Gamma_1$, it must be the case that $\mathfrak{v} \vDash \varphi$ as $\Gamma_1 \vDash \varphi$, and since $\mathfrak{v} \vDash \Gamma_2$, $\mathfrak{v} \vDash \psi$ since $\Gamma_2 \vDash \psi$. Together, $\mathfrak{v} \vDash \varphi \wedge \psi$.

2. The last inference is $\vee$Elim: Exercise.

3. The last inference is $\rightarrow$Elim. $\psi$ is inferred from the premises $\varphi \rightarrow \psi$ and $\varphi$. The derivation $\delta$ looks like this:

$$\dfrac{\begin{array}{cc}\Gamma_1 & \Gamma_2 \\ \vdots & \vdots \\ \delta_1 & \delta_2 \\ \vdots & \vdots \\ \varphi \rightarrow \psi & \varphi\end{array}}{\psi} \rightarrow\text{Elim}$$

   By induction hypothesis, $\varphi \rightarrow \psi$ follows from the undischarged assumptions $\Gamma_1$ of $\delta_1$ and $\varphi$ follows from the undischarged assumptions $\Gamma_2$ of $\delta_2$. Consider a valuation $\mathfrak{v}$. We need to show that, if $\mathfrak{v} \vDash \Gamma_1 \cup \Gamma_2$, then $\mathfrak{v} \vDash \psi$. Suppose $\mathfrak{v} \vDash \Gamma_1 \cup \Gamma_2$. Since $\Gamma_1 \vDash \varphi \rightarrow \psi$, $\mathfrak{v} \vDash \varphi \rightarrow \psi$. Since $\Gamma_2 \vDash \varphi$, we have $\mathfrak{v} \vDash \varphi$. This means that $\mathfrak{v} \vDash \psi$ (For if $\mathfrak{v} \nvDash \psi$, since $\mathfrak{v} \vDash \varphi$, we'd have $\mathfrak{v} \nvDash \varphi \rightarrow \psi$, contradicting $\mathfrak{v} \vDash \varphi \rightarrow \psi$).

4. The last inference is $\neg$Elim: Exercise.

□

**Corollary 8.23.** *If $\vdash \varphi$, then $\varphi$ is a tautology.*

**Corollary 8.24.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then $\Gamma \vdash \bot$, i.e., there is a derivation of $\bot$ from undischarged assumptions in $\Gamma$. By Theorem 8.22, any valuation $\mathfrak{v}$ that satisfies $\Gamma$ must satisfy $\bot$. Since $\mathfrak{v} \nvDash \bot$ for every valuation $\mathfrak{v}$, no $\mathfrak{v}$ can satisfy $\Gamma$, i.e., $\Gamma$ is not satisfiable. □

## Problems

**Problem 8.1.** Give derivations of the following:

1. $\neg(\varphi \rightarrow \psi) \rightarrow (\varphi \wedge \neg\psi)$
2. $(\varphi \rightarrow \chi) \vee (\psi \rightarrow \chi)$ from the assumption $(\varphi \wedge \psi) \rightarrow \chi$

**Problem 8.2.** Prove Proposition 8.13

**Problem 8.3.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 8.4.** Complete the proof of Theorem 8.22.

# Chapter 9

# Tableaux

This chapter presents a signed analytic tableaux system.
To include or exclude material relevant to natural deduction as a proof system, use the "prfTab" tag.

## 9.1 Rules and Tableaux

A tableau is a systematic survey of the possible ways a sentence can be true or false in a structure. The bulding blocks of a tableau are signed formulas: sentences plus a truth value "sign," either $\mathbb{T}$ or $\mathbb{F}$. These signed formulas are arranged in a (downward growing) tree.

**Definition 9.1.** A *signed formula* is a pair consisting of a truth value and a sentence, i.e., either:

$$\mathbb{T}\varphi \text{ or } \mathbb{F}\varphi.$$

Intuitively, we might read $\mathbb{T}\varphi$ as "$\varphi$ might be true" and $\mathbb{F}\varphi$ as "$\varphi$ might be false" (in some structure).

Each signed formula in the tree is either an *assumption* (which are listed at the very top of the tree), or it is obtained from a signed formula above it by one of a number of rules of inference. There are two rules for each possible main operator of the preceding formula, one for the case when the sign is $\mathbb{T}$, and one for the case where the sign is $\mathbb{F}$. Some rules allow the tree to branch, and some only add signed formulas to the branch. A rule may be (and often must be) applied not to the immediately preceding signed formula, but to any signed formula in the branch from the root to the place the rule is applied.

A branch is *closed* when it contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$. A closed tableau is one where every branch is closed. Under the intuitive interpretation, any branch describes a joint possibility, but $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$ are not jointly possible. In other words, if a branch is closed, the possibility it describes has been ruled

out. In particular, that means that a closed tableau rules out all possibilities of simultaneously making every assumption of the form $\mathbb{T}\varphi$ true and every assumption of the form $\mathbb{F}\varphi$ false.

A closed tableau *for* $\varphi$ is a closed tableau with root $\mathbb{F}\varphi$. If such a closed tableau exists, all possibilities for $\varphi$ being false have been ruled out; i.e., $\varphi$ must be true in every structure.

## 9.2 Propositional Rules

**Rules for $\neg$**

$$\frac{\mathbb{T}\neg\varphi}{\mathbb{F}\varphi}\ \neg\mathbb{T} \qquad\qquad \frac{\mathbb{F}\neg\varphi}{\mathbb{T}\varphi}\ \neg\mathbb{F}$$

**Rules for $\wedge$**

$$\frac{\mathbb{T}\varphi\wedge\psi}{\begin{array}{c}\mathbb{T}\varphi\\ \mathbb{T}\psi\end{array}}\ \wedge\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\wedge\psi}{\mathbb{F}\varphi \quad\mid\quad \mathbb{F}\psi}\ \wedge\mathbb{F}$$

**Rules for $\vee$**

$$\frac{\mathbb{T}\varphi\vee\psi}{\mathbb{T}\varphi \quad\mid\quad \mathbb{T}\psi}\ \vee\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\vee\psi}{\begin{array}{c}\mathbb{F}\varphi\\ \mathbb{F}\psi\end{array}}\ \vee\mathbb{F}$$

**Rules for $\rightarrow$**

$$\frac{\mathbb{T}\varphi\rightarrow\psi}{\mathbb{F}\varphi \quad\mid\quad \mathbb{T}\psi}\ \rightarrow\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\rightarrow\psi}{\begin{array}{c}\mathbb{T}\varphi\\ \mathbb{F}\psi\end{array}}\ \rightarrow\mathbb{F}$$

### The Cut Rule

$$\frac{}{\mathbb{T}\varphi \quad | \quad \mathbb{F}\varphi}\ \text{Cut}$$

The Cut rule is not applied "to" a previous signed formula; rather, it allows every branch in a tableau to be split in two, one branch containing $\mathbb{T}\varphi$, the other $\mathbb{F}\varphi$. It is not necessary—any set of signed formulas with a closed tableau has one not using Cut—but it allows us to combine tableaux in a convenient way.

## 9.3 Tableaux

We've said what an assumption is, and we've given the rules of inference. Tableaux are inductively generated from these: each tableau either is a single branch consisting of one or more assumptions, or it results from a tableau by applying one of the rules of inference on a branch.

**Definition 9.2** (Tableau)**.** A tableau for assumptions $S_1\varphi_1, \ldots, S_n\varphi_n$ (where each $S_i$ is either $\mathbb{T}$ or $\mathbb{F}$) is a tree of signed formulas satisfying the following conditions:

1. The $n$ topmost signed formulas of the tree are $S_i\varphi_i$, one below the other.
2. Every signed formula in the tree that is not one of the assumptions results from a correct application of an inference rule to a signed formula in the branch above it.

A branch of a tableau is *closed* iff it contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$, and *open* otherwise. A tableau in which every branch is closed is a *closed tableau* (for its set of assumptions). If a tableau is not closed, i.e., if it contains at least one open branch, it is *open*.

**Example 9.3.** Every set of assumptions on its own is a tableau, but it will generally not be closed. (Obviously, it is closed only if the assumptions already contain a pair of signed formulas $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$.)

From a tableau (open or closed) we can obtain a new, larger one by applying one of the rules of inference to a signed formula $\varphi$ in it. The rule will append one or more signed formulas to the end of any branch containing the occurrence of $\varphi$ to which we apply the rule.

For instance, consider the assumption $\mathbb{T}\varphi \wedge \neg\varphi$. Here is the (open) tableau consisting of just that assumption:

$$1. \quad \mathbb{T}\varphi \wedge \neg\varphi \qquad \text{Assumption}$$

We obtain a new tableau from it by applying the $\wedge\mathbb{T}$ rule to the assumption. That rule allows us to add two new lines to the tableau, $\mathbb{T}\varphi$ and $\mathbb{T}\neg\varphi$:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\varphi \wedge \neg\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,1$ |
| 3. | $\mathbb{T}\neg\varphi$ | $\wedge\mathbb{T}\,1$ |

When we write down tableaux, we record the rules we've applied on the right (e.g., $\wedge\mathbb{T}1$ means that the signed formula on that line is the result of applying the $\wedge\mathbb{T}$ rule to the signed formula on line 1). This new tableau now contains additional signed formulas, but to only one ($\mathbb{T}\neg\varphi$) can we apply a rule (in this case, the $\neg\mathbb{T}$ rule). This results in the closed tableau

| | | |
|---|---|---|
| 1. | $\mathbb{T}\varphi \wedge \neg\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,1$ |
| 3. | $\mathbb{T}\neg\varphi$ | $\wedge\mathbb{T}\,1$ |
| 4. | $\mathbb{F}\varphi$ | $\neg\mathbb{T}\,3$ |
| | $\otimes$ | |

## 9.4 Examples of Tableaux

**Example 9.4.** Let's find a closed tableau for the sentence $(\varphi \wedge \psi) \to \varphi$.

We begin by writing the corresponding assumption at the top of the tableau.

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ | Assumption |

There is only one assumption, so only one signed formula to which we can apply a rule. (For every signed formula, there is always at most one rule that can be applied: it's the rule for the corresponding sign and main operator of the sentence.) In this case, this means, we must apply $\to\mathbb{F}$.

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ ✓ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\varphi$ | $\to\mathbb{F}\,1$ |

To keep track of which signed formulas we have applied their corresponding rules to, we write a checkmark next to the sentence. However, *only* write a checkmark if the rule has been applied to all open branches. Once a signed formula has had the corresponding rule applied in every open branch, we will not have to return to it and apply the rule again. In this case, there is only one branch, so the rule only has to be applied once. (Note that checkmarks are only a convenience for constructing tableaux and are not officially part of the syntax of tableaux.)

There is one new signed formula to which we can apply a rule: the $\mathbb{T}\varphi \wedge \psi$ on line 3. Applying the $\wedge\mathbb{T}$ rule results in:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ ✓ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ ✓ | $\to\mathbb{F}$ 1 |
| 3. | $\mathbb{F}\varphi$ | $\to\mathbb{F}$ 1 |
| 4. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}$ 2 |
| 5. | $\mathbb{T}\psi$ | $\wedge\mathbb{T}$ 2 |
| | $\otimes$ | |

Since the branch now contains both $\mathbb{T}\varphi$ (on line 4) and $\mathbb{F}\varphi$ (on line 3), the branch is closed. Since it is the only branch, the tableau is closed. We have found a closed tableau for $(\varphi \wedge \psi) \to \varphi$.

**Example 9.5.** Now let's find a closed tableau for $(\neg\varphi \vee \psi) \to (\varphi \to \psi)$.

We begin with the corresponding assumption:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ | Assumption |

The one signed formula in this tableau has main operator $\to$ and sign $\mathbb{F}$, so we apply the $\to\mathbb{F}$ rule to it to obtain:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | Assumption |
| 2. | $\mathbb{T}\neg\varphi \vee \psi$ | $\to\mathbb{F}$ 1 |
| 3. | $\mathbb{F}(\varphi \to \psi)$ | $\to\mathbb{F}$ 1 |

We now have a choice as to whether to apply $\vee\mathbb{T}$ to line 2 or $\to\mathbb{F}$ to line 3. It actually doesn't matter which order we pick, as long as each signed formula has its corresponding rule applied in every branch. So let's pick the first one. The $\vee\mathbb{T}$ rule allows the tableau to branch, and the two conclusions of the rule will be the new signed formulas added to the two new branches. This results in:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | Assumption |
| 2. | $\mathbb{T}\neg\varphi \vee \psi$ ✓ | $\to\mathbb{F}$ 1 |
| 3. | $\mathbb{F}(\varphi \to \psi)$ | $\to\mathbb{F}$ 1 |
| 4. | $\mathbb{T}\neg\varphi$ $\quad$ $\mathbb{T}\psi$ | $\vee\mathbb{T}$ 2 |

We have not applied the $\to\mathbb{F}$ rule to line 3 yet: let's do that now. To save time, we apply it to both branches. Recall that we write a checkmark next to a signed formula only if we have applied the corresponding rule in every open branch. So it's a good idea to apply a rule at the end of every branch that contains the signed formula the rule applies to. That way we won't have to return to that signed formula lower down in the various branches.

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | Assumption |
| 2. | $\mathbb{T}\,\neg\varphi \vee \psi$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,(\varphi \to \psi)$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\,\neg\varphi$ \| $\mathbb{T}\,\psi$ | $\vee\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\,\varphi$ \| $\mathbb{T}\,\varphi$ | $\to\mathbb{F}\,3$ |
| 6. | $\mathbb{F}\,\psi$ \| $\mathbb{F}\,\psi$ | $\to\mathbb{F}\,3$ |
| | $\otimes$ | |

The right branch is now closed. On the left branch, we can still apply the $\neg\mathbb{T}$ rule to line 4. This results in $\mathbb{F}\,\varphi$ and closes the left branch:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | Assumption |
| 2. | $\mathbb{T}\,\neg\varphi \vee \psi$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,(\varphi \to \psi)$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\,\neg\varphi$ \| $\mathbb{T}\,\psi$ | $\vee\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\,\varphi$ \| $\mathbb{T}\,\varphi$ | $\to\mathbb{F}\,3$ |
| 6. | $\mathbb{F}\,\psi$ \| $\mathbb{F}\,\psi$ | $\to\mathbb{F}\,3$ |
| 7. | $\mathbb{F}\,\varphi$ \| $\otimes$ | $\neg\mathbb{T}\,4$ |
| | $\otimes$ | |

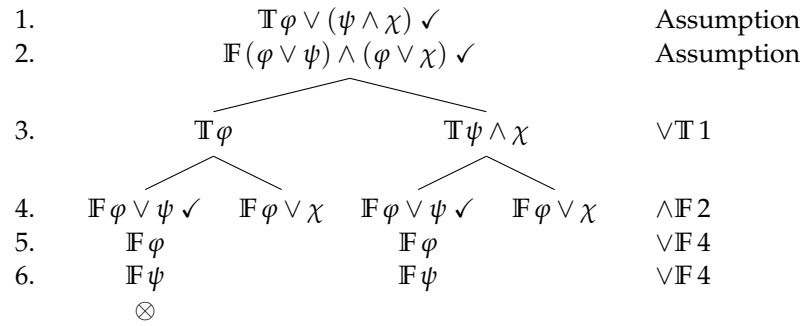
**Example 9.6.** We can give tableaux for any number of signed formulas as assumptions. Often it is also necessary to apply more than one rule that allows branching; and in general a tableau can have any number of branches. For instance, consider a tableau for $\{\mathbb{T}\,\varphi \vee (\psi \wedge \chi), \mathbb{F}\,(\varphi \vee \psi) \wedge (\varphi \vee \chi)\}$. We start by applying the $\vee\mathbb{T}$ to the first assumption:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\,\varphi \vee (\psi \wedge \chi)$ ✓ | Assumption |
| 2. | $\mathbb{F}\,(\varphi \vee \psi) \wedge (\varphi \vee \chi)$ | Assumption |
| 3. | $\mathbb{T}\,\varphi$ \| $\mathbb{T}\,\psi \wedge \chi$ | $\vee\mathbb{T}\,1$ |

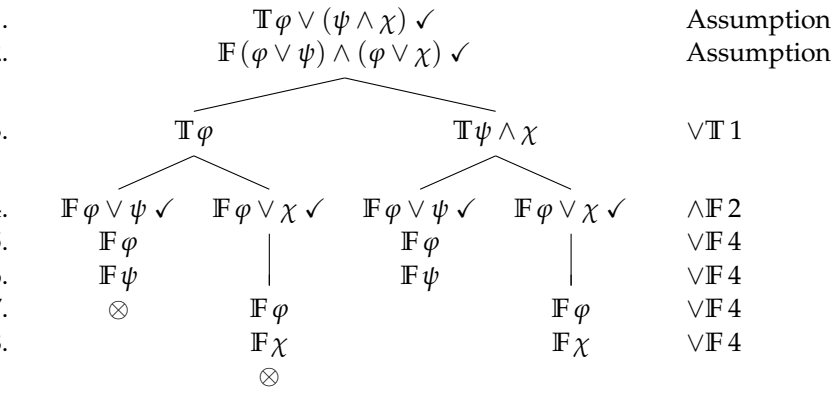
Now we can apply the $\wedge\mathbb{F}$ rule to line 2. We do this on both branches simultaneously, and can therefore check off line 2:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\,\varphi \vee (\psi \wedge \chi)$ ✓ | Assumption |
| 2. | $\mathbb{F}\,(\varphi \vee \psi) \wedge (\varphi \vee \chi)$ ✓ | Assumption |
| 3. | $\mathbb{T}\,\varphi$ \| $\mathbb{T}\,\psi \wedge \chi$ | $\vee\mathbb{T}\,1$ |
| 4. | $\mathbb{F}\,\varphi \vee \psi$ \| $\mathbb{F}\,\varphi \vee \chi$ \| $\mathbb{F}\,\varphi \vee \psi$ \| $\mathbb{F}\,\varphi \vee \chi$ | $\wedge\mathbb{F}\,2$ |

Now we can apply $\vee\mathbb{F}$ to all the branches containing $\varphi \vee \psi$:

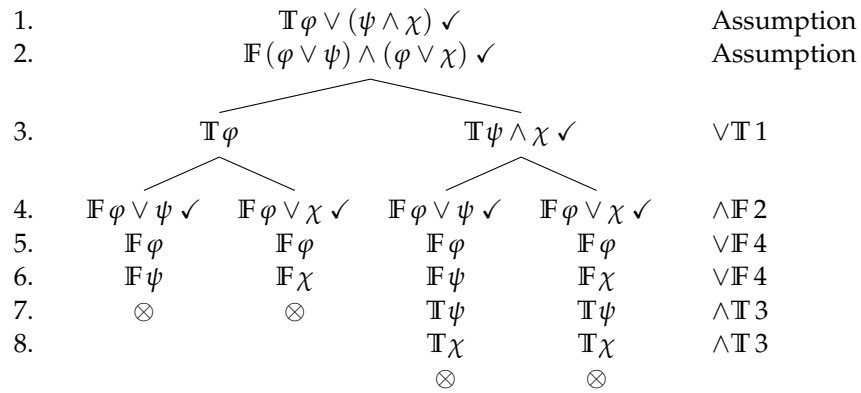



The leftmost branch is now closed. Let's now apply $\vee\mathbb{F}$ to $\varphi \vee \chi$:

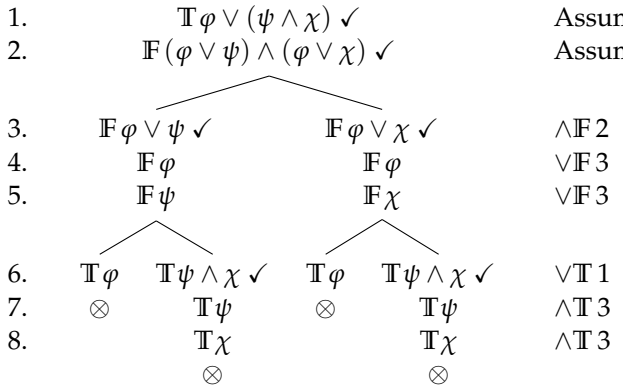



Note that we moved the result of applying $\vee\mathbb{F}$ a second time below for clarity. In this instance it would not have been needed, since the justifications would have been the same.

Two branches remain open, and $\mathbb{T}\psi \wedge \chi$ on line 3 remains unchecked. We apply $\wedge\mathbb{T}$ to it to obtain a closed tableau:




For comparison, here's a closed tableau for the same set of assumptions in which the rules are applied in a different order:




## 9.5 Proof-Theoretic Notions

This section collects the definitions of the provability relation and consistency for tableaux.

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the existence of certain closed tableaux. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorems*.

**Definition 9.7** (Theorems)**.** A sentence $\varphi$ is a *theorem* if there is a closed tableau for $\mathbb{F}\varphi$. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 9.8** (Derivability)**.** A sentence $\varphi$ is *derivable from* a set of sentences $\Gamma$, $\Gamma \vdash \varphi$, iff there is a finite set $\{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ and a closed tableau for the set

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n, \}$$

If $\varphi$ is not derivable from $\Gamma$ we write $\Gamma \nvdash \varphi$.

**Definition 9.9** (Consistency)**.** A set of sentences $\Gamma$ is *inconsistent* iff there is a finite set $\{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ and a closed tableau for the set

$$\{\mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n, \}.$$

If $\Gamma$ is not inconsistent, we say it is *consistent*.

**Proposition 9.10** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* If $\varphi \in \Gamma$, $\{\varphi\}$ is a finite subset of $\Gamma$ and the tableau

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | Assumption |
| | $\otimes$ | |

is closed. □

**Proposition 9.11** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Any finite subset of $\Gamma$ is also a finite subset of $\Delta$. □

**Proposition 9.12** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* If $\{\varphi\} \cup \Delta \vdash \psi$, then there is a finite subset $\Delta_0 = \{\chi_1, \ldots, \chi_n\} \subseteq \Delta$ such that

$$\{\mathbb{F}\psi, \mathbb{T}\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_n\}$$

has a closed tableau. If $\Gamma \vdash \varphi$ then there are $\theta_1, \ldots, \theta_m$ such that

$$\{\mathbb{F}\varphi, \mathbb{T}\theta_1, \ldots, \mathbb{T}\theta_m\}$$

has a closed tableau.

Now consider the tableau with assumptions

$$\mathbb{F}\psi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_n, \mathbb{T}\theta_1, \ldots, \mathbb{T}\theta_m.$$

Apply the Cut rule on $\varphi$. This generates two branches, one has $\mathbb{T}\varphi$ in it, the other $\mathbb{F}\varphi$. Thus, on the one branch, all of

$$\{\mathbb{F}\psi, \mathbb{T}\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_n\}$$

are available. Since there is a closed tableau for these assumptions, we can attach it to that branch; every branch through $\mathbb{T}\varphi_1$ closes. On the other branch, all of

$$\{\mathbb{F}\varphi, \mathbb{T}\theta_1, \ldots, \mathbb{T}\theta_m\}$$

are available, so we can also complete the other side to obtain a closed tableau. This shows $\Gamma \cup \Delta \vdash \psi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 9.13.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every sentence $\varphi$.*

*Proof.* Exercise. □

**Proposition 9.14** (Compactness)**.** 1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*

2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a finite subset $\Gamma_0 = \{\psi_1, \ldots, \psi_n\}$ and a closed tableau for

$$\mathbb{F}\varphi, \mathbb{T}\psi_1, \cdots \mathbb{T}\psi_n$$

This tableau also shows $\Gamma_0 \vdash \varphi$.

2. If $\Gamma$ is inconsistent, then for some finite subset $\Gamma_0 = \{\psi_1, \ldots, \psi_n\}$ there is a closed tableau for

$$\mathbb{T}\psi_1, \cdots \mathbb{T}\psi_n$$

This closed tableau shows that $\Gamma_0$ is inconsistent.

□

## 9.6 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 9.15.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are finite $\Gamma_0 = \{\psi_1, \ldots, \psi_n\}$ and $\Gamma_1 = \{\chi_1, \ldots, \chi_n\} \subseteq \Gamma$ such that

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$
$$\{\mathbb{T}\neg\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_m\}$$

have closed tableaux. Using the Cut rule on $\varphi$ we can combine these into a single closed tableau that shows $\Gamma_0 \cup \Gamma_1$ is inconsistent. Since $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$, hence $\Gamma$ is inconsistent. □

**Proposition 9.16.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$, i.e., there is a closed tableau for

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

Using the $\neg\mathbb{T}$ rule, this can be turned into a closed tableau for

$$\{\mathbb{T}\neg\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}.$$

On the other hand, if there is a closed tableau for the latter, we can turn it into a closed tableau of the former by removing every formula that results from $\neg\mathbb{T}$applied to the first assumption $\mathbb{T}\neg\varphi$ as well as that assumption, and adding the assumption $\mathbb{F}\varphi$. For if a branch was closed before because it contained the conclusion of $\neg\mathbb{T}$applied to $\mathbb{T}\neg\varphi$, i.e., $\mathbb{F}\varphi$, the corresponding branch in the new tableau is also closed. If a branch in the old tableau was closed because it contained the assumption $\mathbb{T}\neg\varphi$ as well as $\mathbb{F}\neg\varphi$ we can turn it into a closed branch by applying $\neg\mathbb{F}$ to $\mathbb{F}\neg\varphi$ to obtain $\mathbb{T}\varphi$. This closes the branch since we added $\mathbb{F}\varphi$ as an assumption. □

**Proposition 9.17.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* Suppose $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$. Then there are $\psi_1, \ldots, \psi_n \in \Gamma$ such that

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

has a closed tableau. Replace the assumption $\mathbb{F}\varphi$by $\mathbb{T}\neg\varphi$, and insert the conclusion of $\neg\mathbb{T}$applied to $\mathbb{F}\varphi$after the assumptions. Any sentence in the tableau justified by appeal to line 1 in the old tableau is now justified by appeal to line $n + 1$. So if the old tableau was closed, the new one is. It shows that $\Gamma$ is inconsistent, since all assumptions are in $\Gamma$. □

**Proposition 9.18.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* If there are $\psi_1, \ldots, \psi_n \in \Gamma$ and $\chi_1, \ldots, \chi_m \in \Gamma$ such that

$$\{\mathbb{T}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$
$$\{\mathbb{T}\neg\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_m\}$$

both have closed tableaux, we can construct a tableau that shows that $\Gamma$ is inconsistent by using as assumptions $\mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n$ together with $\mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_m$, followed by an application of the Cut rule, yielding two branches, one starting with $\mathbb{T}\varphi$, the other with $\mathbb{F}\varphi$. Add on the part below the assumptions of the first tableau on the left side. Here, every rule application is still correct, and every branch closes. On the right side, add the part below the assumptions of the seond tableau, with the results of any applications of $\neg\mathbb{T}$ to $\mathbb{T}\neg\varphi$ removed.

For if a branch was closed before because it contained the conclusion of $\neg\mathbb{T}$applied to $\mathbb{T}\neg\varphi$, i.e., $\mathbb{F}\varphi$, as well as $\mathbb{F}\varphi$, the corresponding branch in the new tableau is also closed. If a branch in the old tableau was closed because it contained the assumption $\mathbb{T}\neg\varphi$ as well as $\mathbb{F}\neg\varphi$ we can turn it into a closed branch by applying $\neg\mathbb{F}$ to $\mathbb{F}\neg\varphi$ to obtain $\mathbb{T}\varphi$. □

## 9.7 Derivability and the Propositional Connectives

**Proposition 9.19.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$.*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. Both $\{\mathbb{F}\varphi, \mathbb{T}\varphi \wedge \psi\}$ and $\{\mathbb{F}\psi, \mathbb{T}\varphi \wedge \psi\}$ have closed tableaux

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,2$ |
| 4. | $\mathbb{T}\psi$ | $\wedge\mathbb{T}\,2$ |
| | $\otimes$ | |

| | | |
|---|---|---|
| 1. | $\mathbb{F}\psi$ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,2$ |
| 4. | $\mathbb{T}\psi$ | $\wedge\mathbb{T}\,2$ |
| | $\otimes$ | |

2. Here is a closed tableau for $\{\mathbb{T}\varphi, \mathbb{T}\psi, \mathbb{F}\varphi \wedge \psi\}$:

| | | | |
|---|---|---|---|
| 1. | $\mathbb{F}\varphi \wedge \psi$ | | Assumption |
| 2. | $\mathbb{T}\varphi$ | | Assumption |
| 3. | $\mathbb{T}\psi$ | | Assumption |
| 4. | $\mathbb{F}\varphi$ | $\mathbb{F}\psi$ | $\wedge\mathbb{F}\,1$ |
| | $\otimes$ | $\otimes$ | |

□

**Proposition 9.20.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. We give a closed tableau of $\{\mathbb{T}\varphi \vee \psi, \mathbb{T}\neg\varphi, \mathbb{T}\neg\psi\}$:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\varphi \vee \psi$ | Assumption |
| 2. | $\mathbb{T}\neg\varphi$ | Assumption |
| 3. | $\mathbb{T}\neg\psi$ | Assumption |
| 4. | $\mathbb{F}\varphi$ | $\neg\mathbb{T}\,2$ |
| 5. | $\mathbb{F}\psi$ | $\neg\mathbb{T}\,3$ |
| 6. | $\mathbb{T}\varphi$ $\quad$ $\mathbb{T}\psi$ <br> $\otimes$ $\quad$ $\otimes$ | $\vee\mathbb{T}\,1$ |

2. Both $\{\mathbb{F}\varphi \vee \psi, \mathbb{T}\varphi\}$ and $\{\mathbb{F}\varphi \vee \psi, \mathbb{T}\psi\}$ have closed tableaux:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \vee \psi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | Assumption |
| 3. | $\mathbb{F}\varphi$ | $\vee\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\vee\mathbb{F}\,1$ |
| | $\otimes$ | |

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \vee \psi$ | Assumption |
| 2. | $\mathbb{T}\psi$ | Assumption |
| 3. | $\mathbb{F}\varphi$ | $\vee\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\vee\mathbb{F}\,1$ |
| | $\otimes$ | |

□

**Proposition 9.21.** *1. $\varphi, \varphi \to \psi \vdash \psi$.*

*2. Both $\neg\varphi \vdash \varphi \to \psi$ and $\psi \vdash \varphi \to \psi$.*

*Proof.* 1. $\{\mathbb{F}\psi, \mathbb{T}\varphi \to \psi, \mathbb{T}\varphi\}$ has a closed tableau:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\psi$ | Assumption |
| 2. | $\mathbb{T}\varphi \to \psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | Assumption |
| 4. | $\mathbb{F}\varphi \quad \mathbb{T}\psi$ <br> $\otimes \quad \otimes$ | $\to\mathbb{T}\,2$ |

2. Both $\mathrm{s}\{\mathbb{F}\varphi \to \psi, \mathbb{T}\neg\varphi\}$ and $\{\mathbb{F}\varphi \to \psi, \mathbb{T}\neg\psi\}$ have closed tableaux:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \to \psi$ | Assumption |
| 2. | $\mathbb{T}\neg\varphi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\to\mathbb{F}\,1$ |
| 5. | $\mathbb{F}\varphi$ <br> $\otimes$ | $\neg\mathbb{T}\,2$ |

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \to \psi$ | Assumption |
| 2. | $\mathbb{T}\neg\psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\to\mathbb{F}\,1$ |
| 5. | $\mathbb{F}\psi$ <br> $\otimes$ | $\neg\mathbb{T}\,2$ |

□

## 9.8 Soundness

A derivation system, such as tableaux, is *sound* if it cannot derive things that do not actually hold. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable $\varphi$ is a tautology;
2. if a sentence is derivable from some others, it is also a consequence of them;
3. if a set of sentences is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

Because all these proof-theoretic properties are defined via closed tableaux of some kind or other, proving (1)–(3) above requires proving something about the semantic properties of closed tableaux. We will first define what it means for a signed formula to be satisfied in a structure, and then show that if a tableau is closed, no structure satisfies all its assumptions. (1)–(3) then follow as corollaries from this result.

**Definition 9.22.** A valuation $\mathfrak{v}$ *satisfies* a signed formula $\mathbb{T}\varphi$ iff $\mathfrak{v} \vDash \varphi$, and it satisfies $\mathbb{F}\varphi$ iff $\mathfrak{v} \nvDash \varphi$. $\mathfrak{v}$ satisfies a set of signed formulas $\Gamma$ iff it satisfies every $S\,\varphi \in \Gamma$. $\Gamma$ is *satisfiable* if there is a valuation that satisfies it, and *unsatisfiable* otherwise.

**Theorem 9.23** (Soundness)**.** *If $\Gamma$ has a closed tableau, $\Gamma$ is unsatisfiable.*

*Proof.* Let's call a branch of a tableau satisfiable iff the set of signed formulas on it is satisfiable, and let's call a tableau satisfiable if it contains at least one satisfiable branch.

We show the following: Extending a satisfiable tableau by one of the rules of inference always results in a satisfiable tableau. This will prove the theorem: any closed tableau results by applying rules of inference to the tableau consisting only of assumptions from $\Gamma$. So if $\Gamma$ were satisfiable, any tableau for it would be satisfiable. A closed tableau, however, is clearly not satisfiable: every branch contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$, and no structure can both satisfy and not satisfy $\varphi$.

Suppose we have a satisfiable tableau, i.e., a tableau with at least one satisfiable branch. Applying a rule of inference either adds signed formulas to a branch, or splits a branch in two. If the tableau has a satisfiable branch which is not extended by the rule application in question, it remains a satisfiable branch in the extended tableau, so the extended tableau is satisfiable. So we only have to consider the case where a rule is applied to a satisfiable branch.

Let $\Gamma$ be the set of signed formulas on that branch, and let $S\,\varphi \in \Gamma$ be the signed formula to which the rule is applied. If the rule does not result in a split branch, we have to show that the extended branch, i.e., $\Gamma$ together with the conclusions of the rule, is still satisfiable. If the rule results in split branch, we have to show that at least one of the two resulting branches is satisfiable.

First, we consider the possible inferences with only one premise.

1. The branch is expanded by applying $\neg\mathbb{T}$ to $\mathbb{T}\neg\psi \in \Gamma$. Then the extended branch contains the signed formulas $\Gamma \cup \{\mathbb{F}\psi\}$. Suppose $\mathfrak{v} \vDash \Gamma$. In particular, $\mathfrak{v} \vDash \neg\psi$. Thus, $\mathfrak{v} \nvDash \psi$, i.e., $\mathfrak{v}$ satisfies $\mathbb{F}\psi$.

2. The branch is expanded by applying $\neg\mathbb{F}$ to $\mathbb{F}\neg\psi \in \Gamma$: Exercise.

3. The branch is expanded by applying $\wedge\mathbb{T}$ to $\mathbb{T}\psi \wedge \chi \in \Gamma$, which results in two new signed formulas on the branch: $\mathbb{T}\psi$ and $\mathbb{T}\chi$. Suppose $\mathfrak{v} \vDash \Gamma$,

in particular $\mathfrak{v} \vDash \psi \wedge \chi$. Then $\mathfrak{v} \vDash \psi$ and $\mathfrak{v} \vDash \chi$. This means that $\mathfrak{v}$ satisfies both $\mathbb{T}\psi$ and $\mathbb{T}\chi$.

4. The branch is expanded by applying $\vee\mathbb{F}$ to $\mathbb{T}\psi \vee \chi \in \Gamma$: Exercise.

5. The branch is expanded by applying $\to\mathbb{F}$ to $\mathbb{T}\psi \to \chi \in \Gamma$: This results in two new signed formulas on the branch: $\mathbb{T}\psi$ and $\mathbb{F}\chi$. Suppose $\mathfrak{v} \vDash \Gamma$, in particular $\mathfrak{v} \nvDash \psi \to \chi$. Then $\mathfrak{v} \vDash \psi$ and $\mathfrak{v} \nvDash \chi$. This means that $\mathfrak{v}$ satisfies both $\mathbb{T}\psi$ and $\mathbb{F}\chi$.

Now let's consider the possible inferences with two premises.

1. The branch is expanded by applying $\wedge\mathbb{F}$ to $\mathbb{F}\psi \wedge \chi \in \Gamma$, which results in two branches, a left one continuing through $\mathbb{F}\psi$ and a right one through $\mathbb{F}\chi$. Suppose $\mathfrak{v} \vDash \Gamma$, in particular $\mathfrak{v} \nvDash \psi \wedge \chi$. Then $\mathfrak{v} \nvDash \psi$ or $\mathfrak{v} \nvDash \chi$. In the former case, $\mathfrak{v}$ satisfies $\mathbb{F}\psi$, i.e., $\mathfrak{v}$ satisfies the formulas on the left branch. In the latter, $\mathfrak{v}$ satisfies $\mathbb{F}\chi$, i.e., $\mathfrak{v}$ satisfies the formulas on the right branch.

2. The branch is expanded by applying $\vee\mathbb{T}$ to $\mathbb{T}\psi \vee \chi \in \Gamma$: Exercise.

3. The branch is expanded by applying $\to\mathbb{T}$ to $\mathbb{T}\psi \to \chi \in \Gamma$: Exercise.

4. The branch is expanded by Cut: This results in two branches, one containing $\mathbb{T}\psi$, the other containing $\mathbb{F}\psi$. Since $\mathfrak{v} \vDash \Gamma$ and either $\mathfrak{v} \vDash \psi$ or $\mathfrak{v} \nvDash \psi$, $\mathfrak{v}$ satisfies either the left or the right branch.

□

**Corollary 9.24.** *If $\vdash \varphi$ then $\varphi$ is a tautology.*

**Corollary 9.25.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* If $\Gamma \vdash \varphi$ then for some $\psi_1, \ldots, \psi_n \in \Gamma$, $\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$ has a closed tableau. By Theorem 9.23, every valuation $\mathfrak{v}$ either makes some $\psi_i$ false or makes $\varphi$ true. Hence, if $\mathfrak{v} \vDash \Gamma$ then also $\mathfrak{v} \vDash \varphi$. □

**Corollary 9.26.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then there are $\psi_1, \ldots, \psi_n \in \Gamma$ and a closed tableau for $\{\mathbb{T}\psi, \ldots, \mathbb{T}\psi\}$. By Theorem 9.23, there is no $\mathfrak{v}$ such that $\mathfrak{v} \vDash \psi_i$ for all $i = 1, \ldots, n$. But then $\Gamma$ is not satisfiable. □

## Problems

**Problem 9.1.** Give closed tableaux of the following:

1. $\mathbb{F}\,\neg(\varphi \to \psi) \to (\varphi \wedge \neg\psi)$
2. $\mathbb{F}\,(\varphi \to \chi) \vee (\psi \to \chi), \mathbb{T}\,(\varphi \wedge \psi) \to \chi$

**Problem 9.2.** Prove Proposition 9.13

**Problem 9.3.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 9.4.** Complete the proof of Theorem 9.23.

# Chapter 10

# Axiomatic Derivations

No effort has been made yet to ensure that the material in this chapter respects various tags indicating which connectives and quantifiers are primitive or defined: all are assumed to be primitive. If the FOL tag is true, we produce a version with quantifiers, otherwise without.

## 10.1 Rules and Derivations

Axiomatic derivations are perhaps the simplest proof system for logic. A derivation is just a sequence of formulas. To count as a derivation, every formula in the sequence must either be an instance of an axiom, or must follow from one or more formulas that precede it in the sequence by a rule of inference. A derivation derives its last formula.

**Definition 10.1** (Derivability)**.** If $\Gamma$ is a set of formulas of $\mathcal{L}$ then a *derivation* from $\Gamma$ is a finite sequence $\varphi_1, \ldots, \varphi_n$ of formulas where for each $i \leq n$ one of the following holds:

1. $\varphi_i \in \Gamma$; or
2. $\varphi_i$ is an axiom; or
3. $\varphi_i$ follows from some $\varphi_j$ (and $\varphi_k$) with $j < i$ (and $k < i$) by a rule of inference.

What counts as a correct derivation depends on which inference rules we allow (and of course what we take to be axioms). And an inference rule is an if-then statement that tells us that, under certain conditions, a step $A_i$ in is a correct inference step.

**Definition 10.2** (Rule of inference)**.** A *rule of inference* gives a sufficient condition for what counts as a correct inference step in a derivation from $\Gamma$.

For instance, since any one-element sequence $\varphi$ with $\varphi \in \Gamma$ trivially counts as a derivation, the following might be a very simple rule of inference:

> If $\varphi \in \Gamma$, then $\varphi$ is always a correct inference step in any derivation from $\Gamma$.

Similarly, if $\varphi$ is one of the axioms, then $\varphi$ by itself is a derivation, and so this is also a rule of inference:

> If $\varphi$ is an axiom, then $\varphi$ is a correct inference step.

It gets more interesting if the rule of inference appeals to formulas that appear before the step considered. The following rule is called *modus ponens:*

> If $\psi \to \varphi$ and $\psi$ occur higher up in the derivation, then $\varphi$ is a correct inference step.

If this is the only rule of inference, then our definition of derivation above amounts to this: $\varphi_1, \ldots, \varphi_n$ is a derivation iff for each $i \leq n$ one of the following holds:

1. $\varphi_i \in \Gamma$; or
2. $\varphi_i$ is an axiom; or
3. for some $j < i$, $\varphi_j$ is $\psi \to \varphi_i$, and for some $k < i$, $\varphi_k$ is $\psi$.

The last clause says that $\varphi_i$ follows from $\varphi_j$ ($\psi$) and $\varphi_k$ ($\psi \to \varphi_i$) by modus ponens. If we can go from 1 to $n$, and each time we find a formula $\varphi_i$ that is either in $\Gamma$, an axiom, or which a rule of inference tells us that it is a correct inference step, then the entire sequence counts as a correct derivation.

**Definition 10.3** (Derivability)**.** A formula $\varphi$ is *derivable* from $\Gamma$, written $\Gamma \vdash \varphi$, if there is a derivation from $\Gamma$ ending in $\varphi$.

**Definition 10.4** (Theorems)**.** A formula $\varphi$ is a *theorem* if there is a derivation of $\varphi$ from the empty set. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

## 10.2 Axiom and Rules for the Propositional Connectives

**Definition 10.5** (Axioms)**.** The set of $\mathrm{Ax}_0$ of *axioms* for the propositional connectives comprises all formulas of the following forms:

$$(\varphi \wedge \psi) \to \varphi \tag{10.1}$$
$$(\varphi \wedge \psi) \to \psi \tag{10.2}$$
$$\varphi \to (\psi \to (\varphi \wedge \psi)) \tag{10.3}$$
$$\varphi \to (\varphi \vee \psi) \tag{10.4}$$
$$\varphi \to (\psi \vee \varphi) \tag{10.5}$$
$$(\varphi \to \chi) \to ((\psi \to \chi) \to ((\varphi \vee \psi) \to \chi)) \tag{10.6}$$
$$\varphi \to (\psi \to \varphi) \tag{10.7}$$
$$(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi)) \tag{10.8}$$
$$(\varphi \to \psi) \to ((\varphi \to \neg\psi) \to \neg\varphi) \tag{10.9}$$
$$\neg\varphi \to (\varphi \to \psi) \tag{10.10}$$
$$\top \tag{10.11}$$
$$\bot \to \varphi \tag{10.12}$$
$$(\varphi \to \bot) \to \neg\varphi \tag{10.13}$$
$$\neg\neg\varphi \to \varphi \tag{10.14}$$

**Definition 10.6** (Modus ponens)**.** If $\psi$ and $\psi \to \varphi$ already occur in a derivation, then $\varphi$ is a correct inference step.

We'll abbreviate the rule modus ponens as "MP."

## 10.3 Examples of Derivations

**Example 10.7.** Suppose we want to prove $(\neg\theta \vee \alpha) \to (\theta \to \alpha)$. Clearly, this is not an instance of any of our axioms, so we have to use the MP rule to derive it. Our only rule is MP, which given $\varphi$ and $\varphi \to \psi$ allows us to justify $\psi$. One strategy would be to use eq. (10.6) with $\varphi$ being $\neg\theta$, $\psi$ being $\alpha$, and $\chi$ being $\theta \to \alpha$, i.e., the instance

$$(\neg\theta \to (\theta \to \alpha)) \to ((\alpha \to (\theta \to \alpha)) \to ((\neg\theta \vee \alpha) \to (\theta \to \alpha))).$$

Why? Two applications of MP yield the last part, which is what we want. And we easily see that $\neg\theta \to (\theta \to \alpha)$ is an instance of eq. (10.10), and $\alpha \to (\theta \to \alpha)$ is an instance of eq. (10.7). So our derivation is:

| | | |
|---|---|---|
| 1. | $\neg\theta \to (\theta \to \alpha)$ | eq. (10.7) |
| 2. | $(\neg\theta \to (\theta \to \alpha)) \to$ $((\alpha \to (\theta \to \alpha)) \to ((\neg\theta \vee \alpha) \to (\theta \to \alpha)))$ | eq. (10.6) |
| 3. | $((\alpha \to (\theta \to \alpha)) \to ((\neg\theta \vee \alpha) \to (\theta \to \alpha))$ | 1, 2, MP |
| 4. | $\alpha \to (\theta \to \alpha)$ | eq. (10.7) |
| 5. | $(\neg\theta \vee \alpha) \to (\theta \to \alpha)$ | 3, 4, MP |

**Example 10.8.** Let's try to find a derivation of $\theta \to \theta$. It is not an instance of an axiom, so we have to use MP to derive it. eq. (10.7) is an axiom of the form $\varphi \to \psi$ to which we could apply MP. To be useful, of course, the $\psi$ which MP would justify as a correct step in this case would have to be $\theta \to \theta$, since this is what we want to derive. That means $\varphi$ would also have to be $\theta$, i.e., we might look at this instance of eq. (10.7):

$$\theta \to (\theta \to \theta)$$

In order to apply MP, we would also need to justify the corresponding second premise, namely $\varphi$. But in our case, that would be $\theta$, and we won't be able to derive $\theta$ by itself. So we need a different strategy.

The other axiom involving just $\to$ is eq. (10.8), i.e.,

$$(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$$

We could get to the last nested conditional by applying MP twice. Again, that would mean that we want an instance of eq. (10.8) where $\varphi \to \chi$ is $\theta \to \theta$, the formula we are aiming for. Then of course, $\varphi$ and $\chi$ are both $\theta$. How should we pick $\psi$ so that both $\varphi \to (\psi \to \chi)$ and $\varphi \to \psi$, i.e., in our case $\theta \to (\psi \to \theta)$ and $\theta \to \psi$, are also derivable? Well, the first of these is already an instance of eq. (10.7), whatever we decide $\psi$ to be. And $\theta \to \psi$ would be another instance of eq. (10.7) if $\psi$ were $(\theta \to \theta)$. So, our derivation is:

| | | |
|---|---|---|
| 1. | $\theta \to ((\theta \to \theta) \to \theta)$ | eq. (10.7) |
| 2. | $(\theta \to ((\theta \to \theta) \to \theta)) \to$ $((\theta \to (\theta \to \theta)) \to (\theta \to \theta))$ | eq. (10.8) |
| 3. | $(\theta \to (\theta \to \theta)) \to (\theta \to \theta)$ | 1, 2, MP |
| 4. | $\theta \to (\theta \to \theta)$ | eq. (10.7) |
| 5. | $\theta \to \theta$ | 3, 4, MP |

**Example 10.9.** Sometimes we want to show that there is a derivation of some formula from some other formulas $\Gamma$. For instance, let's show that we can derive $\varphi \to \chi$ from $\Gamma = \{\varphi \to \psi, \psi \to \chi\}$.

| | | |
|---|---|---|
| 1. | $\varphi \to \psi$ | HYP |
| 2. | $\psi \to \chi$ | HYP |
| 3. | $(\psi \to \chi) \to (\varphi \to (\psi \to \chi))$ | eq. (10.7) |
| 4. | $\varphi \to (\psi \to \chi)$ | 2, 3, MP |
| 5. | $(\varphi \to (\psi \to \chi)) \to$ $((\varphi \to \psi) \to (\varphi \to \chi))$ | eq. (10.8) |
| 6. | $((\varphi \to \psi) \to (\varphi \to \chi))$ | 4, 5, MP |
| 7. | $\varphi \to \chi$ | 1, 6, MP |

The lines labelled "HYP" (for "hypothesis") indicate that the formula on that line is an element of $\Gamma$.

**Proposition 10.10.** *If $\Gamma \vdash \varphi \to \psi$ and $\Gamma \vdash \psi \to \chi$, then $\Gamma \vdash \varphi \to \chi$*

*Proof.* Suppose $\Gamma \vdash \varphi \to \psi$ and $\Gamma \vdash \psi \to \chi$. Then there is a derivation of $\varphi \to \psi$ from $\Gamma$; and a derivation of $\psi \to \chi$ from $\Gamma$ as well. Combine these into a single derivation by concatenating them. Now add lines 3–7 of the derivation in the preceding example. This is a derivation of $\varphi \to \chi$—which is the last line of the new derivation—from $\Gamma$. Note that the justifications of lines 4 and 7 remain valid if the reference to line number 2 is replaced by reference to the last line of the derivation of $\varphi \to \psi$, and reference to line number 1 by reference to the last line of the derivation of $B \to \chi$. □

## 10.4 Proof-Theoretic Notions

Just as we've defined a number of important semantic notions (tautology, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the derivability or non-derivability of certain formulas. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorems*.

**Definition 10.11** (Derivability)**.** A formula $\varphi$ is *derivable* from $\Gamma$, written $\Gamma \vdash \varphi$, if there is a derivation from $\Gamma$ ending in $\varphi$.

**Definition 10.12** (Theorems)**.** A formula $\varphi$ is a *theorem* if there is a derivation of $\varphi$ from the empty set. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 10.13** (Consistency)**.** A set $\Gamma$ of formulas is *consistent* if and only if $\Gamma \nvdash \bot$; it is *inconsistent* otherwise.

**Proposition 10.14** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* The formula $\varphi$ by itself is a derivation of $\varphi$ from $\Gamma$. □

**Proposition 10.15** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Any derivation of $\varphi$ from $\Gamma$ is also a derivation of $\varphi$ from $\Delta$. □

**Proposition 10.16** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* Suppose $\{\varphi\} \cup \Delta \vdash \psi$. Then there is a derivation $\psi_1, \ldots, \psi_l = \psi$ from $\{\varphi\} \cup \Delta$. Some of the steps in that derivation will be correct because of a rule which refers to a prior line $\psi_i = \varphi$. By hypothesis, there is a derivation of $\varphi$ from $\Gamma$, i.e., a derivation $\varphi_1, \ldots, \varphi_k = \varphi$ where every $\varphi_i$ is an axiom, an element of $\Gamma$, or correct by a rule of inference. Now consider the sequence

$$\varphi_1, \ldots, \varphi_k = \varphi, \psi_1, \ldots, \psi_l = \psi.$$

This is a correct derivation of $\psi$ from $\Gamma \cup \Delta$ since every $B_i = \varphi$ is now justified by the same rule which justifies $\varphi_k = \varphi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 10.17.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every $\varphi$.*

*Proof.* Exercise. □

**Proposition 10.18** (Compactness)**.**

1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*
2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.*

1. If $\Gamma \vdash \varphi$, then there is a finite sequence of formulas $\varphi_1, \ldots, \varphi_n$ so that $\varphi \equiv \varphi_n$ and each $\varphi_i$ is either a logical axiom, an element of $\Gamma$ or follows from previous formulas by modus ponens. Take $\Gamma_0$ to be those $\varphi_i$ which are in $\Gamma$. Then the derivation is likewise a derivation from $\Gamma_0$, and so $\Gamma_0 \vdash \varphi$.
2. This is the contrapositive of (1) for the special case $\varphi \equiv \bot$.

□

## 10.5 The Deduction Theorem

As we've seen, giving derivations in an axiomatic system is cumbersome, and derivations may be hard to find. Rather than actually write out long lists of formulas, it is generally easier to argue that such derivations exist, by making use of a few simple results. We've already established three such results: Proposition 10.14 says we can always assert that $\Gamma \vdash \varphi$ when we know that $\varphi \in \Gamma$. Proposition 10.15 says that if $\Gamma \vdash \varphi$ then also $\Gamma \cup \{\psi\} \vdash \varphi$. And Proposition 10.16 implies that if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. Here's another simple result, a "meta"-version of modus ponens:

**Proposition 10.19.** *If $\Gamma \vdash \varphi$ and $\Gamma \vdash \varphi \to \psi$, then $\Gamma \vdash \psi$.*

*Proof.* We have that $\{\varphi, \varphi \to \psi\} \vdash \psi$:

1. $\varphi$ Hyp.
2. $\varphi \to \psi$ Hyp.
3. $\psi$ 1, 2, MP

By Proposition 10.16, $\Gamma \vdash \psi$. □

The most important result we'll use in this context is the deduction theorem:

**Theorem 10.20** (Deduction Theorem)**.** *$\Gamma \cup \{\varphi\} \vdash \psi$ if and only if $\Gamma \vdash \varphi \to \psi$.*

*Proof.* The "if" direction is immediate. If $\Gamma \vdash \varphi \to \psi$ then also $\Gamma \cup \{\varphi\} \vdash \varphi \to \psi$ by Proposition 10.15. Also, $\Gamma \cup \{\varphi\} \vdash \varphi$ by Proposition 10.14. So, by Proposition 10.19, $\Gamma \cup \{\varphi\} \vdash \psi$.

For the "only if" direction, we proceed by induction on the length of the derivation of $\psi$ from $\Gamma \cup \{\varphi\}$.

For the induction basis, we prove the claim for every derivation of length 1. A derivation of $\psi$ from $\Gamma \cup \{\varphi\}$ of length 1 consists of $\psi$ by itself; and if it is correct $\psi$ is either $\in \Gamma \cup \{\varphi\}$ or is an axiom. If $\psi \in \Gamma$ or is an axiom, then $\Gamma \vdash \psi$. We also have that $\Gamma \vdash \psi \to (\varphi \to \psi)$ by eq. (10.7), and Proposition 10.19 gives $\Gamma \vdash \varphi \to \psi$. If $\psi \in \{\varphi\}$ then $\Gamma \vdash \varphi \to \psi$ because then last sentence $\varphi \to \psi$ is the same as $\varphi \to \varphi$, and we have derived that in Example 10.8.

For the inductive step, suppose a derivation of $\psi$ from $\Gamma \cup \{\varphi\}$ ends with a step $\psi$ which is justified by modus ponens. (If it is not justified by modus ponens, $\psi \in \Gamma$, $\psi \equiv \varphi$, or $\psi$ is an axiom, and the same reasoning as in the induction basis applies.) Then some previous steps in the derivation are $\chi \to \psi$ and $\chi$, for some formula $\chi$, i.e., $\Gamma \cup \{\varphi\} \vdash \chi \to \psi$ and $\Gamma \cup \{\varphi\} \vdash \chi$, and the respective derivations are shorter, so the inductive hypothesis applies to them. We thus have both:

$$\Gamma \vdash \varphi \to (\chi \to \psi);$$
$$\Gamma \vdash \varphi \to \chi.$$

But also

$$\Gamma \vdash (\varphi \to (\chi \to \psi)) \to ((\varphi \to \chi) \to (\varphi \to \psi)),$$

by eq. (10.8), and two applications of Proposition 10.19 give $\Gamma \vdash \varphi \to \psi$, as required. □

Notice how eq. (10.7) and eq. (10.8) were chosen precisely so that the Deduction Theorem would hold.

The following are some useful facts about derivability, which we leave as exercises.

**Proposition 10.21.** *1.* $\vdash (\varphi \to \psi) \to ((\psi \to \chi) \to (\varphi \to \chi))$;

2. *If* $\Gamma \cup \{\neg\varphi\} \vdash \neg\psi$ *then* $\Gamma \cup \{\psi\} \vdash \varphi$ *(Contraposition);*

3. $\{\varphi, \neg\varphi\} \vdash \psi$ *(Ex Falso Quodlibet, Explosion);*

4. $\{\neg\neg\varphi\} \vdash \varphi$ *(Double Negation Elimination);*

5. *If* $\Gamma \vdash \neg\neg\varphi$ *then* $\Gamma \vdash \varphi$*;*

## 10.6 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 10.22.** *If* $\Gamma \vdash \varphi$ *and* $\Gamma \cup \{\varphi\}$ *is inconsistent, then* $\Gamma$ *is inconsistent.*

*Proof.* If $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma \cup \{\varphi\} \vdash \bot$. By Proposition 10.14, $\Gamma \vdash \psi$ for every $\psi \in \Gamma$. Since also $\Gamma \vdash \varphi$ by hypothesis, $\Gamma \vdash \psi$ for every $\psi \in \Gamma \cup \{\varphi\}$. By Proposition 10.16, $\Gamma \vdash \bot$, i.e., $\Gamma$ is inconsistent. □

**Proposition 10.23.** $\Gamma \vdash \varphi$ *iff* $\Gamma \cup \{\neg\varphi\}$ *is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$. Then $\Gamma \cup \{\neg\varphi\} \vdash \varphi$ by Proposition 10.15. $\Gamma \cup \{\neg\varphi\} \vdash \neg\varphi$ by Proposition 10.14. We also have $\vdash \neg\varphi \to (\varphi \to \bot)$ by eq. (10.10). So by two applications of Proposition 10.19, we have $\Gamma \cup \{\neg\varphi\} \vdash \bot$.

Now assume $\Gamma \cup \{\neg\varphi\}$ is inconsistent, i.e., $\Gamma \cup \{\neg\varphi\} \vdash \bot$. By the deduction theorem, $\Gamma \vdash \neg\varphi \to \bot$. $\Gamma \vdash (\neg\varphi \to \bot) \to \neg\neg\varphi$ by eq. (10.13), so $\Gamma \vdash \neg\neg\varphi$ by Proposition 10.19. Since $\Gamma \vdash \neg\neg\varphi \to \varphi$ (eq. (10.14)), we have $\Gamma \vdash \varphi$ by Proposition 10.19 again. □

**Proposition 10.24.** *If* $\Gamma \vdash \varphi$ *and* $\neg\varphi \in \Gamma$*, then* $\Gamma$ *is inconsistent.*

*Proof.* $\Gamma \vdash \neg\varphi \to (\varphi \to \bot)$ by eq. (10.10). $\Gamma \vdash \bot$ by two applications of Proposition 10.19. □

**Proposition 10.25.** *If* $\Gamma \cup \{\varphi\}$ *and* $\Gamma \cup \{\neg\varphi\}$ *are both inconsistent, then* $\Gamma$ *is inconsistent.*

*Proof.* Exercise. □

## 10.7 Derivability and the Propositional Connectives

**Proposition 10.26.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. From eq. (10.1) and eq. (10.1) by modus ponens.

2. From eq. (10.3) by two applications of modus ponens.

□

**Proposition 10.27.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. From eq. (10.9) we get $\vdash \neg\varphi \to (\varphi \to \bot)$ and $\vdash \neg\varphi \to (\varphi \to \bot)$. So by the deduction theorem, we have $\{\neg\varphi\} \vdash \varphi \to \bot$ and $\{\neg\psi\} \vdash \psi \to \bot$. From eq. (10.6) we get $\{\neg\varphi, \neg\psi\} \vdash (\varphi \vee \psi) \to \bot$. By the deduction theorem, $\{\varphi \vee \psi, \neg\varphi, \neg\psi\} \vdash \bot$.

2. From eq. (10.4) and eq. (10.5) by modus ponsens.

□

**Proposition 10.28.** *1. $\varphi, \varphi \to \psi \vdash \psi$.*

*2. Both $\neg\varphi \vdash \varphi \to \psi$ and $\psi \vdash \varphi \to \psi$.*

*Proof.* 1. We can derive:

| | | |
|---|---|---|
| 1. | $\varphi$ | HYP |
| 2. | $\varphi \to \psi$ | HYP |
| 3. | $\psi$ | 1, 2, MP |

2. By eq. (10.10) and eq. (10.7) and the deduction theorem, respectively.

□

## 10.8 Soundness

A derivation system, such as axiomatic deduction, is *sound* if it cannot derive things that do not actually hold. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable $\varphi$ is valid;
2. if $\varphi$ is derivable from some others $\Gamma$, it is also a consequence of them;
3. if a set of formulas $\Gamma$ is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

**Proposition 10.29.** *If $\varphi$ is an axiom, then $\mathfrak{v} \vDash \varphi$ for each valuation $\mathfrak{v}$.*

*Proof.* Do truth tables for each axiom to verify that they are tautologies. □

**Theorem 10.30** (Soundness)**.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* By induction on the length of the derivation of $\varphi$ from $\Gamma$. If there are no steps justified by inferences, then all formulas in the derivation are either instances of axioms or are in $\Gamma$. By the previous proposition, all the axioms are tautologies, and hence if $\varphi$ is an axiom then $\Gamma \vDash \varphi$. If $\varphi \in \Gamma$, then trivially $\Gamma \vDash \varphi$.

If the last step of the derivation of $\varphi$ is justified by modus ponens, then there are formulas $\psi$ and $\psi \to \varphi$ in the derivation, and the induction hypothesis applies to the part of the derivation ending in those formulas (since they contain at least one fewer steps justified by an inference). So, by induction hypothesis, $\Gamma \vDash \psi$ and $\Gamma \vDash \psi \to \varphi$. Then $\Gamma \vDash \varphi$ by Theorem 5.17.

□

**Corollary 10.31.** *If $\vdash \varphi$, then $\varphi$ is a tautology.*

**Corollary 10.32.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then $\Gamma \vdash \bot$, i.e., there is a derivation of $\bot$ from $\Gamma$. By Theorem 10.30, any valuation $\mathfrak{v}$ that satisfies $\Gamma$ must satisfy $\bot$. Since $\mathfrak{v} \nvDash \bot$ for every valuation $\mathfrak{v}$, no $\mathfrak{v}$ can satisfy $\Gamma$, i.e., $\Gamma$ is not satisfiable. □

## Problems

**Problem 10.1.** Show that the following hold by exhibiting derivations from the axioms:

1. $(\varphi \wedge \psi) \to (\psi \wedge \varphi)$
2. $((\varphi \wedge \psi) \to \chi) \to (\varphi \to (\psi \to \chi))$
3. $\neg(\varphi \vee \psi) \to \neg\varphi$

**Problem 10.2.** Prove Proposition 10.17.

**Problem 10.3.** Prove Proposition 10.21

**Problem 10.4.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 10.5.** Prove Proposition 10.25

# Chapter 11

# The Completeness Theorem

## 11.1 Introduction

The completeness theorem is one of the most fundamental results about logic. It comes in two formulations, the equivalence of which we'll prove. In its first formulation it says something fundamental about the relationship between semantic consequence and our proof system: if a sentence $\varphi$ follows from some sentences $\Gamma$, then there is also a derivation that establishes $\Gamma \vdash \varphi$. Thus, the proof system is as strong as it can possibly be without proving things that don't actually follow.

In its second formulation, it can be stated as a model existence result: every consistent set of sentences is satisfiable. Consistency is a proof-theoretic notion: it says that our proof system is unable to produce certain derivations. But who's to say that just because there are no derivations of a certain sort from $\Gamma$, it's guaranteed that there is valuation $\mathfrak{v}$ with $\mathfrak{v} \vDash \Gamma$? Before the completeness theorem was first proved—in fact before we had the proof systems we now do—the great German mathematician David Hilbert held the view that consistency of mathematical theories guarantees the existence of the objects they are about. He put it as follows in a letter to Gottlob Frege:

> If the arbitrarily given axioms do not contradict one another with all their consequences, then they are true and the things defined by the axioms exist. This is for me the criterion of truth and existence.

Frege vehemently disagreed. The second formulation of the completeness theorem shows that Hilbert was right in at least the sense that if the axioms are consistent, then *some* valuation exists that makes them all true.

These aren't the only reasons the completeness theorem—or rather, its proof—is important. It has a number of important consequences, some of which we'll discuss separately. For instance, since any derivation that shows $\Gamma \vdash \varphi$ is finite and so can only use finitely many of the sentences in $\Gamma$, it follows by the completeness theorem that if $\varphi$ is a consequence of $\Gamma$, it is already

a consequence of a finite subset of $\Gamma$. This is called *compactness*. Equivalently, if every finite subset of $\Gamma$ is consistent, then $\Gamma$ itself must be consistent.

Although the compactness theorem follows from the completeness theorem via the detour through derivations, it is also possible to use the *the proof of* the completeness theorem to establish it directly. For what the proof does is take a set of sentences with a certain property—consistency—and constructs a structure out of this set that has certain properties (in this case, that it satisfies the set). Almost the very same construction can be used to directly establish compactness, by starting from "finitely satisfiable" sets of sentences instead of consistent ones.

## 11.2 Outline of the Proof

The proof of the completeness theorem is a bit complex, and upon first reading it, it is easy to get lost. So let us outline the proof. The first step is a shift of perspective, that allows us to see a route to a proof. When completeness is thought of as "whenever $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$," it may be hard to even come up with an idea: for to show that $\Gamma \vdash \varphi$ we have to find a derivation, and it does not look like the hypothesis that $\Gamma \vDash \varphi$ helps us for this in any way. For some proof systems it is possible to directly construct a derivation, but we will take a slightly different tack. The shift in perspective required is this: completeness can also be formulated as: "if $\Gamma$ is consistent, it has a model." Perhaps we can use the information in $\Gamma$ together with the hypothesis that it is consistent to construct a model. After all, we know what kind of model we are looking for: one that is as $\Gamma$ describes it!

If $\Gamma$ contains only propositional variables, it is easy to construct a model for it. All we have to do is come up with a valuation $\mathfrak{v}$ such that $\mathfrak{v} \vDash p$a for all $p \in \Gamma$. Well, let $\mathfrak{v}(p) = \mathbb{T}$ iff $p \in \Gamma$.

Now suppose $\Gamma$ contains some formula $\neg\psi$, with $\psi$ atomic. We might worry that the construction of $\mathfrak{v}$ interferes with the possibility of making $\neg\psi$ true. But here's where the consistency of $\Gamma$ comes in: if $\neg\psi \in \Gamma$, then $\psi \notin \Gamma$, or else $\Gamma$ would be inconsistent. And if $\psi \notin \Gamma$, then according to our construction of $\mathfrak{v}$, $\mathfrak{v} \nvDash \psi$, so $\mathfrak{v} \vDash \neg\psi$. So far so good.

What if $\Gamma$ contains complex, non-atomic formulas? Say it contains $\varphi \wedge \psi$. To make that true, we should proceed as if both $\varphi$ and $\psi$ were in $\Gamma$. And if $\varphi \vee \psi \in \Gamma$, then we will have to make at least one of them true, i.e., proceed as if one of them was in $\Gamma$.

This suggests the following idea: we add additional formulas to $\Gamma$ so as to (a) keep the resulting set consistent and (b) make sure that for every possible atomic sentence $\varphi$, either $\varphi$ is in the resulting set, or $\neg\varphi$ is, and (c) such that, whenever $\varphi \wedge \psi$ is in the set, so are both $\varphi$ and $\psi$, if $\varphi \vee \psi$ is in the set, at least one of $\varphi$ or $\psi$ is also, etc. We keep doing this (potentially forever). Call the set of all formulas so added $\Gamma^*$. Then our construction above would provide us

with a structure $\mathfrak{v}$ for which we could prove, by induction, that all sentences in $\Gamma^*$ are true in it, and hence also all sentence in $\Gamma$ since $\Gamma \subseteq \Gamma^*$. It turns out that guaranteeing (a) and (b) is enough. A set of sentences for which (b) holds is called *complete*. So our task will be to extend the consistent set $\Gamma$ to a consistent and complete set $\Gamma^*$.

So here's what we'll do. First we investigate the properties of complete consistent sets, in particular we prove that a complete consistent set contains $\varphi \wedge \psi$ iff it contains both $\varphi$ and $\psi$, $\varphi \vee \psi$ iff it contains at least one of them, etc. (Proposition 11.2). We'll then take the consistent set $\Gamma$ and show that it can be extended to a consistent and complete set $\Gamma^*$ (Lemma 11.3). This set $\Gamma^*$ is what we'll use to define our valuation $\mathfrak{v}(\Gamma^*)$. The valuation is determined by the propositional variables in $\Gamma^*$ (Definition 11.4). We'll use the properties of complete consistent sets to show that indeed $\mathfrak{v}(\Gamma^*) \vDash \varphi$ iff $\varphi \in \Gamma^*$ (Lemma 11.5), and thus in particular, $\mathfrak{v}(\Gamma^*) \vDash \Gamma$.

## 11.3 Complete Consistent Sets of Sentences

**Definition 11.1** (Complete set)**.** A set $\Gamma$ of sentences is *complete* iff for any sentence $\varphi$, either $\varphi \in \Gamma$ or $\neg\varphi \in \Gamma$.

Complete sets of sentences leave no questions unanswered. For any sentence $A$, $\Gamma$ "says" if $\varphi$ is true or false. The importance of complete sets extends beyond the proof of the completeness theorem. A theory which is complete and axiomatizable, for instance, is always decidable.

Complete consistent sets are important in the completeness proof since we can guarantee that every consistent set of sentences $\Gamma$ is contained in a complete consistent set $\Gamma^*$. A complete consistent set contains, for each sentence $\varphi$, either $\varphi$ or its negation $\neg\varphi$, but not both. This is true in particular for atomic sentences, so from a complete consistent set in a language suitably expanded by constant symbols, we can construct a structure where the interpretation of predicate symbols is defined according to which atomic sentences are in $\Gamma^*$. This structure can then be shown to make all sentences in $\Gamma^*$ (and hence also all those in $\Gamma$) true. The proof of this latter fact requires that $\neg\varphi \in \Gamma^*$ iff $\varphi \notin \Gamma^*$, $(\varphi \vee \psi) \in \Gamma^*$ iff $\varphi \in \Gamma^*$ or $\psi \in \Gamma^*$, etc.

In what follows, we will often tacitly use the properties of reflexivity, monotonicity, and transitivity of $\vdash$ (see sections 7.6, 8.5, 9.5 and 10.4).

**Proposition 11.2.** *Suppose $\Gamma$ is complete and consistent. Then:*

1. *If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.*
2. *$\varphi \wedge \psi \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$.*
3. *$\varphi \vee \psi \in \Gamma$ iff either $\varphi \in \Gamma$ or $\psi \in \Gamma$.*
4. *$\varphi \to \psi \in \Gamma$ iff either $\varphi \notin \Gamma$ or $\psi \in \Gamma$.*

*Proof.* Let us suppose for all of the following that $\Gamma$ is complete and consistent.

1. If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.

   Suppose that $\Gamma \vdash \varphi$. Suppose to the contrary that $\varphi \notin \Gamma$. Since $\Gamma$ is complete, $\neg\varphi \in \Gamma$. By Propositions 8.17, 9.17, 7.19 and 10.24, $\Gamma$ is inconsistent. This contradicts the assumption that $\Gamma$ is consistent. Hence, it cannot be the case that $\varphi \notin \Gamma$, so $\varphi \in \Gamma$.

2. Exercise.

3. First we show that if $\varphi \vee \psi \in \Gamma$, then either $\varphi \in \Gamma$ or $\psi \in \Gamma$. Suppose $\varphi \vee \psi \in \Gamma$ but $\varphi \notin \Gamma$ and $\psi \notin \Gamma$. Since $\Gamma$ is complete, $\neg\varphi \in \Gamma$ and $\neg\psi \in \Gamma$. By Propositions 8.20, 9.20, 7.22 and 10.27, item (1), $\Gamma$ is inconsistent, a contradiction. Hence, either $\varphi \in \Gamma$ or $\psi \in \Gamma$.

   For the reverse direction, suppose that $\varphi \in \Gamma$ or $\psi \in \Gamma$. By Propositions 8.20, 9.20, 7.22 and 10.27, item (2), $\Gamma \vdash \varphi \vee \psi$. By (1), $\varphi \vee \psi \in \Gamma$, as required.

4. Exercise.

□

## 11.4 Lindenbaum's Lemma

We now prove a lemma that shows that any consistent set of sentences is contained in some set of sentences which is not just consistent, but also complete. The proof works by adding one sentence at a time, guaranteeing at each step that the set remains consistent. We do this so that for every $\varphi$, either $\varphi$ or $\neg\varphi$ gets added at some stage. The union of all stages in that construction then contains either $\varphi$ or its negation $\neg\varphi$ and is thus complete. It is also consistent, since we made sure at each stage not to introduce an inconsistency.

**Lemma 11.3** (Lindenbaum's Lemma)**.** *Every consistent set $\Gamma$ in a language $\mathcal{L}$ can be extended to a complete and consistent set $\Gamma^*$.*

*Proof.* Let $\Gamma$ be consistent. Let $\varphi_0$, $\varphi_1$, ... be an enumeration of all the sentences of $\mathcal{L}$. Define $\Gamma_0 = \Gamma$, and

$$\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{\varphi_n\} & \text{if } \Gamma_n \cup \{\varphi_n\} \text{ is consistent;} \\ \Gamma_n \cup \{\neg\varphi_n\} & \text{otherwise.} \end{cases}$$

Let $\Gamma^* = \bigcup_{n \geq 0} \Gamma_n$.

Each $\Gamma_n$ is consistent: $\Gamma_0$ is consistent by definition. If $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$, this is because the latter is consistent. If it isn't, $\Gamma_{n+1} = \Gamma_n \cup \{\neg\varphi_n\}$. We have to verify that $\Gamma_n \cup \{\neg\varphi_n\}$ is consistent. Suppose it's not. Then *both* $\Gamma_n \cup \{\varphi_n\}$

and $\Gamma_n \cup \{\neg\varphi_n\}$ are inconsistent. This means that $\Gamma_n$ would be inconsistent by Propositions 8.17, 9.17, 7.19 and 10.24, contrary to the induction hypothesis.

For every $n$ and every $i < n$, $\Gamma_i \subseteq \Gamma_n$. This follows by a simple induction on $n$. For $n = 0$, there are no $i < 0$, so the claim holds automatically. For the inductive step, suppose it is true for $n$. We have $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$ or $= \Gamma_n \cup \{\neg\varphi_n\}$ by construction. So $\Gamma_n \subseteq \Gamma_{n+1}$. If $i < n$, then $\Gamma_i \subseteq \Gamma_n$ by inductive hypothesis, and so $\subseteq \Gamma_{n+1}$ by transitivity of $\subseteq$.

From this it follows that every finite subset of $\Gamma^*$ is a subset of $\Gamma_n$ for some $n$, since each $\psi \in \Gamma^*$ not already in $\Gamma_0$ is added at some stage $i$. If $n$ is the last one of these, then all $\psi$ in the finite subset are in $\Gamma_n$. So, every finite subset of $\Gamma^*$ is consistent. By Propositions 8.14, 9.14, 7.16 and 10.18, $\Gamma^*$ is consistent.

Every sentence of $\mathrm{Frm}(\mathcal{L})$ appears on the list used to define $\Gamma^*$. If $\varphi_n \notin \Gamma^*$, then that is because $\Gamma_n \cup \{\varphi_n\}$ was inconsistent. But then $\neg\varphi_n \in \Gamma^*$, so $\Gamma^*$ is complete. □

## 11.5 Construction of a Model

We are now ready to define a valuation that makes all $\varphi \in \Gamma$ true. To do this, we first apply Lindenbaum's Lemma: we get a complete consistent $\Gamma^* \supseteq \Gamma$. We let the propositional variables in $\Gamma^*$ determine $\mathfrak{v}(\Gamma^*)$.

**Definition 11.4.** Suppose $\Gamma^*$ is a complete consistent set of formulas. Then we let

$$\mathfrak{v}(\Gamma^*)(p) = \begin{cases} \mathbb{T} & \text{if } p \in \Gamma^* \\ \mathbb{F} & \text{if } p \notin \Gamma^* \end{cases}$$

**Lemma 11.5** (Truth Lemma)**.** $\mathfrak{v}(\Gamma^*) \vDash \varphi$ *iff* $\varphi \in \Gamma^*$.

*Proof.* We prove both directions simultaneously, and by induction on $\varphi$.

1. $\varphi \equiv \bot$: $\mathfrak{v}(\Gamma^*) \nvDash \bot$ by definition of satisfaction. On the other hand, $\bot \notin \Gamma^*$ since $\Gamma^*$ is consistent.

2. $\varphi \equiv p$: $\mathfrak{v}(\Gamma^*) \vDash p$ iff $\mathfrak{v}(\Gamma^*)(p) = \mathbb{T}$ (by the definition of satisfaction) iff $p \in \Gamma^*$ (by the construction of $\mathfrak{v}(\Gamma^*)$).

3. $\varphi \equiv \neg\psi$: $\mathfrak{v}(\Gamma^*) \vDash \varphi$ iff $\mathfrak{M}(\Gamma^*) \nvDash \psi$ (by definition of satisfaction). By induction hypothesis, $\mathfrak{M}(\Gamma^*) \nvDash \psi$ iff $\psi \notin \Gamma^*$. Since $\Gamma^*$ is consistent and complete, $\psi \notin \Gamma^*$ iff $\neg\psi \in \Gamma^*$.

4. $\varphi \equiv \psi \wedge \chi$: exercise.

5. $\varphi \equiv \psi \vee \chi$: $\mathfrak{v}(\Gamma^*) \vDash \varphi$ iff at $\mathfrak{v}(\Gamma^*) \vDash \psi$ or $\mathfrak{v}(\Gamma^*) \vDash \chi$ (by definition of satisfaction) iff $\psi \in \Gamma^*$ or $\chi \in \Gamma^*$ (by induction hypothesis). This is the case iff $(\psi \vee \chi) \in \Gamma^*$ (by Proposition 11.2(3)).

6. $\varphi \equiv \psi \to \chi$: exercise.

□

## 11.6 The Completeness Theorem

Let's combine our results: we arrive at the completeness theorem.

**Theorem 11.6** (Completeness Theorem)**.** *Let $\Gamma$ be a set of sentences. If $\Gamma$ is consistent, it is satisfiable.*

*Proof.* Suppose $\Gamma$ is consistent. By Lemma 11.3, there is a $\Gamma^* \supseteq \Gamma$ which is consistent and complete. By Lemma 11.5, $\mathfrak{v}(\Gamma^*) \vDash \varphi$ iff $\varphi \in \Gamma^*$. From this it follows in particular that for all $\varphi \in \Gamma$, $\mathfrak{v}(\Gamma^*) \vDash \varphi$, so $\Gamma$ is satisfiable. □

**Corollary 11.7** (Completeness Theorem, Second Version)**.** *For all $\Gamma$ and $\varphi$ sentences: if $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.*

*Proof.* Note that the $\Gamma$'s in Corollary 11.7 and Theorem 11.6 are universally quantified. To make sure we do not confuse ourselves, let us restate Theorem 11.6 using a different variable: for any set of sentences $\Delta$, if $\Delta$ is consistent, it is satisfiable. By contraposition, if $\Delta$ is not satisfiable, then $\Delta$ is inconsistent. We will use this to prove the corollary.

Suppose that $\Gamma \vDash \varphi$. Then $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable by Proposition 5.16. Taking $\Gamma \cup \{\neg\varphi\}$ as our $\Delta$, the previous version of Theorem 11.6 gives us that $\Gamma \cup \{\neg\varphi\}$ is inconsistent. By Propositions 8.16, 9.16, 7.18 and 10.23, $\Gamma \vdash \varphi$. □

## 11.7 The Compactness Theorem

One important consequence of the completeness theorem is the compactness theorem. The compactness theorem states that if each *finite* subset of a set of sentences is satisfiable, the entire set is satisfiable—even if the set itself is infinite. This is far from obvious. There is nothing that seems to rule out, at first glance at least, the possibility of there being infinite sets of sentences which are contradictory, but the contradiction only arises, so to speak, from the infinite number. The compactness theorem says that such a scenario can be ruled out: there are no unsatisfiable infinite sets of sentences each finite subset of which is satisfiable. Like the completeness theorem, it has a version related to entailment: if an infinite set of sentences entails something, already a finite subset does.

**Definition 11.8.** A set $\Gamma$ of formulas is *finitely satisfiable* if and only if every finite $\Gamma_0 \subseteq \Gamma$ is satisfiable.

**Theorem 11.9** (Compactness Theorem)**.** *The following hold for any sentences $\Gamma$ and $\varphi$:*

1. *$\Gamma \vDash \varphi$ iff there is a finite $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vDash \varphi$.*

2. *$\Gamma$ is satisfiable if and only if it is finitely satisfiable.*

*Proof.* We prove (2). If $\Gamma$ is satisfiable, then there is a valuation $\mathfrak{v}$ such that $\mathfrak{v} \vDash \varphi$ for all $\varphi \in \Gamma$. Of course, this $\mathfrak{v}$ also satisfies every finite subset of $\Gamma$, so $\Gamma$ is finitely satisfiable.

Now suppose that $\Gamma$ is finitely satisfiable. Then every finite subset $\Gamma_0 \subseteq \Gamma$ is satisfiable. By soundness (Corollaries 8.24, 9.26, 7.28 and 10.32), every finite subset is consistent. Then $\Gamma$ itself must be consistent by Propositions 8.14, 9.14, 7.16 and 10.18. By completeness (Theorem 11.6), since $\Gamma$ is consistent, it is satisfiable. □

## 11.8 A Direct Proof of the Compactness Theorem

We can prove the Compactness Theorem directly, without appealing to the Completeness Theorem, using the same ideas as in the proof of the completeness theorem. In the proof of the Completeness Theorem we started with a consistent set $\Gamma$ of sentences, expanded it to a consistent and complete set $\Gamma^*$ of sentences, and then showed that in the valuation $\mathfrak{v}(\Gamma^*)$ constructed from $\Gamma^*$, all sentences of $\Gamma$ are true, so $\Gamma$ is satisfiable.

We can use the same method to show that a finitely satisfiable set of sentences is satisfiable. We just have to prove the corresponding versions of the results leading to the truth lemma where we replace "consistent" with "finitely satisfiable."

**Proposition 11.10.** *Suppose $\Gamma$ is complete and finitely satisfiable. Then:*

1. *$(\varphi \wedge \psi) \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$.*

2. *$(\varphi \vee \psi) \in \Gamma$ iff either $\varphi \in \Gamma$ or $\psi \in \Gamma$.*

3. *$(\varphi \to \psi) \in \Gamma$ iff either $\varphi \notin \Gamma$ or $\psi \in \Gamma$.*

**Lemma 11.11.** *Every finitely satisfiable set $\Gamma$ can be extended to a complete and finitely satisfiable set $\Gamma^*$.*

**Theorem 11.12** (Compactness)**.** *$\Gamma$ is satisfiable if and only if it is finitely satisfiable.*

*Proof.* If $\Gamma$ is satisfiable, then there is a valuation $\mathfrak{v}$ such that $pSat v \varphi$ for all $\varphi \in \Gamma$. Of course, this $\mathfrak{v}$ also satisfies every finite subset of $\Gamma$, so $\Gamma$ is finitely satisfiable.

Now suppose that $\Gamma$ is finitely satisfiable. By Lemma 11.11, $\Gamma$ can be extended to a complete and finitely satisfiable set $\Gamma^*$. Construct the valuation $\mathfrak{v}(\Gamma^*)$ as in Definition 11.4. The proof of the Truth Lemma (Lemma 11.5) goes through if we replace references to Proposition 11.2. □

## Problems

**Problem 11.1.** Complete the proof of Proposition 11.2.

**Problem 11.2.** Complete the proof of Lemma 11.5.

**Problem 11.3.** Use Corollary 11.7 to prove Theorem 11.6, thus showing that the two formulations of the completeness theorem are equivalent.

**Problem 11.4.** In order for a derivation system to be complete, its rules must be strong enough to prove every unsatisfiable set inconsistent. Which of the rules of derivation were necessary to prove completeness? Are any of these rules not used anywhere in the proof? In order to answer these questions, make a list or diagram that shows which of the rules of derivation were used in which results that lead up to the proof of Theorem 11.6. Be sure to note any tacit uses of rules in these proofs.

**Problem 11.5.** Prove (1) of Theorem 11.9.

**Problem 11.6.** Prove Proposition 11.10. Avoid the use of $\vdash$.

**Problem 11.7.** Prove Lemma 11.11. (Hint: the crucial step is to show that if $\Gamma_n$ is finitely satisfiable, then either $\Gamma_n \cup \{\varphi_n\}$ or $\Gamma_n \cup \{\neg\varphi_n\}$ is finitely satisfiable.)

**Problem 11.8.** Write out the complete proof of the Truth Lemma (Lemma 11.5) in the version required for the proof of Theorem 11.12.

# Part III

# First-order Logic

This part covers the metatheory of first-order logic through completeness. Currently it does not rely on a separate treatment of propositional logic; everything is proved. The source files will exclude the material on quantifiers (and replace "structure" with "valuation", $\mathfrak{M}$ with $\mathfrak{v}$, etc.) if the "FOL" tag is false. In fact, most of the material in the part on propositional logic is simply the first-order material with the "FOL" tag turned off.

If the part on propositional logic is included, this results in a lot of repetition. It is planned, however, to make it possible to let this part take into account the material on propositional logic (and exclude the material already covered, as well as shorten proofs with references to the respective places in the propositional part).

Currently four different proof systems are offered as alternatives, sequent calculus, natural deduction, signed tableaux, and axiomatic proofs. This part still needs an introduction (issue #69).

# Chapter 12

# Syntax and Semantics

## 12.1 Introduction

In order to develop the theory and metatheory of first-order logic, we must first define the syntax and semantics of its expressions. The expressions of first-order logic are terms and formulas. Terms are formed from variables, constant symbols, and function symbols. Formulas, in turn, are formed from predicate symbols together with terms (these form the smallest, "atomic" formulas), and then from atomic formulas we can form more complex ones using logical connectives and quantifiers. There are many different ways to set down the formation rules; we give just one possible one. Other systems will chose different symbols, will select different sets of connectives as primitive, will use parentheses differently (or even not at all, as in the case of so-called Polish notation). What all approaches have in common, though, is that the formation rules define the set of terms and formulas *inductively*. If done properly, every expression can result essentially in only one way according to the formation rules. The inductive definition resulting in expressions that are *uniquely readable* means we can give meanings to these expressions using the same method—inductive definition.

Giving the meaning of expressions is the domain of semantics. The central concept in semantics is that of satisfaction in a structure. A structure gives meaning to the building blocks of the language: a domain is a non-empty set of objects. The quantifiers are interpreted as ranging over this domain, constant symbols are assigned elements in the domain, function symbols are assigned functions from the domain to itself, and predicate symbols are assigned relations on the domain. The domain together with assignments to the basic vocabulary constitutes a structure. Variables may appear in formulas, and in order to give a semantics, we also have to assign elements of the domain to them—this is a variable assignment. The satisfaction relation, finally, brings these together. A formula may be satisfied in a structure $\mathfrak{M}$ relative to a variable assignment $s$, written as $\mathfrak{M}, s \vDash \varphi$. This relation is also defined by in-

duction on the structure of $\varphi$, using the truth tables for the logical connectives to define, say, satisfaction of $\varphi \wedge \psi$ in terms of satisfaction (or not) of $\varphi$ and $\psi$. It then turns out that the variable assignment is irrelevant if the formula $\varphi$ is a sentence, i.e., has no free variables, and so we can talk of sentences being simply satisfied (or not) in structures.

On the basis of the satisfaction relation $\mathfrak{M} \vDash \varphi$ for sentences we can then define the basic semantic notions of validity, entailment, and satisfiability. A sentence is valid, $\vDash \varphi$, if every structure satisfies it. It is entailed by a set of sentences, $\Gamma \vDash \varphi$, if every structure that satisfies all the sentences in $\Gamma$ also satisfies $\varphi$. And a set of sentences is satisfiable if some structure satisfies all sentences in it at the same time. Because formulas are inductively defined, and satisfaction is in turn defined by induction on the structure of formulas, we can use induction to prove properties of our semantics and to relate the semantic notions defined.

## 12.2 First-Order Languages

Expressions of first-order logic are built up from a basic vocabulary containing *variables*, *constant symbols*, *predicate symbols* and sometimes *function symbols*. From them, together with logical connectives, quantifiers, and punctuation symbols such as parentheses and commas, *terms* and *formulas* are formed.

Informally, predicate symbols are names for properties and relations, constant symbols are names for individual objects, and function symbols are names for mappings. These, except for the identity predicate $=$, are the *non-logical symbols* and together make up a language. Any first-order language $\mathcal{L}$ is determined by its non-logical symbols. In the most general case, $\mathcal{L}$ contains infinitely many symbols of each kind.

In the general case, we make use of the following symbols in first-order logic:

1. Logical symbols
   a) Logical connectives: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (conditional), $\forall$ (universal quantifier), $\exists$ (existential quantifier).
   b) The propositional constant for falsity $\bot$.
   c) The two-place identity predicate $=$.
   d) A denumerable set of variables: $v_0$, $v_1$, $v_2$, ...
2. Non-logical symbols, making up the *standard language* of first-order logic
   a) A denumerable set of $n$-place predicate symbols for each $n > 0$: $A^n_0$, $A^n_1$, $A^n_2$, ...
   b) A denumerable set of constant symbols: $c_0$, $c_1$, $c_2$, ....

c) A denumerable set of $n$-place function symbols for each $n > 0$: $f^n_0$, $f^n_1$, $f^n_2$, ...

3. Punctuation marks: (, ), and the comma.

Most of our definitions and results will be formulated for the full standard language of first-order logic. However, depending on the application, we may also restrict the language to only a few predicate symbols, constant symbols, and function symbols.

**Example 12.1.** The language $\mathcal{L}_A$ of arithmetic contains a single two-place predicate symbol $<$, a single constant symbol $0$, one one-place function symbol $'$, and two two-place function symbols $+$ and $\times$.

**Example 12.2.** The language of set theory $\mathcal{L}_Z$ contains only the single two-place predicate symbol $\in$.

**Example 12.3.** The language of orders $\mathcal{L}_\leq$ contains only the two-place predicate symbol $\leq$.

Again, these are conventions: officially, these are just aliases, e.g., $<$, $\in$, and $\leq$ are aliases for $A^2_0$, $0$ for $c_0$, $'$ for $f^1_0$, $+$ for $f^2_0$, $\times$ for $f^2_1$.

In addition to the primitive connectives and quantifiers introduced above, we also use the following *defined* symbols: $\leftrightarrow$ (biconditional), truth $\top$

A defined symbol is not officially part of the language, but is introduced as an informal abbreviation: it allows us to abbreviate formulas which would, if we only used primitive symbols, get quite long. This is obviously an advantage. The bigger advantage, however, is that proofs become shorter. If a symbol is primitive, it has to be treated separately in proofs. The more primitive symbols, therefore, the longer our proofs.

You may be familiar with different terminology and symbols than the ones we use above. Logic texts (and teachers) commonly use either $\sim$, $\neg$, and ! for "negation", $\wedge$, $\cdot$, and & for "conjunction". Commonly used symbols for the "conditional" or "implication" are $\rightarrow$, $\Rightarrow$, and $\supset$. Symbols for "biconditional," "bi-implication," or "(material) equivalence" are $\leftrightarrow$, $\Leftrightarrow$, and $\equiv$. The $\bot$ symbol is variously called "falsity," "falsum,", "absurdity,", or "bottom." The $\top$ symbol is variously called "truth," "verum,", or "top."

It is conventional to use lower case letters (e.g., $a$, $b$, $c$) from the beginning of the Latin alphabet for constant symbols (sometimes called names), and lower case letters from the end (e.g., $x$, $y$, $z$) for variables. Quantifiers combine with variables, e.g., $x$; notational variations include $\forall x$, $(\forall x)$, $(x)$, $\Pi x$, $\bigwedge_x$ for the universal quantifier and $\exists x$, $(\exists x)$, $(Ex)$, $\Sigma x$, $\bigvee_x$ for the existential quantifier.

We might treat all the propositional operators and both quantifiers as primitive symbols of the language. We might instead choose a smaller stock of primitive symbols and treat the other logical operators as defined. "Truth

functionally complete" sets of Boolean operators include $\{\neg, \vee\}$, $\{\neg, \wedge\}$, and $\{\neg, \rightarrow\}$—these can be combined with either quantifier for an expressively complete first-order language.

You may be familiar with two other logical operators: the Sheffer stroke $|$ (named after Henry Sheffer), and Peirce's arrow $\downarrow$, also known as Quine's dagger. When given their usual readings of "nand" and "nor" (respectively), these operators are truth functionally complete by themselves.

## 12.3 Terms and Formulas

Once a first-order language $\mathcal{L}$ is given, we can define expressions built up from the basic vocabulary of $\mathcal{L}$. These include in particular *terms* and *formulas*.

**Definition 12.4** (Terms)**.** The set of *terms* $\mathrm{Trm}(\mathcal{L})$ of $\mathcal{L}$ is defined inductively by:

1. Every variable is a term.
2. Every constant symbol of $\mathcal{L}$ is a term.
3. If $f$ is an $n$-place function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.
4. Nothing else is a term.

A term containing no variables is a *closed term*.

The constant symbols appear in our specification of the language and the terms as a separate category of symbols, but they could instead have been included as zero-place function symbols. We could then do without the second clause in the definition of terms. We just have to understand $f(t_1, \ldots, t_n)$ as just $f$ by itself if $n = 0$.

**Definition 12.5** (Formula)**.** The set of *formulas* $\mathrm{Frm}(\mathcal{L})$ of the language $\mathcal{L}$ is defined inductively as follows:

1. $\bot$ is an atomic formula.
2. If $R$ is an $n$-place predicate symbol of $\mathcal{L}$ and $t_1, \ldots, t_n$ are terms of $\mathcal{L}$, then $R(t_1, \ldots, t_n)$ is an atomic formula.
3. If $t_1$ and $t_2$ are terms of $\mathcal{L}$, then $=(t_1, t_2)$ is an atomic formula.
4. If $\varphi$ is a formula, then $\neg\varphi$ is formula.
5. If $\varphi$ and $\psi$ are formulas, then $(\varphi \wedge \psi)$ is a formula.
6. If $\varphi$ and $\psi$ are formulas, then $(\varphi \vee \psi)$ is a formula.

7. If $\varphi$ and $\psi$ are formulas, then $(\varphi \to \psi)$ is a formula.
8. If $\varphi$ is a formula and $x$ is a variable, then $\forall x\, \varphi$ is a formula.
9. If $\varphi$ is a formula and $x$ is a variable, then $\exists x\, \varphi$ is a formula.
10. Nothing else is a formula.

The definitions of the set of terms and that of formulas are *inductive definitions*. Essentially, we construct the set of formulas in infinitely many stages. In the initial stage, we pronounce all atomic formulas to be formulas; this corresponds to the first few cases of the definition, i.e., the cases for $\bot$, $R(t_1, \ldots, t_n)$ and $=(t_1, t_2)$. "Atomic formula" thus means any formula of this form.

The other cases of the definition give rules for constructing new formulas out of formulas already constructed. At the second stage, we can use them to construct formulas out of atomic formulas. At the third stage, we construct new formulas from the atomic formulas and those obtained in the second stage, and so on. A formula is anything that is eventually constructed at such a stage, and nothing else.

By convention, we write $=$ between its arguments and leave out the parentheses: $t_1 = t_2$ is an abbreviation for $=(t_1, t_2)$. Moreover, $\neg{=}(t_1, t_2)$ is abbreviated as $t_1 \neq t_2$. When writing a formula $(\psi * \chi)$ constructed from $\psi$, $\chi$ using a two-place connective $*$, we will often leave out the outermost pair of parentheses and write simply $\psi * \chi$.

Some logic texts require that the variable $x$ must occur in $\varphi$ in order for $\exists x\, \varphi$ and $\forall x\, \varphi$ to count as formulas. Nothing bad happens if you don't require this, and it makes things easier.

**Definition 12.6.** Formulas constructed using the defined operators are to be understood as follows:

1. $\top$ abbreviates $\neg\bot$.
2. $\varphi \leftrightarrow \psi$ abbreviates $(\varphi \to \psi) \wedge (\psi \to \varphi)$.

If we work in a language for a specific application, we will often write two-place predicate symbols and function symbols between the respective terms, e.g., $t_1 < t_2$ and $(t_1 + t_2)$ in the language of arithmetic and $t_1 \in t_2$ in the language of set theory. The successor function in the language of arithmetic is even written conventionally *after* its argument: $t'$. Officially, however, these are just conventional abbreviations for $A^2_0(t_1, t_2)$, $f^2_0(t_1, t_2)$, $A^2_0(t_1, t_2)$ and $f^1_0(t)$, respectively.

**Definition 12.7** (Syntactic identity)**.** The symbol $\equiv$ expresses syntactic identity between strings of symbols, i.e., $\varphi \equiv \psi$ iff $\varphi$ and $\psi$ are strings of symbols of the same length and which contain the same symbol in each place.

The $\equiv$ symbol may be flanked by strings obtained by concatenation, e.g., $\varphi \equiv (\psi \vee \chi)$ means: the string of symbols $\varphi$ is the same string as the one obtained by concatenating an opening parenthesis, the string $\psi$, the $\vee$ symbol, the string $\chi$, and a closing parenthesis, in this order. If this is the case, then we know that the first symbol of $\varphi$ is an opening parenthesis, $\varphi$ contains $\psi$ as a substring (starting at the second symbol), that substring is followed by $\vee$, etc.

## 12.4 Unique Readability

The way we defined formulas guarantees that every formula has a *unique reading*, i.e., there is essentially only one way of constructing it according to our formation rules for formulas and only one way of "interpreting" it. If this were not so, we would have ambiguous formulas, i.e., formulas that have more than one reading or intepretation—and that is clearly something we want to avoid. But more importantly, without this property, most of the definitions and proofs we are going to give will not go through.

Perhaps the best way to make this clear is to see what would happen if we had given bad rules for forming formulas that would not guarantee unique readability. For instance, we could have forgotten the parentheses in the formation rules for connectives, e.g., we might have allowed this:

> If $\varphi$ and $\psi$ are formulas, then so is $\varphi \to \psi$.

Starting from an atomic formula $\theta$, this would allow us to form $\theta \to \theta$. From this, together with $\theta$, we would get $\theta \to \theta \to \theta$. But there are two ways to do this:

1. We take $\theta$ to be $\varphi$ and $\theta \to \theta$ to be $\psi$.

2. We take $\varphi$ to be $\theta \to \theta$ and $\psi$ is $\theta$.

Correspondingly, there are two ways to "read" the formula $\theta \to \theta \to \theta$. It is of the form $\psi \to \chi$ where $\psi$ is $\theta$ and $\chi$ is $\theta \to \theta$, but *it is also* of the form $\psi \to \chi$ with $\psi$ being $\theta \to \theta$ and $\chi$ being $\theta$.

If this happens, our definitions will not always work. For instance, when we define the main operator of a formula, we say: in a formula of the form $\psi \to \chi$, the main operator is the indicated occurrence of $\to$. But if we can match the formula $\theta \to \theta \to \theta$ with $\psi \to \chi$ in the two different ways mentioned above, then in one case we get the first occurrence of $\to$ as the main operator, and in the second case the second occurrence. But we intend the main operator to be a *function* of the formula, i.e., every formula must have exactly one main operator occurrence.

**Lemma 12.8.** *The number of left and right parentheses in a formula $\varphi$ are equal.*

*Proof.* We prove this by induction on the way $\varphi$ is constructed. This requires two things: (a) We have to prove first that all atomic formulas have the property in question (the induction basis). (b) Then we have to prove that when we construct new formulas out of given formulas, the new formulas have the property provided the old ones do.

Let $l(\varphi)$ be the number of left parentheses, and $r(\varphi)$ the number of right parentheses in $\varphi$, and $l(t)$ and $r(t)$ similarly the number of left and right parentheses in a term $t$. We leave the proof that for any term $t$, $l(t) = r(t)$ as an exercise.

1. $\varphi \equiv \bot$: $\varphi$ has 0 left and 0 right parentheses.
2. $\varphi \equiv R(t_1, \ldots, t_n)$: $l(\varphi) = 1 + l(t_1) + \cdots + l(t_n) = 1 + r(t_1) + \cdots + r(t_n) = r(\varphi)$. Here we make use of the fact, left as an exercise, that $l(t) = r(t)$ for any term $t$.
3. $\varphi \equiv t_1 = t_2$: $l(\varphi) = l(t_1) + l(t_2) = r(t_1) + r(t_2) = r(\varphi)$.
4. $\varphi \equiv \neg\psi$: By induction hypothesis, $l(\psi) = r(\psi)$. Thus $l(\varphi) = l(\psi) = r(\psi) = r(\varphi)$.
5. $\varphi \equiv (\psi * \chi)$: By induction hypothesis, $l(\psi) = r(\psi)$ and $l(\chi) = r(\chi)$. Thus $l(\varphi) = 1 + l(\psi) + l(\chi) = 1 + r(\psi) + r(\chi) = r(\varphi)$.
6. $\varphi \equiv \forall x\, \psi$: By induction hypothesis, $l(\psi) = r(\psi)$. Thus, $l(\varphi) = l(\psi) = r(\psi) = r(\varphi)$.
7. $\varphi \equiv \exists x\, \psi$: Similarly.

□

**Definition 12.9** (Proper prefix)**.** A string of symbols $\psi$ is a *proper prefix* of a string of symbols $\varphi$ if concatenating $\psi$ and a non-empty string of symbols yields $\varphi$.

**Lemma 12.10.** *If $\varphi$ is a formula, and $\psi$ is a proper prefix of $\varphi$, then $\psi$ is not a formula.*

*Proof.* Exercise. □

**Proposition 12.11.** *If $\varphi$ is an atomic formula, then it satisfes one, and only one of the following conditions.*

1. *$\varphi \equiv \bot$.*
2. *$\varphi \equiv R(t_1, \ldots, t_n)$ where $R$ is an $n$-place predicate symbol, $t_1, \ldots, t_n$ are terms, and each of $R$, $t_1, \ldots, t_n$ is uniquely determined.*
3. *$\varphi \equiv t_1 = t_2$ where $t_1$ and $t_2$ are uniquely determined terms.*

*Proof.* Exercise. □

**Proposition 12.12** (Unique Readability)**.** *Every formula satisfies one, and only one of the following conditions.*

1. *$\varphi$ is atomic.*
2. *$\varphi$ is of the form $\neg\psi$.*
3. *$\varphi$ is of the form $(\psi \wedge \chi)$.*
4. *$\varphi$ is of the form $(\psi \vee \chi)$.*
5. *$\varphi$ is of the form $(\psi \to \chi)$.*
6. *$\varphi$ is of the form $\forall x\, \psi$.*
7. *$\varphi$ is of the form $\exists x\, \psi$.*

*Moreover, in each case $\psi$, or $\psi$ and $\chi$, are uniquely determined. This means that, e.g., there are no different pairs $\psi$, $\chi$ and $\psi'$, $\chi'$ so that $\varphi$ is both of the form $(\psi \to \chi)$ and $(\psi' \to \chi')$.*

*Proof.* The formation rules require that if a formula is not atomic, it must start with an opening parenthesis (, $\neg$, or with a quantifier. On the other hand, every formula that start with one of the following symbols must be atomic: a predicate symbol, a function symbol, a constant symbol, $\bot$.

So we really only have to show that if $\varphi$ is of the form $(\psi * \chi)$ and also of the form $(\psi' *' \chi')$, then $\psi \equiv \psi'$, $\chi \equiv \chi'$, and $* = *'$.

So suppose both $\varphi \equiv (\psi * \chi)$ and $\varphi \equiv (\psi' *' \chi')$. Then either $\psi \equiv \psi'$ or not. If it is, clearly $* = *'$ and $\chi \equiv \chi'$, since they then are substrings of $\varphi$ that begin in the same place and are of the same length. The other case is $\psi \not\equiv \psi'$. Since $\psi$ and $\psi'$ are both substrings of $\varphi$ that begin at the same place, one must be a proper prefix of the other. But this is impossible by Lemma 12.10. □

## 12.5 Main operator of a Formula

It is often useful to talk about the last operator used in constructing a formula $\varphi$. This operator is called the *main operator* of $\varphi$. Intuitively, it is the "outermost" operator of $\varphi$. For example, the main operator of $\neg\varphi$ is $\neg$, the main operator of $(\varphi \vee \psi)$ is $\vee$, etc.

**Definition 12.13** (Main operator)**.** The *main operator* of a formula $\varphi$ is defined as follows:

1. $\varphi$ is atomic: $\varphi$ has no main operator.
2. $\varphi \equiv \neg\psi$: the main operator of $\varphi$ is $\neg$.

3. $\varphi \equiv (\psi \wedge \chi)$: the main operator of $\varphi$ is $\wedge$.
4. $\varphi \equiv (\psi \vee \chi)$: the main operator of $\varphi$ is $\vee$.
5. $\varphi \equiv (\psi \to \chi)$: the main operator of $\varphi$ is $\to$.
6. $\varphi \equiv \forall x\, \psi$: the main operator of $\varphi$ is $\forall$.
7. $\varphi \equiv \exists x\, \psi$: the main operator of $\varphi$ is $\exists$.

In each case, we intend the specific indicated *occurrence* of the main operator in the formula. For instance, since the formula $((\theta \to \alpha) \to (\alpha \to \theta))$ is of the form $(\psi \to \chi)$ where $\psi$ is $(\theta \to \alpha)$ and $\chi$ is $(\alpha \to \theta)$, the second occurrence of $\to$ is the main operator.

This is a *recursive* definition of a function which maps all non-atomic formulas to their main operator occurrence. Because of the way formulas are defined inductively, every formula $\varphi$ satisfies one of the cases in Definition 12.13. This guarantees that for each non-atomic formula $\varphi$ a main operator exists. Because each formula satisfies only one of these conditions, and because the smaller formulas from which $\varphi$ is constructed are uniquely determined in each case, the main operator occurrence of $\varphi$ is unique, and so we have defined a function.

We call formulas by the following names depending on which symbol their main operator is:

| Main operator | Type of formula | Example |
|---|---|---|
| none | atomic (formula) | $\bot$, $R(t_1, \ldots, t_n)$, $t_1 = t_2$ |
| $\neg$ | negation | $\neg\varphi$ |
| $\wedge$ | conjunction | $(\varphi \wedge \psi)$ |
| $\vee$ | disjunction | $(\varphi \vee \psi)$ |
| $\to$ | conditional | $(\varphi \to \psi)$ |
| $\forall$ | universal (formula) | $\forall x\, \varphi$ |
| $\exists$ | existential (formula) | $\exists x\, \varphi$ |

## 12.6 Subformulas

It is often useful to talk about the formulas that "make up" a given formula. We call these its *subformulas*. Any formula counts as a subformula of itself; a subformula of $\varphi$ other than $\varphi$ itself is a *proper subformula*.

**Definition 12.14** (Immediate Subformula)**.** If $\varphi$ is a formula, the *immediate subformulas* of $\varphi$ are defined inductively as follows:

1. Atomic formulas have no immediate subformulas.
2. $\varphi \equiv \neg\psi$: The only immediate subformula of $\varphi$ is $\psi$.
3. $\varphi \equiv (\psi * \chi)$: The immediate subformulas of $\varphi$ are $\psi$ and $\chi$ ($*$ is any one of the two-place connectives).

4. $\varphi \equiv \forall x\, \psi$: The only immediate subformula of $\varphi$ is $\psi$.

5. $\varphi \equiv \exists x\, \psi$: The only immediate subformula of $\varphi$ is $\psi$.

**Definition 12.15** (Proper Subformula)**.** If $\varphi$ is a formula, the *proper subformulas* of $\varphi$ are recursively as follows:

1. Atomic formulas have no proper subformulas.

2. $\varphi \equiv \neg\psi$: The proper subformulas of $\varphi$ are $\psi$ together with all proper subformulas of $\psi$.

3. $\varphi \equiv (\psi * \chi)$: The proper subformulas of $\varphi$ are $\psi$, $\chi$, together with all proper subformulas of $\psi$ and those of $\chi$.

4. $\varphi \equiv \forall x\, \psi$: The proper subformulas of $\varphi$ are $\psi$ together with all proper subformulas of $\psi$.

5. $\varphi \equiv \exists x\, \psi$: The proper subformulas of $\varphi$ are $\psi$ together with all proper subformulas of $\psi$.

**Definition 12.16** (Subformula)**.** The subformulas of $\varphi$ are $\varphi$ itself together with all its proper subformulas.

Note the subtle difference in how we have defined immediate subformulas and proper subformulas. In the first case, we have directly defined the immediate subformulas of a formula $\varphi$ for each possible form of $\varphi$. It is an explicit definition by cases, and the cases mirror the inductive definition of the set of formulas. In the second case, we have also mirrored the way the set of all formulas is defined, but in each case we have also included the proper subformulas of the smaller formulas $\psi$, $\chi$ in addition to these formulas themselves. This makes the definition *recursive*. In general, a definition of a function on an inductively defined set (in our case, formulas) is recursive if the cases in the definition of the function make use of the function itself. To be well defined, we must make sure, however, that we only ever use the values of the function for arguments that come "before" the one we are defining—in our case, when defining "proper subformula" for $(\psi * \chi)$ we only use the proper subformulas of the "earlier" formulas $\psi$ and $\chi$.

## 12.7 Free Variables and Sentences

**Definition 12.17** (Free occurrences of a variable)**.** The *free* occurrences of a variable in a formula are defined inductively as follows:

1. $\varphi$ is atomic: all variable occurrences in $\varphi$ are free.

2. $\varphi \equiv \neg\psi$: the free variable occurrences of $\varphi$ are exactly those of $\psi$.

3. $\varphi \equiv (\psi * \chi)$: the free variable occurrences of $\varphi$ are those in $\psi$ together with those in $\chi$.

4. $\varphi \equiv \forall x\, \psi$: the free variable occurrences in $\varphi$ are all of those in $\psi$ except for occurrences of $x$.

5. $\varphi \equiv \exists x\, \psi$: the free variable occurrences in $\varphi$ are all of those in $\psi$ except for occurrences of $x$.

**Definition 12.18** (Bound Variables)**.** An occurrence of a variable in a formula $\varphi$ is *bound* if it is not free.

**Definition 12.19** (Scope)**.** If $\forall x\, \psi$ is an occurrence of a subformula in a formula $\varphi$, then the corresponding occurrence of $\psi$ in $\varphi$ is called the *scope* of the corresponding occurrence of $\forall x$. Similarly for $\exists x$.

If $\psi$ is the scope of a quantifier occurrence $\forall x$ or $\exists x$ in $\varphi$, then all occurrences of $x$ which are free in $\psi$ are said to be *bound by* the mentioned quantifier occurrence.

**Example 12.20.** Consider the following formula:

$$\exists v_0\ \underbrace{A_0^2(v_0, v_1)}_{\psi}$$

$\psi$ represents the scope of $\exists v_0$. The quantifier binds the occurence of $v_0$ in $\psi$, but does not bind the occurence of $v_1$. So $v_1$ is a free variable in this case.

We can now see how this might work in a more complicated formula $\varphi$:

$$\forall v_0\ \underbrace{(A_0^1(v_0) \to A_0^2(v_0, v_1))}_{\psi} \to \exists v_1\ \underbrace{(A_1^2(v_0, v_1) \vee \forall v_0\ \overbrace{\neg A_1^1(v_0)}^{\theta})}_{\chi}$$

$\psi$ is the scope of the first $\forall v_0$, $\chi$ is the scope of $\exists v_1$, and $\theta$ is the scope of the second $\forall v_0$. The first $\forall v_0$ binds the occurrences of $v_0$ in $\psi$, $\exists v_1$ the occurrence of $v_1$ in $\chi$, and the second $\forall v_0$ binds the occurrence of $v_0$ in $\theta$. The first occurrence of $v_1$ and the fourth occurrence of $v_0$ are free in $\varphi$. The last occurrence of $v_0$ is free in $\theta$, but bound in $\chi$ and $\varphi$.

**Definition 12.21** (Sentence)**.** A formula $\varphi$ is a *sentence* iff it contains no free occurrences of variables.

## 12.8 Substitution

**Definition 12.22** (Substitution in a term)**.** We define $s[t/x]$, the result of *substituting* $t$ for every occurrence of $x$ in $s$, recursively:

1. $s \equiv c$: $s[t/x]$ is just $s$.

2. $s \equiv y$: $s[t/x]$ is also just $s$, provided $y$ is a variable and $y \not\equiv x$.

3. $s \equiv x$: $s[t/x]$ is $t$.

4. $s \equiv f(t_1, \ldots, t_n)$: $s[t/x]$ is $f(t_1[t/x], \ldots, t_n[t/x])$.

**Definition 12.23.** A term $t$ is *free for* $x$ in $\varphi$ if none of the free occurrences of $x$ in $\varphi$ occur in the scope of a quantifier that binds a variable in $t$.

**Example 12.24.**

1. $v_8$ is free for $v_1$ in $\exists v_3 A_4^2(v_3, v_1)$

2. $f_1^2(v_1, v_2)$ is *not* free for $v_o$ in $\forall v_2 A_4^2(v_0, v_2)$

**Definition 12.25** (Substitution in a formula)**.** If $\varphi$ is a formula, $x$ is a variable, and $t$ is a term free for $x$ in $\varphi$, then $\varphi[t/x]$ is the result of substituting $t$ for all free occurrences of $x$ in $\varphi$.

1. $\varphi \equiv \bot$: $\varphi[t/x]$ is $\bot$.

2. $\varphi \equiv P(t_1, \ldots, t_n)$: $\varphi[t/x]$ is $P(t_1[t/x], \ldots, t_n[t/x])$.

3. $\varphi \equiv t_1 = t_2$: $\varphi[t/x]$ is $t_1[t/x] = t_2[t/x]$.

4. $\varphi \equiv \neg\psi$: $\varphi[t/x]$ is $\neg\psi[t/x]$.

5. $\varphi \equiv (\psi \wedge \chi)$: $\varphi[t/x]$ is $(\psi[t/x] \wedge \chi[t/x])$.

6. $\varphi \equiv (\psi \vee \chi)$: $\varphi[t/x]$ is $(\psi[t/x] \vee \chi[t/x])$.

7. $\varphi \equiv (\psi \to \chi)$: $\varphi[t/x]$ is $(\psi[t/x] \to \chi[t/x])$.

8. $\varphi \equiv \forall y\, \psi$: $\varphi[t/x]$ is $\forall y\, \psi[t/x]$, provided $y$ is a variable other than $x$; otherwise $\varphi[t/x]$ is just $\varphi$.

9. $\varphi \equiv \exists y\, \psi$: $\varphi[t/x]$ is $\exists y\, \psi[t/x]$, provided $y$ is a variable other than $x$; otherwise $\varphi[t/x]$ is just $\varphi$.

Note that substitution may be vacuous: If $x$ does not occur in $\varphi$ at all, then $\varphi[t/x]$ is just $\varphi$.

The restriction that $t$ must be free for $x$ in $\varphi$ is necessary to exclude cases like the following. If $\varphi \equiv \exists y\, x < y$ and $t \equiv y$, then $\varphi[t/x]$ would be $\exists y\, y < y$. In this case the free variable $y$ is "captured" by the quantifier $\exists y$ upon substitution, and that is undesirable. For instance, we would like it to be the case that whenever $\forall x\, \psi$ holds, so does $\psi[t/x]$. But consider $\forall x\, \exists y\, x < y$ (here $\psi$ is $\exists y\, x < y$). It is sentence that is true about, e.g., the natural numbers: for every number $x$ there is a number $y$ greater than it. If we allowed $y$ as a possible substitution for $x$, we would end up with $\psi[y/x] \equiv \exists y\, y < y$, which

is false. We prevent this by requiring that none of the free variables in $t$ would end up being bound by a quantifier in $\varphi$.

We often use the following convention to avoid cumbersume notation: If $\varphi$ is a formula with a free variable $x$, we write $\varphi(x)$ to indicate this. When it is clear which $\varphi$ and $x$ we have in mind, and $t$ is a term (assumed to be free for $x$ in $\varphi(x)$), then we write $\varphi(t)$ as short for $\varphi(x)[t/x]$.

## 12.9 Structures for First-order Languages

First-order languages are, by themselves, *uninterpreted:* the constant symbols, function symbols, and predicate symbols have no specific meaning attached to them. Meanings are given by specifying a *structure*. It specifies the *domain*, i.e., the objects which the constant symbols pick out, the function symbols operate on, and the quantifiers range over. In addition, it specifies which constant symbols pick out which objects, how a function symbol maps objects to objects, and which objects the predicate symbols apply to. Structures are the basis for *semantic* notions in logic, e.g., the notion of consequence, validity, satisfiablity. They are variously called "structures," "interpretations," or "models" in the literature.

**Definition 12.26** (Structures)**.** A *structure* $\mathfrak{M}$, for a language $\mathcal{L}$ of first-order logic consists of the following elements:

1. *Domain:* a non-empty set, $|\mathfrak{M}|$
2. *Interpretation of constant symbols:* for each constant symbol $c$ of $\mathcal{L}$, an element $c^{\mathfrak{M}} \in |\mathfrak{M}|$
3. *Interpretation of predicate symbols:* for each $n$-place predicate symbol $R$ of $\mathcal{L}$ (other than $=$), an $n$-place relation $R^{\mathfrak{M}} \subseteq |\mathfrak{M}|^n$
4. *Interpretation of function symbols:* for each $n$-place function symbol $f$ of $\mathcal{L}$, an $n$-place function $f^{\mathfrak{M}} \colon |\mathfrak{M}|^n \to |\mathfrak{M}|$

**Example 12.27.** A structure $\mathfrak{M}$ for the language of arithmetic consists of a set, an element of $|\mathfrak{M}|$, $\mathsf{0}^{\mathfrak{M}}$, as interpretation of the constant symbol $\mathsf{0}$, a one-place function $\mathsf{'}^{\mathfrak{M}} \colon |\mathfrak{M}| \to |\mathfrak{M}|$, two two-place functions $+^{\mathfrak{M}}$ and $\times^{\mathfrak{M}}$, both $|\mathfrak{M}|^2 \to |\mathfrak{M}|$, and a two-place relation $<^{\mathfrak{M}} \subseteq |\mathfrak{M}|^2$.

An obvious example of such a structure is the following:

1. $|\mathfrak{N}| = \mathbb{N}$
2. $\mathsf{0}^{\mathfrak{N}} = 0$
3. $\mathsf{'}^{\mathfrak{N}}(n) = n + 1$ for all $n \in \mathbb{N}$
4. $+^{\mathfrak{N}}(n, m) = n + m$ for all $n, m \in \mathbb{N}$

5. $\times^{\mathfrak{N}}(n, m) = n \cdot m$ for all $n, m \in \mathbb{N}$

6. $<^{\mathfrak{N}} = \{\langle n, m\rangle : n \in \mathbb{N}, m \in \mathbb{N}, n < m\}$

The structure $\mathfrak{N}$ for $\mathcal{L}_A$ so defined is called the *standard model of arithmetic*, because it interprets the non-logical constants of $\mathcal{L}_A$ exactly how you would expect.

However, there are many other possible structures for $\mathcal{L}_A$. For instance, we might take as the domain the set $\mathbb{Z}$ of integers instead of $\mathbb{N}$, and define the interpretations of $0$, $'$, $+$, $\times$, $<$ accordingly. But we can also define structures for $\mathcal{L}_A$ which have nothing even remotely to do with numbers.

**Example 12.28.** A structure $\mathfrak{M}$ for the language $\mathcal{L}_Z$ of set theory requires just a set and a single-two place relation. So technically, e.g., the set of people plus the relation "$x$ is older than $y$" could be used as a structure for $\mathcal{L}_Z$, as well as $\mathbb{N}$ together with $n \geq m$ for $n, m \in \mathbb{N}$.

A particularly interesting structure for $\mathcal{L}_Z$ in which the elements of the domain are actually sets, and the interpretation of $\in$ actually is the relation "$x$ is an element of $y$" is the structure $\mathfrak{HF}$ of *hereditarily finite sets*:

1. $|\mathfrak{HF}| = \emptyset \cup \wp(\emptyset) \cup \wp(\wp(\emptyset)) \cup \wp(\wp(\wp(\emptyset))) \cup \ldots$;

2. $\in^{\mathfrak{HF}} = \{\langle x, y\rangle : x, y \in |\mathfrak{HF}|, x \in y\}$.

The stipulations we make as to what counts as a structure impact our logic. For example, the choice to prevent empty domains ensures, given the usual account of satisfaction (or truth) for quantified sentences, that $\exists x\, (\varphi(x) \vee \neg\varphi(x))$ is valid—that is, a logical truth. And the stipulation that all constant symbols must refer to an object in the domain ensures that the existential generalization is a sound pattern of inference: $\varphi(a)$, therefore $\exists x\, \varphi(x)$. If we allowed names to refer outside the domain, or to not refer, then we would be on our way to a *free logic*, in which existential generalization requires an additional premise: $\varphi(a)$ and $\exists x\, x = a$, therefore $\exists x\, \varphi(x)$.

## 12.10 Covered Structures for First-order Languages

Recall that a term is *closed* if it contains no variables.

**Definition 12.29** (Value of closed terms)**.** If $t$ is a closed term of the language $\mathcal{L}$ and $\mathfrak{M}$ is a structure for $\mathcal{L}$, the *value* $\mathrm{Val}^{\mathfrak{M}}(t)$ is defined as follows:

1. If $t$ is just the constant symbol $c$, then $\mathrm{Val}^{\mathfrak{M}}(c) = c^{\mathfrak{M}}$.

2. If $t$ is of the form $f(t_1, \ldots, t_n)$, then

$$\mathrm{Val}^{\mathfrak{M}}(t) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}(t_n)).$$

**Definition 12.30** (Covered structure)**.** A structure is *covered* if every element of the domain is the value of some closed term.

**Example 12.31.** Let $\mathcal{L}$ be the language with constant symbols $\mathit{zero}$, $\mathit{one}$, $\mathit{two}$, ..., the binary predicate symbol $<$, and the binary function symbols $+$ and $\times$. Then a structure $\mathfrak{M}$ for $\mathcal{L}$ is the one with domain $|\mathfrak{M}| = \{0, 1, 2, \ldots\}$ and assignments $\mathit{zero}^{\mathfrak{M}} = 0$, $\mathit{one}^{\mathfrak{M}} = 1$, $\mathit{two}^{\mathfrak{M}} = 2$, and so forth. For the binary relation symbol $<$, the set $<^{\mathfrak{M}}$ is the set of all pairs $\langle c_1, c_2\rangle \in |\mathfrak{M}|^2$ such that $c_1$ is less than $c_2$: for example, $\langle 1, 3\rangle \in <^{\mathfrak{M}}$ but $\langle 2, 2\rangle \notin <^{\mathfrak{M}}$. For the binary function symbol $+$, define $+^{\mathfrak{M}}$ in the usual way—for example, $+^{\mathfrak{M}}(2, 3)$ maps to 5, and similarly for the binary function symbol $\times$. Hence, the value of $\mathit{four}$ is just 4, and the value of $\times(\mathit{two}, +(\mathit{three}, \mathit{zero}))$ (or in infix notation, $\mathit{two} \times (\mathit{three} + \mathit{zero})$ ) is

$$\begin{aligned}
\mathrm{Val}^{\mathfrak{M}}(\times(\mathit{two}, +(\mathit{three}, \mathit{zero})) &= \\
&= \times^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\mathit{two}), \mathrm{Val}^{\mathfrak{M}}(\mathit{two}, +(\mathit{three}, \mathit{zero}))) \\
&= \times^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\mathit{two}), +^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\mathit{three}), \mathrm{Val}^{\mathfrak{M}}(\mathit{zero}))) \\
&= \times^{\mathfrak{M}}(\mathit{two}^{\mathfrak{M}}, +^{\mathfrak{M}}(\mathit{three}^{\mathfrak{M}}, \mathit{zero}^{\mathfrak{M}})) \\
&= \times^{\mathfrak{M}}(2, +^{\mathfrak{M}}(3, 0)) \\
&= \times^{\mathfrak{M}}(2, 3) \\
&= 6
\end{aligned}$$

## 12.11 Satisfaction of a Formula in a Structure

The basic notion that relates expressions such as terms and formulas, on the one hand, and structures on the other, are those of *value* of a term and *satisfaction* of a formula. Informally, the value of a term is an element of a structure—if the term is just a constant, its value is the object assigned to the constant by the structure, and if it is built up using function symbols, the value is computed from the values of constants and the functions assigned to the functions in the term. A formula is *satisfied* in a structure if the interpretation given to the predicates makes the formula true in the domain of the structure. This notion of satisfaction is specified inductively: the specification of the structure directly states when atomic formulas are satisfied, and we define when a complex formula is satisfied depending on the main connective or quantifier and whether or not the immediate subformulas are satisfied. The case of the quantifiers here is a bit tricky, as the immediate subformula of a quantified formula has a free variable, and structures don't specify the values of variables. In order to deal with this difficulty, we also introduce *variable assignments* and define satisfaction not with respect to a structure alone, but with respect to a structure plus a variable assignment.

**Definition 12.32** (Variable Assignment)**.** A *variable assignment* $s$ for a structure $\mathfrak{M}$ is a function which maps each variable to an element of $|\mathfrak{M}|$, i.e., $s\colon \mathrm{Var} \to |\mathfrak{M}|$.

A structure assigns a value to each constant symbol, and a variable assignment to each variable. But we want to use terms built up from them to also name elements of the domain. For this we define the value of terms inductively. For constant symbols and variables the value is just as the structure or the variable assignment specifies it; for more complex terms it is computed recursively using the functions the structure assigns to the function symbols.

**Definition 12.33** (Value of Terms)**.** If $t$ is a term of the language $\mathcal{L}$, $\mathfrak{M}$ is a structure for $\mathcal{L}$, and $s$ is a variable assignment for $\mathfrak{M}$, the *value* $\mathrm{Val}^{\mathfrak{M}}_s(t)$ is defined as follows:

1. $t \equiv c$: $\mathrm{Val}^{\mathfrak{M}}_s(t) = c^{\mathfrak{M}}$.
2. $t \equiv x$: $\mathrm{Val}^{\mathfrak{M}}_s(t) = s(x)$.
3. $t \equiv f(t_1, \ldots, t_n)$:
$$\mathrm{Val}^{\mathfrak{M}}_s(t) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_s(t_n)).$$

**Definition 12.34** ($x$-Variant)**.** If $s$ is a variable assignment for a structure $\mathfrak{M}$, then any variable assignment $s'$ for $\mathfrak{M}$ which differs from $s$ at most in what it assigns to $x$ is called an *$x$-variant* of $s$. If $s'$ is an $x$-variant of $s$ we write $s \sim_x s'$.

Note that an $x$-variant of an assignment $s$ does not *have* to assign something different to $x$. In fact, every assignment counts as an $x$-variant of itself.

**Definition 12.35** (Satisfaction)**.** Satisfaction of a formula $\varphi$ in a structure $\mathfrak{M}$ relative to a variable assignment $s$, in symbols: $\mathfrak{M}, s \vDash \varphi$, is defined recursively as follows. (We write $\mathfrak{M}, s \nvDash \varphi$ to mean "not $\mathfrak{M}, s \vDash \varphi$.")

1. $\varphi \equiv \bot$: $\mathfrak{M}, s \nvDash \varphi$.
2. $\varphi \equiv R(t_1, \ldots, t_n)$: $\mathfrak{M}, s \vDash \varphi$ iff $\langle \mathrm{Val}^{\mathfrak{M}}_s(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_s(t_n)\rangle \in R^{\mathfrak{M}}$.
3. $\varphi \equiv t_1 = t_2$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathrm{Val}^{\mathfrak{M}}_s(t_1) = \mathrm{Val}^{\mathfrak{M}}_s(t_2)$.
4. $\varphi \equiv \neg\psi$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \nvDash \psi$.
5. $\varphi \equiv (\psi \wedge \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \vDash \psi$ and $\mathfrak{M}, s \vDash \chi$.
6. $\varphi \equiv (\psi \vee \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \vDash \varphi$ or $\mathfrak{M}, s \vDash \psi$ (or both).
7. $\varphi \equiv (\psi \to \chi)$: $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s \nvDash \psi$ or $\mathfrak{M}, s \vDash \chi$ (or both).
8. $\varphi \equiv \forall x\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for every $x$-variant $s'$ of $s$, $\mathfrak{M}, s' \vDash \psi$.

9. $\varphi \equiv \exists x\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff there is an $x$-variant $s'$ of $s$ so that $\mathfrak{M}, s' \vDash \psi$.

The variable assignments are important in the last two clauses. We cannot define satisfaction of $\forall x\, \psi(x)$ by "for all $a \in |\mathfrak{M}|$, $\mathfrak{M} \vDash \psi(a)$." We cannot define satisfaction of $\exists x\, \psi(x)$ by "for at least one $a \in |\mathfrak{M}|$, $\mathfrak{M} \vDash \psi(a)$." The reason is that $a$ is not symbol of the language, and so $\psi(a)$ is not a formula (that is, $\psi[a/x]$ is undefined). We also cannot assume that we have constant symbols or terms available that name every element of $\mathfrak{M}$, since there is nothing in the definition of structures that requires it. Even in the standard language the set of constant symbols is denumerable, so if $|\mathfrak{M}|$ is not enumerable there aren't even enough constant symbols to name every object.

**Example 12.36.** Let $=\{a, b, f, R\}$ where $a$ and $b$ are constant symbols, $f$ is a two-place function symbol, and $R$ is a two-place predicate symbol. Consider the structure $\mathfrak{M}$ defined by:

1. $|\mathfrak{M}| = \{1, 2, 3, 4\}$
2. $a^{\mathfrak{M}} = 1$
3. $b^{\mathfrak{M}} = 2$
4. $f^{\mathfrak{M}}(x, y) = x + y$ if $x + y \leq 3$ and $= 3$ otherwise.
5. $R^{\mathfrak{M}} = \{\langle 1, 1\rangle, \langle 1, 2\rangle, \langle 2, 3\rangle, \langle 2, 4\rangle\}$

The function $s(x) = 1$ that assigns $1 \in |\mathfrak{M}|$ to every variable is a variable assignment for $\mathfrak{M}$.

Then

$$\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(a), \mathrm{Val}^{\mathfrak{M}}_s(b)).$$

Since $a$ and $b$ are constant symbols, $\mathrm{Val}^{\mathfrak{M}}_s(a) = a^{\mathfrak{M}} = 1$ and $\mathrm{Val}^{\mathfrak{M}}_s(b) = b^{\mathfrak{M}} = 2$. So

$$\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)) = f^{\mathfrak{M}}(1, 2) = 1 + 2 = 3.$$

To compute the value of $f(f(a, b), a)$ we have to consider

$$\mathrm{Val}^{\mathfrak{M}}_s(f(f(a, b), a)) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)), \mathrm{Val}^{\mathfrak{M}}_s(a)) = f^{\mathfrak{M}}(3, 1) = 3,$$

since $3 + 1 > 3$. Since $s(x) = 1$ and $\mathrm{Val}^{\mathfrak{M}}_s(x) = s(x)$, we also have

$$\mathrm{Val}^{\mathfrak{M}}_s(f(f(a, b), x)) = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(f(a, b)), \mathrm{Val}^{\mathfrak{M}}_s(x)) = f^{\mathfrak{M}}(3, 1) = 3,$$

An atomic formula $R(t_1, t_2)$ is satisfied if the tuple of values of its arguments, i.e., $\langle \mathrm{Val}^{\mathfrak{M}}_s(t_1), \mathrm{Val}^{\mathfrak{M}}_s(t_2)\rangle$, is an element of $R^{\mathfrak{M}}$. So, e.g., we have

$\mathfrak{M}, s \vDash R(b, f(a,b))$ since $\langle \mathrm{Val}^{\mathfrak{M}}(b), \mathrm{Val}^{\mathfrak{M}}(f(a,b))\rangle = \langle 2,3\rangle \in R^{\mathfrak{M}}$, but $\mathfrak{M}, s \nvDash R(x, f(a,b))$ since $\langle 1,3\rangle \notin R^{\mathfrak{M}}[s]$.

To determine if a non-atomic formula $\varphi$ is satisfied, you apply the clauses in the inductive definition that applies to the main connective. For instance, the main connective in $R(a,a) \to (R(b,x) \lor R(x,b)$ is the $\to$, and

$$\begin{aligned}&\mathfrak{M}, s \vDash R(a,a) \to (R(b,x) \lor R(x,b)) \text{ iff}\\ &\quad \mathfrak{M}, s \nvDash R(a,a) \text{ or } \mathfrak{M}, s \vDash R(b,x) \lor R(x,b)\end{aligned}$$

Since $\mathfrak{M}, s \vDash R(a,a)$ (because $\langle 1,1\rangle \in R^{\mathfrak{M}}$) we can't yet determine the answer and must first figure out if $\mathfrak{M}, s \vDash R(b,x) \lor R(x,b)$:

$$\begin{aligned}&\mathfrak{M}, s \vDash R(b,x) \lor R(x,b) \text{ iff}\\ &\quad \mathfrak{M}, s \vDash R(b,x) \text{ or } \mathfrak{M}, s \vDash R(x,b)\end{aligned}$$

And this is the case, since $\mathfrak{M}, s \vDash R(x,b)$ (because $\langle 1,2\rangle \in R^{\mathfrak{M}}$).

Recall that an $x$-variant of $s$ is a variable assignment that differs from $s$ at most in what it assigns to $x$. For every element of $|\mathfrak{M}|$, there is an $x$-variant of $s$: $s_1(x) = 1$, $s_2(x) = 2$, $s_3(x) = 3$, $s_4(x) = 4$, and with $s_i(y) = s(y) = 1$ for all variables $y$ other than $x$. These are all the $x$-variants of $s$ for the structure $\mathfrak{M}$, since $|\mathfrak{M}| = \{1,2,3,4\}$. Note, in particular, that $s_1 = s$ is also an $x$-variant of $s$, i.e., $s$ is always an $x$-variant of itself.

To determine if an existentially quantified formula $\exists x\, \varphi(x)$ is satisfied, we have to determine if $\mathfrak{M}, s' \vDash \varphi(x)$ for at least one $x$-variant $s'$ of $s$. So,

$$\mathfrak{M}, s \vDash \exists x\, (R(b,x) \lor R(x,b)),$$

since $\mathfrak{M}, s_1 \vDash R(b,x) \lor R(x,b)$ ($s_3$ would also fit the bill). But,

$$\mathfrak{M}, s \nvDash \exists x\, (R(b,x) \land R(x,b))$$

since for none of the $s_i$, $\mathfrak{M}, s_i \vDash R(b,x) \land R(x,b)$.

To determine if a universally quantified formula $\forall x\, \varphi(x)$ is satisfied, we have to determine if $\mathfrak{M}, s' \vDash \varphi(x)$ for all $x$-variants $s'$ of $s$. So,

$$\mathfrak{M}, s \vDash \forall x\, (R(x,a) \to R(a,x)),$$

since $\mathfrak{M}, s_i \vDash R(x,a) \to R(a,x)$ for all $s_i$ ($\mathfrak{M}, s_1 \vDash R(a,x)$ and $\mathfrak{M}, s_j \nvDash R(x,a)$ for $j = 2$, 3, and 4). But,

$$\mathfrak{M}, s \nvDash \forall x\, (R(a,x) \to R(x,a))$$

since $\mathfrak{M}, s_2 \nvDash R(a,x) \to R(x,a)$ (because $\mathfrak{M}, s_2 \vDash R(a,x)$ and $\mathfrak{M}, s_2 \nvDash R(x,a)$).

For a more complicated case, consider

$$\forall x\,(R(a,x) \to \exists y\,R(x,y)).$$

Since $\mathfrak{M}, s_3 \not\models R(a,x)$ and $\mathfrak{M}, s_4 \not\models R(a,x)$, the interesting cases where we have to worry about the consequent of the conditional are only $s_1$ and $s_2$. Does $\mathfrak{M}, s_1 \models \exists y\,R(x,y)$ hold? It does if there is at least one $y$-variant $s_1'$ of $s_1$ so that $\mathfrak{M}, s_1' \models R(x,y)$. In fact, $s_1$ is such a $y$-variant ($s_1(x) = 1$, $s_1(y) = 1$, and $\langle 1,1\rangle \in R^{\mathfrak{M}}$), so the answer is yes. To determine if $\mathfrak{M}, s_2 \models \exists y\,R(x,y)$ we have to look at the $y$-variants of $s_2$. Here, $s_2$ itself does not satisfy $R(x,y)$ ($s_2(x) = 2$, $s_2(y) = 1$, and $\langle 2,1\rangle \notin R^{\mathfrak{M}}$). However, consider $s_2' \sim_y s_2$ with $s_2'(y) = 3$. $\mathfrak{M}, s_2' \models R(x,y)$ since $\langle 2,3\rangle \in R^{\mathfrak{M}}$, and so $\mathfrak{M}, s_2 \models \exists y\,R(x,y)$. In sum, for every $x$-variant $s_i$ of $s$, either $\mathfrak{M}, s_i \not\models R(a,x)$ ($i = 3, 4$) or $\mathfrak{M}, s_i \models \exists y\,R(x,y)$ ($i = 1, 2$), and so

$$\mathfrak{M}, s \models \forall x\,(R(a,x) \to \exists y\,R(x,y)).$$

On the other hand,

$$\mathfrak{M}, s \not\models \exists x\,(R(a,x) \land \forall y\,R(x,y)).$$

The only $x$-variants $s_i$ of $s$ with $\mathfrak{M}, s_i \models R(a,x)$ are $s_1$ and $s_2$. But for each, there is in turn a $y$-variant $s_i' \sim_y s_i$ with $s_i'(y) = 4$ so that $\mathfrak{M}, s_i' \not\models R(x,y)$ and so $\mathfrak{M}, s_i \not\models \forall y\,R(x,y)$ for $i = 1, 2$. In sum, none of the $x$-variants $s_i \sim_x s$ are such that $\mathfrak{M}, s_i \models R(a,x) \land \forall y\,R(x,y)$.

## 12.12 Variable Assignments

A variable assignment $s$ provides a value for *every* variable—and there are infinitely many of them. This is of course not necessary. We require variable assignments to assign values to all variables simply because it makes things a lot easier. The value of a term $t$, and whether or not a formula $\varphi$ is satisfied in a structure with respect to $s$, only depend on the assignments $s$ makes to the variables in $t$ and the free variables of $\varphi$. This is the content of the next two propositions. To make the idea of "depends on" precise, we show that any two variable assignments that agree on all the variables in $t$ give the same value, and that $\varphi$ is satisfied relative to one iff it is satisfied relative to the other if two variable assignments agree on all free variables of $\varphi$.

**Proposition 12.37.** *If the variables in a term $t$ are among $x_1, \ldots, x_n$, and $s_1(x_i) = s_2(x_i)$ for $i = 1, \ldots, n$, then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$.*

*Proof.* By induction on the complexity of $t$. For the base case, $t$ can be a constant symbol or one one of the variables $x_1, \ldots, x_n$. If $t = c$, then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = c^{\mathfrak{M}} = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$. If $t = x_i$, $s_1(x_i) = s_2(x_i)$ by the hypothesis of the proposition, and so $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) = s_1(x_i) = s_2(x_i) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t)$.

For the inductive step, assume that $t = f(t_1, \ldots, t_k)$ and that the claim holds for $t_1, \ldots, t_k$. Then

$$\begin{aligned}\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(f(t_1, \ldots, t_k)) = \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_k))\end{aligned}$$

For $j = 1, \ldots, k$, the variables of $t_j$ are among $x_1, \ldots, x_n$. So by induction hypothesis, $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_j) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_j)$. So,

$$\begin{aligned}\mathrm{Val}^{\mathfrak{M}}_{s_1}(t) &= \mathrm{Val}^{\mathfrak{M}}_{s_2}(f(t_1, \ldots, t_k)) = \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_k)) = \\ &= f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s_2}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_k)) = \\ &= \mathrm{Val}^{\mathfrak{M}}_{s_2}(f(t_1, \ldots, t_k)) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t).\end{aligned}$$

□

**Proposition 12.38.** *If the free variables in $\varphi$ are among $x_1, \ldots, x_n$, and $s_1(x_i) = s_2(x_i)$ for $i = 1, \ldots, n$, then $\mathfrak{M}, s_1 \vDash \varphi$ iff $\mathfrak{M}, s_2 \vDash \varphi$.*

*Proof.* We use induction on the complexity of $\varphi$. For the base case, where $\varphi$ is atomic, $\varphi$ can be: $\bot$, $R(t_1, \ldots, t_k)$ for a $k$-place predicate $R$ and terms $t_1, \ldots, t_k$, or $t_1 = t_2$ for terms $t_1$ and $t_2$.

1. $\varphi \equiv \bot$: both $\mathfrak{M}, s_1 \nvDash \varphi$ and $\mathfrak{M}, s_2 \nvDash \varphi$.

2. $\varphi \equiv R(t_1, \ldots, t_k)$: let $\mathfrak{M}, s_1 \vDash \varphi$. Then

   $$\langle \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_k) \rangle \in R^{\mathfrak{M}}.$$

   For $i = 1, \ldots, k$, $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_i) = \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_i)$ by Proposition 12.37. So we also have $\langle \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_i), \ldots, \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_k) \rangle \in R^{\mathfrak{M}}$.

3. $\varphi \equiv t_1 = t_2$: suppose $\mathfrak{M}, s_1 \vDash \varphi$. Then $\mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1) = \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_2)$. So,

   $$\begin{aligned}\mathrm{Val}^{\mathfrak{M}}_{s_2}(t_1) &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_1) && \text{(by Proposition 12.37)} \\ &= \mathrm{Val}^{\mathfrak{M}}_{s_1}(t_2) && \text{(since } \mathfrak{M}, s_1 \vDash t_1 = t_2\text{)} \\ &= \mathrm{Val}^{\mathfrak{M}}_{s_2}(t_2) && \text{(by Proposition 12.37),}\end{aligned}$$

   so $\mathfrak{M}, s_2 \vDash t_1 = t_2$.

Now assume $\mathfrak{M}, s_1 \vDash \psi$ iff $\mathfrak{M}, s_2 \vDash \psi$ for all formulas $\psi$ less complex than $\varphi$. The induction step proceeds by cases determined by the main operator of $\varphi$. In each case, we only demonstrate the forward direction of the biconditional; the proof of the reverse direction is symmetrical. In all cases except those for

the quantifiers, we apply the induction hypothesis to sub-formulas $\psi$ of $\varphi$. The free variables of $\psi$ are among those of $\varphi$. Thus, if $s_1$ and $s_2$ agree on the free variables of $\varphi$, they also agree on those of $\psi$, and the induction hypothesis applies to $\psi$.

1. $\varphi \equiv \neg\psi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then $\mathfrak{M}, s_1 \nvDash \psi$, so by the induction hypothesis, $\mathfrak{M}, s_2 \nvDash \psi$, hence $\mathfrak{M}, s_2 \vDash \varphi$.

2. $\varphi \equiv \psi \wedge \chi$: exercise.

3. $\varphi \equiv \psi \vee \chi$: if $\mathfrak{M}, s_1 \vDash \varphi$, then $\mathfrak{M}, s_1 \vDash \psi$ or $\mathfrak{M}, s_1 \vDash \chi$. By induction hypothesis, $\mathfrak{M}, s_2 \vDash \psi$ or $\mathfrak{M}, s_2 \vDash \chi$, so $\mathfrak{M}, s_2 \vDash \varphi$.

4. $\varphi \equiv \psi \to \chi$: exercise.

5. $\varphi \equiv \exists x\, \psi$: if $\mathfrak{M}, s_1 \vDash \varphi$, there is an $x$-variant $s_1'$ of $s_1$ so that $\mathfrak{M}, s_1' \vDash \psi$. Let $s_2'$ be the $x$-variant of $s_2$ that assigns the same thing to $x$ as does $s_1'$. The free variables of $\psi$ are among $x_1, \ldots, x_n$, and $x$. $s_1'(x_i) = s_2'(x_i)$, since $s_1'$ and $s_2'$ are $x$-variants of $s_1$ and $s_2$, respectively, and by hypothesis $s_1(x_i) = s_2(x_i)$. $s_1'(x) = s_2'(x)$ by the way we have defined $s_2'$. Then the induction hypothesis applies to $\psi$ and $s_1'$, $s_2'$, so $\mathfrak{M}, s_2' \vDash \psi$. Hence, there is an $x$-variant of $s_2$ that satisfies $\psi$, and so $\mathfrak{M}, s_2 \vDash \varphi$.

6. $\varphi \equiv \forall x\, \psi$: exercise.

By induction, we get that $\mathfrak{M}, s_1 \vDash \varphi$ iff $\mathfrak{M}, s_2 \vDash \varphi$ whenever the free variables in $\varphi$ are among $x_1, \ldots, x_n$ and $s_1(x_i) = s_2(x_i)$ for $i = 1, \ldots, n$. □

Sentences have no free variables, so any two variable assignments assign the same things to all the (zero) free variables of any sentence. The proposition just proved then means that whether or not a sentence is satisfied in a structure relative to a variable assignment is completely independent of the assignment. We'll record this fact. It justifies the definition of satisfaction of a sentence in a structure (without mentioning a variable assignment) that follows.

**Corollary 12.39.** *If $\varphi$ is a sentence and $s$ a variable assignment, then $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s' \vDash \varphi$ for every variable assignment $s'$.*

*Proof.* Let $s'$ be any variable assignment. Since $\varphi$ is a sentence, it has no free variables, and so every variable assignment $s'$ trivially assigns the same things to all free variables of $\varphi$ as does $s$. So the condition of Proposition 12.38 is satisfied, and we have $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}, s' \vDash \varphi$. □

**Definition 12.40.** If $\varphi$ is a sentence, we say that a structure $\mathfrak{M}$ *satisfies* $\varphi$, $\mathfrak{M} \vDash \varphi$, iff $\mathfrak{M}, s \vDash \varphi$ for all variable assignments $s$.

If $\mathfrak{M} \vDash \varphi$, we also simply say that $\varphi$ *is true in* $\mathfrak{M}$.

**Proposition 12.41.** *Let $\mathfrak{M}$ be a structure, $\varphi$ be a sentence, and $s$ a variable assignment. $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}, s \vDash \varphi$.*

*Proof.* Exercise. □

**Proposition 12.42.** *Suppose $\varphi(x)$ only contains $x$ free, and $\mathfrak{M}$ is a structure. Then:*

1. *$\mathfrak{M} \vDash \exists x\, \varphi(x)$ iff $\mathfrak{M}, s \vDash \varphi(x)$ for at least one variable assignment $s$.*
2. *$\mathfrak{M} \vDash \forall x\, \varphi(x)$ iff $\mathfrak{M}, s \vDash \varphi(x)$ for all variable assignments $s$.*

*Proof.* Exercise. □

## 12.13 Extensionality

Extensionality, sometimes called relevance, can be expressed informally as follows: the only factors that bears upon the satisfaction of formula $\varphi$ in a structure $\mathfrak{M}$ relative to a variable assignment $s$, are the size of the domain and the assignments made by $\mathfrak{M}$ and $s$ to the elements of the language that actually appear in $\varphi$.

One immediate consequence of extensionality is that where two structures $\mathfrak{M}$ and $\mathfrak{M}'$ agree on all the elements of the language appearing in a sentence $\varphi$ and have the same domain, $\mathfrak{M}$ and $\mathfrak{M}'$ must also agree on whether or not $\varphi$ itself is true.

**Proposition 12.43** (Extensionality)**.** *Let $\varphi$ be a formula, and $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be structures with $|\mathfrak{M}_1| = |\mathfrak{M}_2|$, and $s$ a variable assignment on $|\mathfrak{M}_1| = |\mathfrak{M}_2|$. If $c^{\mathfrak{M}_1} = c^{\mathfrak{M}_2}$, $R^{\mathfrak{M}_1} = R^{\mathfrak{M}_2}$, and $f^{\mathfrak{M}_1} = f^{\mathfrak{M}_2}$ for every constant symbol $c$, relation symbol $R$, and function symbol $f$ occurring in $\varphi$, then $\mathfrak{M}_1, s \vDash \varphi$ iff $\mathfrak{M}_2, s \vDash \varphi$.*

*Proof.* First prove (by induction on $t$) that for every term, $\mathrm{Val}_s^{\mathfrak{M}_1}(t) = \mathrm{Val}_s^{\mathfrak{M}_2}(t)$. Then prove the proposition by induction on $\varphi$, making use of the claim just proved for the induction basis (where $\varphi$ is atomic). □

**Corollary 12.44** (Extensionality for Sentences)**.** *Let $\varphi$ be a sentence and $\mathfrak{M}_1$, $\mathfrak{M}_2$ as in Proposition 12.43. Then $\mathfrak{M}_1 \vDash \varphi$ iff $\mathfrak{M}_2 \vDash \varphi$.*

*Proof.* Follows from Proposition 12.43 by Corollary 12.39. □

Moreover, the value of a term, and whether or not a structure satisfies a formula, only depends on the values of its subterms.

**Proposition 12.45.** *Let $\mathfrak{M}$ be a structure, $t$ and $t'$ terms, and $s$ a variable assignment. Let $s' \sim_x s$ be the $x$-variant of $s$ given by $s'(x) = \mathrm{Val}_s^{\mathfrak{M}}(t')$. Then $\mathrm{Val}_s^{\mathfrak{M}}(t[t'/x]) = \mathrm{Val}_{s'}^{\mathfrak{M}}(t)$.*

*Proof.* By induction on $t$.

1. If $t$ is a constant, say, $t \equiv c$, then $t[t'/x] = c$, and $\mathrm{Val}^{\mathfrak{M}}_s(c) = c^{\mathfrak{M}} = \mathrm{Val}^{\mathfrak{M}}_{s'}(c)$.
2. If $t$ is a variable other than $x$, say, $t \equiv y$, then $t[t'/x] = y$, and $\mathrm{Val}^{\mathfrak{M}}_s(y) = \mathrm{Val}^{\mathfrak{M}}_{s'}(y)$ since $s' \sim_x s$.
3. If $t \equiv x$, then $t[t'/x] = t'$. But $\mathrm{Val}^{\mathfrak{M}}_{s'}(x) = \mathrm{Val}^{\mathfrak{M}}_s(t')$ by definition of $s'$.
4. If $t \equiv f(t_1, \ldots, t_n)$ then we have:

$$\begin{aligned}
\mathrm{Val}^{\mathfrak{M}}_s(t[t'/x]) = & \\
& = \mathrm{Val}^{\mathfrak{M}}_s(f(t_1[t'/x], \ldots, t_n[t'/x])) \\
& \qquad \text{by definition of } t[t'/x] \\
& = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_s(t_1[t'/x]), \ldots, \mathrm{Val}^{\mathfrak{M}}_s(t_n[t'/x])) \\
& \qquad \text{by definition of } \mathrm{Val}^{\mathfrak{M}}_s(f(\ldots)) \\
& = f^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}_{s'}(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_{s'}(t_n)) \\
& \qquad \text{by induction hypothesis} \\
& = \mathrm{Val}^{\mathfrak{M}}_{s'}(t) \text{ by definition of } \mathrm{Val}^{\mathfrak{M}}_{s'}(f(\ldots))
\end{aligned}$$

□

**Proposition 12.46.** *Let $\mathfrak{M}$ be a structure, $\varphi$ a formula, $t$ a term, and $s$ a variable assignment. Let $s' \sim_x s$ be the $x$-variant of $s$ given by $s'(x) = \mathrm{Val}^{\mathfrak{M}}_s(t)$. Then $\mathfrak{M}, s \vDash \varphi[t/x]$ iff $\mathfrak{M}, s' \vDash \varphi$.*

*Proof.* Exercise. □

## 12.14 Semantic Notions

Give the definition of structures for first-order languages, we can define some basic semantic properties of and relationships between sentences. The simplest of these is the notion of *validity* of a sentence. A sentence is valid if it is satisfied in every structure. Valid sentences are those that are satisfied regardless of how the non-logical symbols in it are interpreted. Valid sentences are therefore also called *logical truths*—they are true, i.e., satisfied, in any structure and hence their truth depends only on the logical symbols occurring in them and their syntactic structure, but not on the non-logical symbols or their interpretation.

**Definition 12.47** (Validity)**.** A sentence $\varphi$ is *valid*, $\vDash \varphi$, iff $\mathfrak{M} \vDash \varphi$ for every structure $\mathfrak{M}$.

**Definition 12.48** (Entailment)**.** A set of sentences $\Gamma$ *entails* a sentence $\varphi$, $\Gamma \vDash \varphi$, iff for every structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma$, $\mathfrak{M} \vDash \varphi$.

**Definition 12.49** (Satisfiability)**.** A set of sentences $\Gamma$ is *satisfiable* if $\mathfrak{M} \vDash \Gamma$ for some structure $\mathfrak{M}$. If $\Gamma$ is not satisfiable it is called *unsatisfiable*.

**Proposition 12.50.** *A sentence $\varphi$ is valid iff $\Gamma \vDash \varphi$ for every set of sentences $\Gamma$.*

*Proof.* For the forward direction, let $\varphi$ be valid, and let $\Gamma$ be a set of sentences. Let $\mathfrak{M}$ be a structure so that $\mathfrak{M} \vDash \Gamma$. Since $\varphi$ is valid, $\mathfrak{M} \vDash \varphi$, hence $\Gamma \vDash \varphi$.

For the contrapositive of the reverse direction, let $\varphi$ be invalid, so there is a structure $\mathfrak{M}$ with $\mathfrak{M} \nvDash \varphi$. When $\Gamma = \{\top\}$, since $\top$ is valid, $\mathfrak{M} \vDash \Gamma$. Hence, there is a structure $\mathfrak{M}$ so that $\mathfrak{M} \vDash \Gamma$ but $\mathfrak{M} \nvDash \varphi$, hence $\Gamma$ does not entail $\varphi$. □

**Proposition 12.51.** *$\Gamma \vDash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable.*

*Proof.* For the forward direction, suppose $\Gamma \vDash \varphi$ and suppose to the contrary that there is a structure $\mathfrak{M}$ so that $\mathfrak{M} \vDash \Gamma \cup \{\neg\varphi\}$. Since $\mathfrak{M} \vDash \Gamma$ and $\Gamma \vDash \varphi$, $\mathfrak{M} \vDash \varphi$. Also, since $\mathfrak{M} \vDash \Gamma \cup \{\neg\varphi\}$, $\mathfrak{M} \vDash \neg\varphi$, so we have both $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \nvDash \varphi$, a contradiction. Hence, there can be no such structure $\mathfrak{M}$, so $\Gamma \cup \{\varphi\}$ is unsatisfiable.

For the reverse direction, suppose $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable. So for every structure $\mathfrak{M}$, either $\mathfrak{M} \nvDash \Gamma$ or $\mathfrak{M} \vDash \varphi$. Hence, for every structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma$, $\mathfrak{M} \vDash \varphi$, so $\Gamma \vDash \varphi$. □

**Proposition 12.52.** *If $\Gamma \subseteq \Gamma'$ and $\Gamma \vDash \varphi$, then $\Gamma' \vDash \varphi$.*

*Proof.* Suppose that $\Gamma \subseteq \Gamma'$ and $\Gamma \vDash \varphi$. Let $\mathfrak{M}$ be such that $\mathfrak{M} \vDash \Gamma'$; then $\mathfrak{M} \vDash \Gamma$, and since $\Gamma \vDash \varphi$, we get that $\mathfrak{M} \vDash \varphi$. Hence, whenever $\mathfrak{M} \vDash \Gamma'$, $\mathfrak{M} \vDash \varphi$, so $\Gamma' \vDash \varphi$. □

**Theorem 12.53** (Semantic Deduction Theorem)**.** *$\Gamma \cup \{\varphi\} \vDash \psi$ iff $\Gamma \vDash \varphi \to \psi$.*

*Proof.* For the forward direction, let $\Gamma \cup \{\varphi\} \vDash \psi$ and let $\mathfrak{M}$ be a structure so that $\mathfrak{M} \vDash \Gamma$. If $\mathfrak{M} \vDash \varphi$, then $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$, so since $\Gamma \cup \{\varphi\}$ entails $\psi$, we get $\mathfrak{M} \vDash \psi$. Therefore, $\mathfrak{M} \vDash \varphi \to \psi$, so $\Gamma \vDash \varphi \to \psi$.

For the reverse direction, let $\Gamma \vDash \varphi \to \psi$ and $\mathfrak{M}$ be a structure so that $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$. Then $\mathfrak{M} \vDash \Gamma$, so $\mathfrak{M} \vDash \varphi \to \psi$, and since $\mathfrak{M} \vDash \varphi$, $\mathfrak{M} \vDash \psi$. Hence, whenever $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$, $\mathfrak{M} \vDash \psi$, so $\Gamma \cup \{\varphi\} \vDash \psi$. □

**Proposition 12.54.** *Let $\mathfrak{M}$ be a structure, and $\varphi(x)$ a formula with one free variable $x$, and $t$ a closed term. Then:*

1. $\varphi(t) \vDash \exists x\, \varphi(x)$
2. $\forall x\, \varphi(x) \vDash \varphi(t)$

*Proof.* 1. Suppose $\mathfrak{M} \vDash \varphi(t)$. Let $s$ be a variable assignment with $s(x) = \mathrm{Val}^{\mathfrak{M}}(t)$. Then $\mathfrak{M}, s \vDash \varphi(t)$ since $\varphi(t)$ is a sentence. By Proposition 12.46, $\mathfrak{M}, s \vDash \varphi(x)$. By Proposition 12.42, $\mathfrak{M} \vDash \exists x\, \varphi(x)$.

2. Exercise.

□

## Problems

**Problem 12.1.** Prove Lemma 12.10.

**Problem 12.2.** Prove Proposition 12.11 (Hint: Formulate and prove a version of Lemma 12.10 for terms.)

**Problem 12.3.** Give an inductive definition of the bound variable occurrences along the lines of Definition 12.17.

**Problem 12.4.** Is $\mathfrak{N}$, the standard model of arithmetic, covered? Explain.

**Problem 12.5.** Let $\mathcal{L} = \{c, f, A\}$ with one constant symbol, one one-place function symbol and one two-place predicate symbol, and let the structure $\mathfrak{M}$ be given by

1. $|\mathfrak{M}| = \{1, 2, 3\}$
2. $c^{\mathfrak{M}} = 3$
3. $f^{\mathfrak{M}}(1) = 2, f^{\mathfrak{M}}(2) = 3, f^{\mathfrak{M}}(3) = 2$
4. $A^{\mathfrak{M}} = \{\langle 1, 2\rangle, \langle 2, 3\rangle, \langle 3, 3\rangle\}$

(a) Let $s(v) = 1$ for all variables $v$. Find out whether

$$\mathfrak{M}, s \vDash \exists x\,(A(f(z), c) \to \forall y\,(A(y, x) \vee A(f(y), x)))$$

Explain why or why not.

(b) Give a different structure and variable assignment in which the formula is not satisfied.

**Problem 12.6.** Complete the proof of Proposition 12.38.

**Problem 12.7.** Prove Proposition 12.41

**Problem 12.8.** Prove Proposition 12.42.

**Problem 12.9.** Suppose $\mathcal{L}$ is a language without function symbols. Given a structure $\mathfrak{M}$, $c$ a constant symbol and $a \in |\mathfrak{M}|$, define $\mathfrak{M}[a/c]$ to be the structure that is just like $\mathfrak{M}$, except that $c^{\mathfrak{M}[a/c]} = a$. Define $\mathfrak{M} \mid\!\models \varphi$ for sentences $\varphi$ by:

1. $\varphi \equiv \bot$: not $\mathfrak{M} \mid\!\models \varphi$.
2. $\varphi \equiv R(d_1, \ldots, d_n)$: $\mathfrak{M} \mid\!\models \varphi$ iff $\langle d_1^{\mathfrak{M}}, \ldots, d_n^{\mathfrak{M}}\rangle \in R^{\mathfrak{M}}$.

3. $\varphi \equiv d_1 = d_2$: $\mathfrak{M} \mid\!\models \varphi$ iff $d_1^{\mathfrak{M}} = d_2^{\mathfrak{M}}$.
4. $\varphi \equiv \neg\psi$: $\mathfrak{M} \mid\!\models \varphi$ iff not $\mathfrak{M} \mid\!\models \psi$.
5. $\varphi \equiv (\psi \wedge \chi)$: $\mathfrak{M} \mid\!\models \varphi$ iff $\mathfrak{M} \mid\!\models \psi$ and $\mathfrak{M} \mid\!\models \chi$.
6. $\varphi \equiv (\psi \vee \chi)$: $\mathfrak{M} \mid\!\models \varphi$ iff $\mathfrak{M} \mid\!\models \psi$ or $\mathfrak{M} \mid\!\models \chi$ (or both).
7. $\varphi \equiv (\psi \to \chi)$: $\mathfrak{M} \mid\!\models \varphi$ iff not $\mathfrak{M} \mid\!\models \psi$ or $\mathfrak{M} \mid\!\models \chi$ (or both).
8. $\varphi \equiv \forall x\, \psi$: $\mathfrak{M} \mid\!\models \varphi$ iff for all $a \in |\mathfrak{M}|$, $\mathfrak{M}[a/c] \mid\!\models \psi[c/x]$, if $c$ does not occur in $\psi$.
9. $\varphi \equiv \exists x\, \psi$: $\mathfrak{M} \mid\!\models \varphi$ iff there is an $a \in |\mathfrak{M}|$ such that $\mathfrak{M}[a/c] \mid\!\models \psi[c/x]$, if $c$ does not occur in $\psi$.

Let $x_1, \ldots, x_n$ be all free variables in $\varphi$, $c_1, \ldots, c_n$ constant symbols not in $\varphi$, $a_1, \ldots, a_n \in |\mathfrak{M}|$, and $s(x_i) = a_i$.

Show that $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}[a_1/c_1, \ldots, a_n/c_n] \mid\!\models \varphi[c_1/x_1] \ldots [c_n/x_n]$.

(This problem shows that it is possible to give a semantics for first-order logic that makes do without variable assignments.)

**Problem 12.10.** Suppose that $f$ is a function symbol not in $\varphi(x, y)$. Show that there is a structure $\mathfrak{M}$ such that $\mathfrak{M} \vDash \forall x\, \exists y\, \varphi(x, y)$ iff there is an $\mathfrak{M}'$ such that $\mathfrak{M}' \vDash \forall x\, \varphi(x, f(x))$.

(This problem is a special case of what's known as Skolem's Theorem; $\forall x\, \varphi(x, f(x))$ is called a *Skolem normal form* of $\forall x\, \exists y\, \varphi(x, y)$.)

**Problem 12.11.** Carry out the proof of Proposition 12.43 in detail.

**Problem 12.12.** Prove Proposition 12.46

**Problem 12.13.**

1. Show that $\Gamma \vDash \bot$ iff $\Gamma$ is unsatisfiable.
2. Show that $\Gamma \cup \{\varphi\} \vDash \bot$ iff $\Gamma \vDash \neg\varphi$.
3. Suppose $c$ does not occur in $\varphi$ or $\Gamma$. Show that $\Gamma \vDash \forall x\, \varphi$ iff $\Gamma \vDash \varphi[c/x]$.

**Problem 12.14.** Complete the proof of Proposition 12.54.

# Chapter 13

# Theories and Their Models

## 13.1 Introduction

The development of the axiomatic method is a significant achievement in the history of science, and is of special importance in the history of mathematics. An axiomatic development of a field involves the clarification of many questions: What is the field about? What are the most fundamental concepts? How are they related? Can all the concepts of the field be defined in terms of these fundamental concepts? What laws do, and must, these concepts obey?

The axiomatic method and logic were made for each other. Formal logic provides the tools for formulating axiomatic theories, for proving theorems from the axioms of the theory in a precisely specified way, for studying the properties of all systems satisfying the axioms in a systematic way.

**Definition 13.1.** A set of sentences $\Gamma$ is *closed* iff, whenever $\Gamma \vDash \varphi$ then $\varphi \in \Gamma$. The *closure* of a set of sentences $\Gamma$ is $\{\varphi : \Gamma \vDash \varphi\}$.

We say that $\Gamma$ is *axiomatized by* a set of sentences $\Delta$ if $\Gamma$ is the closure of $\Delta$

We can think of an axiomatic theory as the set of sentences that is axiomatized by its set of axioms $\Delta$. In other words, when we have a first-order language which contains non-logical symbols for the primitives of the axiomatically developed science we wish to study, together with a set of sentences that express the fundamental laws of the science, we can think of the theory as represented by all the sentences in this language that are entailed by the axioms. This ranges from simple examples with only a single primitive and simple axioms, such as the theory of partial orders, to complex theories such as Newtonian mechanics.

The important logical facts that make this formal approach to the axiomatic method so important are the following. Suppose $\Gamma$ is an axiom system for a theory, i.e., a set of sentences.

1. We can state precisely when an axiom system captures an intended class of structures. That is, if we are interested in a certain class of structures, we will successfully capture that class by an axiom system $\Gamma$ iff the structures are exactly those $\mathfrak{M}$ such that $\mathfrak{M} \models \Gamma$.

2. We may fail in this respect because there are $\mathfrak{M}$ such that $\mathfrak{M} \models \Gamma$, but $\mathfrak{M}$ is not one of the structures we intend. This may lead us to add axioms which are not true in $\mathfrak{M}$.

3. If we are successful at least in the respect that $\Gamma$ is true in all the intended structures, then a sentence $\varphi$ is true in all intended structures whenever $\Gamma \models \varphi$. Thus we can use logical tools (such as proof methods) to show that sentences are true in all intended structures simply by showing that they are entailed by the axioms.

4. Sometimes we don't have intended structures in mind, but instead start from the axioms themselves: we begin with some primitives that we want to satisfy certain laws which we codify in an axiom system. One thing that we would like to verify right away is that the axioms do not contradict each other: if they do, there can be no concepts that obey these laws, and we have tried to set up an incoherent theory. We can verify that this doesn't happen by finding a model of $\Gamma$. And if there are models of our theory, we can use logical methods to investigate them, and we can also use logical methods to construct models.

5. The independence of the axioms is likewise an important question. It may happen that one of the axioms is actually a consequence of the others, and so is redundant. We can prove that an axiom $\varphi$ in $\Gamma$ is redundant by proving $\Gamma \setminus \{\varphi\} \models \varphi$. We can also prove that an axiom is not redundant by showing that $(\Gamma \setminus \{\varphi\}) \cup \{\neg\varphi\}$ is satisfiable. For instance, this is how it was shown that the parallel postulate is independent of the other axioms of geometry.

6. Another important question is that of definability of concepts in a theory: The choice of the language determines what the models of a theory consists of. But not every aspect of a theory must be represented separately in its models. For instance, every ordering $\leq$ determines a corresponding strict ordering $<$—given one, we can define the other. So it is not necessary that a model of a theory involving such an order must *also* contain the corresponding strict ordering. When is it the case, in general, that one relation can be defined in terms of others? When is it impossible to define a relation in terms of other (and hence must add it to the primitives of the language)?

## 13.2 Expressing Properties of Structures

It is often useful and important to express conditions on functions and relations, or more generally, that the functions and relations in a structure satisfy these conditions. For instance, we would like to have ways of distinguishing those structures for a language which "capture" what we want the predicate symbols to "mean" from those that do not. Of course we're completely free to specify which structures we "intend," e.g., we can specify that the interpretation of the predicate symbol $\leq$ must be an ordering, or that we are only interested in interpretations of $\mathcal{L}$ in which the domain consists of sets and $\in$ is interpreted by the "is an element of" relation. But can we do this with sentences of the language? In other words, which conditions on a structure $\mathfrak{M}$ can we express by a sentence (or perhaps a set of sentences) in the language of $\mathfrak{M}$? There are some conditions that we will not be able to express. For instance, there is no sentence of $\mathcal{L}_A$ which is only true in a structure $\mathfrak{M}$ if $|\mathfrak{M}| = \mathbb{N}$. We cannot express "the domain contains only natural numbers." But there are "structural properties" of structures that we perhaps can express. Which properties of structures can we express by sentences? Or, to put it another way, which collections of structures can we describe as those making a sentence (or set of sentences) true?

**Definition 13.2** (Model of a set)**.** Let $\Gamma$ be a set of sentences in a language $\mathcal{L}$. We say that a structure $\mathfrak{M}$ *is a model of* $\Gamma$ if $\mathfrak{M} \vDash \varphi$ for all $\varphi \in \Gamma$.

**Example 13.3.** The sentence $\forall x\, x \leq x$ is true in $\mathfrak{M}$ iff $\leq^{\mathfrak{M}}$ is a reflexive relation. The sentence $\forall x \forall y\, ((x \leq y \wedge y \leq x) \rightarrow x = y)$ is true in $\mathfrak{M}$ iff $\leq^{\mathfrak{M}}$ is anti-symmetric. The sentence $\forall x \forall y \forall z\, ((x \leq y \wedge y \leq z) \rightarrow x \leq z)$ is true in $\mathfrak{M}$ iff $\leq^{\mathfrak{M}}$ is transitive. Thus, the models of

$$\begin{aligned} \{ \quad & \forall x\, x \leq x, \\ & \forall x \forall y\, ((x \leq y \wedge y \leq x) \rightarrow x = y), \\ & \forall x \forall y \forall z\, ((x \leq y \wedge y \leq z) \rightarrow x \leq z) \quad \} \end{aligned}$$

are exactly those structures in which $\leq^{\mathfrak{M}}$ is reflexive, anti-symmetric, and transitive, i.e., a partial order. Hence, we can take them as axioms for the *first-order theory of partial orders*.

## 13.3 Examples of First-Order Theories

**Example 13.4.** The theory of strict linear orders in the language $\mathcal{L}_<$ is axiomatized by the set

$$\begin{aligned} & \forall x \neg x < x, \\ & \forall x \forall y\, ((x < y \vee y < x) \vee x = y), \\ & \forall x \forall y \forall z\, ((x < y \wedge y < z) \rightarrow x < z) \end{aligned}$$

It completely captures the intended structures: every strict linear order is a model of this axiom system, and vice versa, if $R$ is a linear order on a set $X$, then the structure $\mathfrak{M}$ with $|\mathfrak{M}| = X$ and $<^{\mathfrak{M}} = R$ is a model of this theory.

**Example 13.5.** The theory of groups in the language $1$ (constant symbol), $\cdot$ (two-place function symbol) is axiomatized by

$$\begin{aligned}
&\forall x\,(x \cdot 1) = x\\
&\forall x\,\forall y\,\forall z\,(x \cdot (y \cdot z)) = ((x \cdot y) \cdot z)\\
&\forall x\,\exists y\,(x \cdot y) = 1
\end{aligned}$$

**Example 13.6.** The theory of Peano arithmetic is axiomatized by the following sentences in the language of arithmetic $\mathcal{L}_A$.

$$\begin{aligned}
&\neg\exists x\, x' = 0\\
&\forall x\,\forall y\,(x' = y' \to x = y)\\
&\forall x\,\forall y\,(x < y \leftrightarrow \exists z\,(x + z' = y))\\
&\forall x\,(x + 0) = x\\
&\forall x\,\forall y\,(x + y') = (x + y)'\\
&\forall x\,(x \times 0) = 0\\
&\forall x\,\forall y\,(x \times y') = ((x \times y) + x)
\end{aligned}$$

plus all sentences of the form

$$(\varphi(0) \wedge \forall x\,(\varphi(x) \to \varphi(x'))) \to \forall x\,\varphi(x)$$

Since there are infinitely many sentences of the latter form, this axiom system is infinite. The latter form is called the *induction schema*. (Actually, the induction schema is a bit more complicated than we let on here.)

The third axiom is an *explicit definition* of $<$.

**Example 13.7.** The theory of pure sets plays an important role in the foundations (and in the philosophy) of mathematics. A set is pure if all its elements are also pure sets. The empty set counts therefore as pure, but a set that has something as an element that is not a set would not be pure. So the pure sets are those that are formed just from the empty set and no "urelements," i.e., objects that are not themselves sets.

The following might be considered as an axiom system for a theory of pure sets:

$$\begin{aligned}
&\exists x\,\neg\exists y\, y \in x\\
&\forall x\,\forall y\,(\forall z(z \in x \leftrightarrow z \in y) \to x = y)\\
&\forall x\,\forall y\,\exists z\,\forall u\,(u \in z \leftrightarrow (u = x \vee u = y))\\
&\forall x\,\exists y\,\forall z\,(z \in y \leftrightarrow \exists u\,(z \in u \wedge u \in x))
\end{aligned}$$

plus all sentences of the form

$$\exists x \, \forall y \, (y \in x \leftrightarrow \varphi(y))$$

The first axiom says that there is a set with no elements (i.e., $\emptyset$ exists); the second says that sets are extensional; the third that for any sets $X$ and $Y$, the set $\{X, Y\}$ exists; the fourth that for any sets $X$ and $Y$, the set $X \cup Y$ exists.

The sentences mentioned last are collectively called the *naive comprehension scheme*. It essentially says that for every $\varphi(x)$, the set $\{x : \varphi(x)\}$ exists—so at first glance a true, useful, and perhaps even necessary axiom. It is called "naive" because, as it turns out, it makes this theory unsatisfiable: if you take $\varphi(y)$ to be $\neg y \in y$, you get the sentence

$$\exists x \, \forall y \, (y \in x \leftrightarrow \neg y \in y)$$

and this sentence is not satisfied in any structure.

**Example 13.8.** In the area of *mereology*, the relation of *parthood* is a fundamental relation. Just like theories of sets, there are theories of parthood that axiomatize various conceptions (sometimes conflicting) of this relation.

The language of mereology contains a single two-place predicate symbol $P$, and $P(x, y)$ "means" that $x$ is a part of $y$. When we have this interpretation in mind, a structure for this language is called a *parthood structure*. Of course, not every structure for a single two-place predicate will really deserve this name. To have a chance of capturing "parthood," $P^{\mathfrak{M}}$ must satisfy some conditions, which we can lay down as axioms for a theory of parthood. For instance, parthood is a partial order on objects: every object is a part (albeit an *improper* part) of itself; no two different objects can be parts of each other; a part of a part of an object is itself part of that object. Note that in this sense "is a part of" resembles "is a subset of," but does not resemble "is an element of" which is neither reflexive nor transitive.

$$\begin{aligned}
&\forall x \, P(x, x), \\
&\forall x \, \forall y \, ((P(x, y) \wedge P(y, x)) \rightarrow x = y), \\
&\forall x \, \forall y \, \forall z \, ((P(x, y) \wedge P(y, z)) \rightarrow P(x, z)),
\end{aligned}$$

Moreover, any two objects have a mereological sum (an object that has these two objects as parts, and is minimal in this respect).

$$\forall x \, \forall y \, \exists z \, \forall u \, (P(z, u) \leftrightarrow (P(x, u) \wedge P(y, u)))$$

These are only some of the basic principles of parthood considered by metaphysicians. Further principles, however, quickly become hard to formulate or write down without first introducting some defined relations. For instance, most metaphysicians interested in mereology also view the following as a valid principle: whenever an object $x$ has a proper part $y$, it also has a part $z$ that has no parts in common with $y$, and so that the fusion of $y$ and $z$ is $x$.

## 13.4 Expressing Relations in a Structure

One main use formulas can be put to is to express properties and relations in a structure $\mathfrak{M}$ in terms of the primitives of the language $\mathcal{L}$ of $\mathfrak{M}$. By this we mean the following: the domain of $\mathfrak{M}$ is a set of objects. The constant symbols, function symbols, and predicate symbols are interpreted in $\mathfrak{M}$ by some objects in$|\mathfrak{M}|$, functions on $|\mathfrak{M}|$, and relations on $|\mathfrak{M}|$. For instance, if $A^2_0$ is in $\mathcal{L}$, then $\mathfrak{M}$ assigns to it a relation $R = A^{2\mathfrak{M}}_0$. Then the formula $A^2_0(v_1, v_2)$ *expresses* that very relation, in the following sense: if a variable assignment $s$ maps $v_1$ to $a \in |\mathfrak{M}|$ and $v_2$ to $b \in |\mathfrak{M}|$, then

$$Rab \quad \text{iff} \quad \mathfrak{M}, s \vDash A^2_0(v_1, v_2).$$

Note that we have to involve variable assignments here: we can't just say "$Rab$ iff $\mathfrak{M} \vDash A^2_0(a, b)$" because $a$ and $b$ are not symbols of our language: they are elements of $|\mathfrak{M}|$.

Since we don't just have atomic formulas, but can combine them using the logical connectives and the quantifiers, more complex formulas can define other relations which aren't directly built into $\mathfrak{M}$. We're interested in how to do that, and specifically, which relations we can define in a structure.

**Definition 13.9.** Let $\varphi(v_1, \ldots, v_n)$ be a formula of $\mathcal{L}$ in which only $v_1, \ldots, v_n$ occur free, and let $\mathfrak{M}$ be a structure for $\mathcal{L}$. $\varphi(v_1, \ldots, v_n)$ *expresses the relation* $R \subseteq |\mathfrak{M}|^n$ iff

$$Ra_1 \ldots a_n \quad \text{iff} \quad \mathfrak{M}, s \vDash \varphi(v_1, \ldots, v_n)$$

for any variable assignment $s$ with $s(v_i) = a_i$ $(i = 1, \ldots, n)$.

**Example 13.10.** In the standard model of arithmetic $\mathfrak{N}$, the formula $v_1 < v_2 \vee v_1 = v_2$ expresses the $\leq$ relation on $\mathbb{N}$. The formula $v_2 = v_1'$ expresses the successor relation, i.e., the relation $R \subseteq \mathbb{N}^2$ where $Rnm$ holds if $m$ is the successor of $n$. The formula $v_1 = v_2'$ expresses the predecessor relation. The formulas $\exists v_3\,(v_3 \neq 0 \wedge v_2 = (v_1 + v_3))$ and $\exists v_3\,(v_1 + v_3{}') = v_2$ both express the $<$ relation. This means that the predicate symbol $<$ is actually superfluous in the language of arithmetic; it can be defined.

This idea is not just interesting in specific structures, but generally whenever we use a language to describe an intended model or models, i.e., when we consider theories. These theories often only contain a few predicate symbols as basic symbols, but in the domain they are used to describe often many other relations play an important role. If these other relations can be systematically expressed by the relations that interpret the basic predicate symbols of the language, we say we can *define* them in the language.

## 13.5 The Theory of Sets

Almost all of mathematics can be developed in the theory of sets. Developing mathematics in this theory involves a number of things. First, it requires a set of axioms for the relation $\in$. A number of different axiom systems have been developed, sometimes with conflicting properties of $\in$. The axiom system known as **ZFC**, Zermelo-Fraenkel set theory with the axiom of choice stands out: it is by far the most widely used and studied, because it turns out that its axioms suffice to prove almost all the things mathematicians expect to be able to prove. But before that can be established, it first is necessary to make clear how we can even *express* all the things mathematicians would like to express. For starters, the language contains no constant symbols or function symbols, so it seems at first glance unclear that we can talk about particular sets (such as $\emptyset$ or $\mathbb{N}$), can talk about operations on sets (such as $X \cup Y$ and $\wp(X)$), let alone other constructions which involve things other than sets, such as relations and functions.

To begin with, "is an element of" is not the only relation we are interested in: "is a subset of" seems almost as important. But we can *define* "is a subset of" in terms of "is an element of." To do this, we have to find a formula $\varphi(x, y)$ in the language of set theory which is satisfied by a pair of sets $\langle X, Y \rangle$ iff $X \subseteq Y$. But $X$ is a subset of $Y$ just in case all elements of $X$ are also elements of $Y$. So we can define $\subseteq$ by the formula

$$\forall z\, (z \in x \to z \in y)$$

Now, whenever we want to use the relation $\subseteq$ in a formula, we could instead use that formula (with $x$ and $y$ suitably replaced, and the bound variable $z$ renamed if necessary). For instance, extensionality of sets means that if any sets $x$ and $y$ are contained in each other, then $x$ and $y$ must be the same set. This can be expressed by $\forall x\, \forall y\, ((x \subseteq y \wedge y \subseteq x) \to x = y)$, or, if we replace $\subseteq$ by the above definition, by

$$\forall x\, \forall y\, ((\forall z\, (z \in x \to z \in y) \wedge \forall z\, (z \in y \to z \in x)) \to x = y).$$

This is in fact one of the axioms of **ZFC**, the "axiom of extensionality."

There is no constant symbol for $\emptyset$, but we can express "$x$ is empty" by $\neg \exists y\, y \in x$. Then "$\emptyset$ exists" becomes the sentence $\exists x\, \neg \exists y\, y \in x$. This is another axiom of **ZFC**. (Note that the axiom of extensionality implies that there is only one empty set.) Whenever we want to talk about $\emptyset$ in the language of set theory, we would write this as "there is a set that's empty and ..." As an example, to express the fact that $\emptyset$ is a subset of every set, we could write

$$\exists x\, (\neg \exists y\, y \in x \wedge \forall z\, x \subseteq z)$$

where, of course, $x \subseteq z$ would in turn have to be replaced by its definition.

To talk about operations on sets, such has $X \cup Y$ and $\wp(X)$, we have to use a similar trick. There are no function symbols in the language of set theory, but we can express the functional relations $X \cup Y = Z$ and $\wp(X) = Y$ by

$$\forall u\,((u \in x \vee u \in y) \leftrightarrow u \in z)$$
$$\forall u\,(u \subseteq x \leftrightarrow u \in y)$$

since the elements of $X \cup Y$ are exactly the sets that are either elements of $X$ or elements of $Y$, and the elements of $\wp(X)$ are exactly the subsets of $X$. However, this doesn't allow us to use $x \cup y$ or $\wp(x)$ as if they were terms: we can only use the entire formulas that define the relations $X \cup Y = Z$ and $\wp(X) = Y$. In fact, we do not know that these relations are ever satisfied, i.e., we do not know that unions and power sets always exist. For instance, the sentence $\forall x\,\exists y\,\wp(x) = y$ is another axiom of **ZFC** (the power set axiom).

Now what about talk of ordered pairs or functions? Here we have to explain how we can think of ordered pairs and functions as special kinds of sets. One way to define the ordered pair $\langle x, y\rangle$ is as the set $\{\{x\}, \{x, y\}\}$. But like before, we cannot introduce a function symbol that names this set; we can only define the relation $\langle x, y\rangle = z$, i.e., $\{\{x\}, \{x, y\}\} = z$:

$$\forall u\,(u \in z \leftrightarrow (\forall v\,(v \in u \leftrightarrow v = x) \vee \forall v\,(v \in u \leftrightarrow (v = x \vee v = y))))$$

This says that the elements $u$ of $z$ are exactly those sets which either have $x$ as its only element or have $x$ and $y$ as its only elements (in other words, those sets that are either identical to $\{x\}$ or identical to $\{x, y\}$). Once we have this, we can say further things, e.g., that $X \times Y = Z$:

$$\forall z\,(z \in Z \leftrightarrow \exists x\,\exists y\,(x \in X \wedge y \in Y \wedge \langle x, y\rangle = z))$$

A function $f\colon X \to Y$ can be thought of as the relation $f(x) = y$, i.e., as the set of pairs $\{\langle x, y\rangle : f(x) = y\}$. We can then say that a set $f$ is a function from $X$ to $Y$ if (a) it is a relation $\subseteq X \times Y$, (b) it is total, i.e., for all $x \in X$ there is some $y \in Y$ such that $\langle x, y\rangle \in f$ and (c) it is functional, i.e., whenever $\langle x, y\rangle, \langle x, y'\rangle \in f$, $y = y'$ (because values of functions must be unique). So "$f$ is a function from $X$ to $Y$" can be written as:

$$\begin{aligned}&\forall u\,(u \in f \to \exists x\,\exists y\,(x \in X \wedge y \in Y \wedge \langle x, y\rangle = u)) \wedge\\ &\forall x\,(x \in X \to (\exists y\,(y \in Y \wedge \mathrm{maps}(f, x, y)) \wedge\\ &\qquad\qquad (\forall y\,\forall y'\,((\mathrm{maps}(f, x, y) \wedge \mathrm{maps}(f, x, y')) \to y = y')))\end{aligned}$$

where $\mathrm{maps}(f, x, y)$ abbreviates $\exists v\,(v \in f \wedge \langle x, y\rangle = v)$ (this formula expresses "$f(x) = y$").

It is now also not hard to express that $f\colon X \to Y$ is injective, for instance:

$$f\colon X \to Y \wedge \forall x\,\forall x'\,((x \in X \wedge x' \in X \wedge \exists y\,(\mathrm{maps}(f, x, y) \wedge \mathrm{maps}(f, x', y))) \to x = x')$$

A function $f\colon X \to Y$ is injective iff, whenever $f$ maps $x, x' \in X$ to a single $y$, $x = x'$. If we abbreviate this formula as $\mathrm{inj}(f, X, Y)$, we're already in a position to state in the language of set theory something as non-trivial as Cantor's theorem: there is no injective function from $\wp(X)$ to $X$:

$$\forall X\, \forall Y\, (\wp(X) = Y \to \neg \exists f\, \mathrm{inj}(f, Y, X))$$

One might think that set theory requires another axiom that guarantees the existence of a set for every defining property. If $\varphi(x)$ is a formula of set theory with the variable $x$ free, we can consider the sentence

$$\exists y\, \forall x\, (x \in y \leftrightarrow \varphi(x)).$$

This sentence states that there is a set $y$ whose elements are all and only those $x$ that satisfy $\varphi(x)$. This schema is called the "comprehension principle." It looks very useful; unfortunately it is inconsistent. Take $\varphi(x) \equiv \neg x \in x$, then the comprehension principle states

$$\exists y\, \forall x\, (x \in y \leftrightarrow x \notin x),$$

i.e., it states the existence of a set of all sets that are not elements of themselves. No such set can exist—this is Russell's Paradox. **ZFC**, in fact, contains a restricted—and consistent—version of this principle, the separation principle:

$$\forall z\, \exists y\, \forall x\, (x \in y \leftrightarrow (x \in z \wedge \varphi(x)).$$

## 13.6 Expressing the Size of Structures

There are some properties of structures we can express even without using the non-logical symbols of a language. For instance, there are sentences which are true in a structure iff the domain of the structure has at least, at most, or exactly a certain number $n$ of elements.

**Proposition 13.11.** *The sentence*

$$\begin{aligned}\varphi_{\geq n} \equiv \exists x_1\, \exists x_2\, \ldots \exists x_n \quad (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge \cdots \wedge x_1 \neq x_n \wedge \\ x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge \cdots \wedge x_2 \neq x_n \wedge \\ \vdots \\ x_{n-1} \neq x_n)\end{aligned}$$

*is true in a structure $\mathfrak{M}$ iff $|\mathfrak{M}|$ contains at least $n$ elements. Consequently, $\mathfrak{M} \models \neg\varphi_{\geq n+1}$ iff $|\mathfrak{M}|$ contains at most $n$ elements.*

**Proposition 13.12.** *The sentence*

$$\begin{aligned}\varphi_{=n} \equiv \exists x_1\, \exists x_2 \ldots \exists x_n \quad (&x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge \cdots \wedge x_1 \neq x_n \wedge \\ &x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge \cdots \wedge x_2 \neq x_n \wedge \\ &\vdots \\ &x_{n-1} \neq x_n \wedge \\ &\forall y\, (y = x_1 \vee \ldots y = x_n) \ldots))\end{aligned}$$

*is true in a structure $\mathfrak{M}$ iff $|\mathfrak{M}|$ contains exactly $n$ elements.*

**Proposition 13.13.** *A structure is infinite iff it is a model of*

$$\{\varphi_{\geq 1}, \varphi_{\geq 2}, \varphi_{\geq 3}, \ldots\}$$

There is no single purely logical sentence which is true in $\mathfrak{M}$ iff $|\mathfrak{M}|$ is infinite. However, one can give sentences with non-logical predicate symbols which only have infinite models (although not every infinite structure is a model of them). The property of being a finite structure, and the property of being a non-enumerable structure cannot even be expressed with an infinite set of sentences. These facts follow from the compactness and Löwenheim-Skolem theorems.

## Problems

**Problem 13.1.** Find formulas in $\mathcal{L}_A$ which define the following relations:

1. $n$ is between $i$ and $j$;
2. $n$ evenly divides $m$ (i.e., $m$ is a multiple of $n$);
3. $n$ is a prime number (i.e., no number other than 1 and $n$ evenly divides $n$).

**Problem 13.2.** Suppose the formula $\varphi(v_1, v_2)$ expresses the relation $R \subseteq |\mathfrak{M}|^2$ in a structure $\mathfrak{M}$. Find formulas that express the following relations:

1. the inverse $R^{-1}$ of $R$;
2. the relative product $R \mid R$;

Can you find a way to express $R^+$, the transitive closure of $R$?

**Problem 13.3.** Let $\mathcal{L}$ be the language containing a 2-place predicate symbol $<$ only (no other constant symbols, function symbols or predicate symbols—except of course $=$). Let $\mathfrak{N}$ be the structure such that $|\mathfrak{N}| = \mathbb{N}$, and $<^{\mathfrak{N}} = \{\langle n, m\rangle : n < m\}$. Prove the following:

1. $\{0\}$ is definable in $\mathfrak{N}$;
2. $\{1\}$ is definable in $\mathfrak{N}$;
3. $\{2\}$ is definable in $\mathfrak{N}$;
4. for each $n \in \mathbb{N}$, the set $\{n\}$ is definable in $\mathfrak{N}$;
5. every finite subset of $|\mathfrak{N}|$ is definable in $\mathfrak{N}$;
6. every co-finite subset of $|\mathfrak{N}|$ is definable in $\mathfrak{N}$ (where $X \subseteq \mathbb{N}$ is co-finite iff $\mathbb{N} \setminus X$ is finite).

**Problem 13.4.** Show that the comprehension principle is inconsistent by giving a derivation that shows

$$\exists y \, \forall x \, (x \in y \leftrightarrow x \notin x) \vdash \bot.$$

It may help to first show $(A \to \neg A) \wedge (\neg A \to A) \vdash \bot$.

# Chapter 14

# Derivation Systems

This chapter collects general material on derivation systems. A textbook using a specific system can insert the introduction section plus the relevant survey section at the beginning of the chapter introducing that system.

## 14.1 Introduction

Logics commonly have both a semantics and a derivation system. The semantics concerns concepts such as truth, satisfiability, validity, and entailment. The purpose of derivation systems is to provide a purely syntactic method of establishing entailment and validity. They are purely syntactic in the sense that a derivation in such a system is a finite syntactic object, usually a sequence (or other finite arrangement) of sentences or formulas. Good derivation systems have the property that any given sequence or arrangement of sentences or formulas can be verified mechanically to be "correct."

The simplest (and historically first) derivation systems for first-order logic were *axiomatic*. A sequence of formulas counts as a derivation in such a system if each individual formula in it is either among a fixed set of "axioms" or follows from formulas coming before it in the sequence by one of a fixed number of "inference rules"—and it can be mechanically verified if a formula is an axiom and whether it follows correctly from other formulas by one of the inference rules. Axiomatic proof systems are easy to describe—and also easy to handle meta-theoretically—but derivations in them are hard to read and understand, and are also hard to produce.

Other derivation systems have been developed with the aim of making it easier to construct derivations or easier to understand derivations once they are complete. Examples are natural deduction, truth trees, also known as tableaux proofs, and the sequent calculus. Some derivation systems are de-

signed especially with mechanization in mind, e.g., the resolution method is easy to implement in software (but its derivations are essentially impossible to understand). Most of these other proof systems represent derivations as trees of formulas rather than sequences. This makes it easier to see which parts of a derivation depend on which other parts.

So for a given logic, such as first-order logic, the different derivation systems will give different explications of what it is for a sentence to be a *theorem* and what it means for a sentence to be derivable from some others. However that is done (via axiomatic derivations, natural deductions, sequent derivations, truth trees, resolution refutations), we want these relations to match the semantic notions of validity and entailment. Let's write $\vdash \varphi$ for "$\varphi$ is a theorem" and "$\Gamma \vdash \varphi$" for "$\varphi$ is derivable from $\Gamma$." However $\vdash$ is defined, we want it to match up with $\models$, that is:

1. $\vdash \varphi$ if and only if $\models \varphi$
2. $\Gamma \vdash \varphi$ if and only if $\Gamma \models \varphi$

The "only if" direction of the above is called *soundness*. A derivation system is sound if derivability guarantees entailment (or validity). Every decent derivation system has to be sound; unsound derivation systems are not useful at all. After all, the entire purpose of a derivation is to provide a syntactic guarantee of validity or entailment. We'll prove soundness for the derivation systems we present.

The converse "if" direction is also important: it is called *completeness*. A complete derivation system is strong enough to show that $\varphi$ is a theorem whenever $\varphi$ is valid, and that there $\Gamma \vdash \varphi$ whenever $\Gamma \models \varphi$. Completeness is harder to establish, and some logics have no complete derivation systems. First-order logic does. Kurt Gödel was the first one to prove completeness for a derivation system of first-order logic in his 1929 dissertation.

Another concept that is connected to derivation systems is that of *consistency*. A set of sentences is called inconsistent if anything whatsoever can be derived from it, and consistent otherwise. Inconsistency is the syntactic counterpart to unsatisfiablity: like unsatisfiable sets, inconsistent sets of sentences do not make good theories, they are defective in a fundamental way. Consistent sets of sentences may not be true or useful, but at least they pass that minimal threshold of logical usefulness. For different derivation systems the specific definition of consistency of sets of sentences might differ, but like $\vdash$, we want consistency to coincide with its semantic counterpart, satisfiability. We want it to always be the case that $\Gamma$ is consistent if and only if it is satisfiable. Here, the "if" direction amounts to completeness (consistency guarantees satisfiability), and the "only if" direction amounts to soundness (satisfiability guarantees consistency). In fact, for classical first-order logic, the two versions of soundness and completeness are equivalent.

## 14.2 The Sequent Calculus

While many derivation systems operate with arrangements of sentences, the sequent calculus operates with *sequents*. A sequent is an expression of the form

$$\varphi_1, \ldots, \varphi_m \Rightarrow \psi_1, \ldots, \psi_m,$$

that is a pair of sequences of sentences, separated by the sequent symbol $\Rightarrow$. Either sequence may be empty. A derivation in the sequent calculus is a tree of sequents, where the topmost sequents are of a special form (they are called "initial sequents" or "axioms") and every other sequent follows from the sequents immediately above it by one of the rules of inference. The rules of inference either manipulate the sentences in the sequents (adding, removing, or rearranging them on either the left or the right), or they introduce a complex formula in the conclusion of the rule. For instance, the $\wedge$L rule allows the inference from $\varphi, \Gamma \Rightarrow \Delta$ to $A \wedge \psi, \Gamma \Rightarrow \Delta$, and the $\rightarrow$R allows the inference from $\varphi, \Gamma \Rightarrow \Delta, \psi$ to $\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi$, for any $\Gamma$, $\Delta$, $\varphi$, and $\psi$. (In particular, $\Gamma$ and $\Delta$ may be empty.)

The $\vdash$ relation based on the sequent calculus is defined as follows: $\Gamma \vdash \varphi$ iff there is some sequence $\Gamma_0$ such that every $\varphi$ in $\Gamma_0$ is in $\Gamma$ and there is a derivation with the sequent $\Gamma_0 \Rightarrow \varphi$ at its root. $\varphi$ is a theorem in the sequent calculus if the sequent $\Rightarrow \varphi$ has a derivation. For instance, here is a derivation that shows that $\vdash (\varphi \wedge \psi) \rightarrow \varphi$:

$$\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi}\wedge\text{L}}{\Rightarrow (\varphi \wedge \psi) \rightarrow \varphi}\rightarrow\text{R}$$

A set $\Gamma$ is inconsistent in the sequent calculus if there is a derivation of $\Gamma_0 \Rightarrow$ (where every $\varphi \in \Gamma_0$ is in $\Gamma$ and the right side of the sequent is empty). Using the rule WR, any sentence can be derived from an inconsistent set.

The sequent calculus was invented in the 1930s by Gerhard Gentzen. Because of its systematic and symmetric design, it is a very useful formalism for developing a theory of derivations. It is relatively easy to find derivations in the sequent calculus, but these derivations are often hard to read and their connection to proofs are sometimes not easy to see. It has proved to be a very elegant approach to derivation systems, however, and many logics have sequent calculus systems.

## 14.3 Natural Deduction

Natural deduction is a derivation system intended to mirror actual reasoning (especially the kind of regimented reasoning employed by mathematicians). Actual reasoning proceeds by a number of "natural" patterns. For instance,

proof by cases allows us to establish a conclusion on the basis of a disjunctive premise, by establishing that the conclusion follows from either of the disjuncts. Indirect proof allows us to establish a conclusion by showing that its negation leads to a contradiction. Conditional proof establishes a conditional claim "if ... then ..." by showing that the consequent follows from the antecedent. Natural deduction is a formalization of some of these natural inferences. Each of the logical connectives and quantifiers comes with two rules, an introduction and an elimination rule, and they each correspond to one such natural inference pattern. For instance, $\rightarrow$Intro corresponds to conditional proof, and $\vee$Elim to proof by cases. A particularly simple rule is $\wedge$Elim which allows the inference from $\varphi \wedge \psi$ to $\varphi$ (or $\psi$).

One feature that distinguishes natural deduction from other derivation systems is its use of assumptions. A derivation in natural deduction is a tree of formulas. A single formula stands at the root of the tree of formulas, and the "leaves" of the tree are formulas from which the conclusion is derived. In natural deduction, some leaf formulas play a role inside the derivation but are "used up" by the time the derivation reaches the conclusion. This corresponds to the practice, in actual reasoning, of introducing hypotheses which only remain in effect for a short while. For instance, in a proof by cases, we assume the truth of each of the disjuncts; in conditional proof, we assume the truth of the antecedent; in indirect proof, we assume the truth of the negation of the conclusion. This way of introducing hypothetical assumptions and then doing away with them in the service of establishing an intermediate step is a hallmark of natural deduction. The formulas at the leaves of a natural deduction derivation are called assumptions, and some of the rules of inference may "discharge" them. For instance, if we have a derivation of $\psi$ from some assumptions which include $\varphi$, then the $\rightarrow$Intro rule allows us to infer $\varphi \rightarrow \psi$ and discharge any assumption of the form $\varphi$. (To keep track of which assumptions are discharged at which inferences, we label the inference and the assumptions it discharges with a number.) The assumptions that remain undischarged at the end of the derivation are together sufficient for the truth of the conclusion, and so a derivation establishes that its undischarged assumptions entail its conclusion.

The relation $\Gamma \vdash \varphi$ based on natural deduction holds iff there is a derivation in which $\varphi$ is the last sentence in the tree, and every leaf which is undischarged is in $\Gamma$. $\varphi$ is a theorem in natural deduction iff there is a derivation in which $\varphi$ is the last sentence and all assumptions are discharged. For instance, here is a derivation that shows that $\vdash (\varphi \wedge \psi) \rightarrow \varphi$:

$$\cfrac{\cfrac{[\varphi \wedge \psi]^1}{\varphi}\,\wedge\text{Elim}}{(\varphi \wedge \psi) \rightarrow \varphi}\,{}^{1}\,\rightarrow\text{Intro}$$

The label 1 indicates that the assumption $\varphi \wedge \psi$ is discharged at the $\rightarrow$Intro inference.

A set $\Gamma$ is inconsistent iff $\Gamma \vdash \bot$ in natural deduction. The rule $\bot_I$ makes it so that from an inconsistent set, any sentence can be derived.

Natural deduction systems were developed by Gerhard Gentzen and Stanisław Jaśkowski in the 1930s, and later developed by Dag Prawitz and Frederic Fitch. Because its inferences mirror natural methods of proof, it is favored by philosophers. The versions developed by Fitch are often used in introductory logic textbooks. In the philosophy of logic, the rules of natural deduction have sometimes been taken to give the meanings of the logical operators ("proof-theoretic semantics").

## 14.4 Tableaux

While many derivation systems operate with arrangements of sentences, tableaux operate with signed formulas. A signed formula is a pair consisting of a truth value sign ($\mathbb{T}$ or $\mathbb{F}$) and a sentence

$$\mathbb{T}\varphi \text{ or } \mathbb{F}\varphi.$$

A tableau consists of signed formulas arranged in a downward-branching tree. It begins with a number of *assumptions* and continues with signed formulas which result from one of the signed formulas above it by applying one of the rules of inference. Each rule allows us to add one or more signed formulas to the end of a branch, or two signed formulas side by side—in this case a branch splits into two, with the two added signed formulas forming the ends of the two branches.

A rule applied to a complex signed formula results in the addition of signed formulas which are immediate sub-formulas. They come in pairs, one rule for each of the two signs. For instance, the $\wedge\mathbb{T}$ rule applies to $\mathbb{T}\varphi \wedge \psi$, and allows the addition of both the two signed formulas $\mathbb{T}\varphi$ and $\mathbb{T}\psi$ to the end of any branch containing $\mathbb{T}\varphi \wedge \psi$, and the rule $\varphi \wedge \psi\mathbb{F}$ allows a branch to be split by adding $\mathbb{F}\varphi$ and $\mathbb{F}\psi$ side-by-side. A tableau is closed if every one of its branches contains a matching pair of signed formulas $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$.

The $\vdash$ relation based on tableaux is defined as follows: $\Gamma \vdash \varphi$ iff there is some finite set $\Gamma_0 = \{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ such that there is a closed tableau for the assumptions

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

For instance, here is a closed tableau that shows that $\vdash (\varphi \wedge \psi) \rightarrow \varphi$:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\varphi$ | $\to\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\psi$ | $\to\mathbb{T}\,2$ |
| | $\otimes$ | |

A set $\Gamma$ is inconsistent in the tableau calculus if there is a closed tableau for assumptions

$$\{\mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

for some $\psi_i \in \Gamma$.

The sequent calculus was invented in the 1950s independently by Evert Beth and Jaakko Hintikka, and simplified and popularized by Raymond Smullyan. It is very easy to use, since constructing a tableau is a very systematic procedure. Because of the systematic nature of tableaux, they also lend themselves to implementation by computer. However, tableau is often hard to read and their connection to proofs are sometimes not easy to see. The approach is also quite general, and many different logics have tableau systems. Tableaux also help us to find structures that satisfy given (sets of) sentences: if the set is satisfiable, it won't have a closed tableau, i.e., any tableau will have an open branch. The satisfying structure can be "read off" an open branch, provided all rules it is possible to apply have been applied on that branch. There is also a very close connection to the sequent calculus: essentially, a closed tableau is a condensed derivation in the sequent calculus, written upside-down.

## 14.5 Axiomatic Derivations

Axiomatic derivations are the oldest and simplest logical derivation systems. Its derivations are simply sequences of sentences. A sequence of sentences conunts as a correct derivation if every sentence $\varphi$ in it satisfies one of the following conditions:

1. $\varphi$ is an axiom, or
2. $\varphi$ is an element of a given set $\Gamma$ of sentences, or
3. $\varphi$ is justified by a rule of inference.

To be an axiom, $\varphi$ has to have the form of on of a number of fixed sentence schemas. There are many sets of axiom schemas that provide a satisfactory (sound and complete) derivation system for first-order logic. Some are organized according to the connectives they govern, e.g., the schemas

$$\varphi \to (\psi \to \varphi) \qquad \psi \to (\psi \vee \chi) \qquad (\psi \wedge \chi) \to \psi$$

are common axioms that govern $\rightarrow$, $\vee$ and $\wedge$. Some axiom systems aim at a minimal number of axioms. Depending on the connectives that are taken as primitives, it is even possible to find axiom systems that consist of a single axiom.

A rule of inference is a conditional statement that gives a sufficient condition for a sentence in a derivation to be justified. Modus ponens is one very common such rule: it says that if $\varphi$ and $\varphi \rightarrow \psi$ are already justified, then $\psi$ is justified. This means that a line in a derivation containing the sentence $\psi$ is justified, provided that both $\varphi$ and $\varphi \rightarrow \psi$ (for some sentence $\varphi$) appear in the derivation before $\psi$.

The $\vdash$ relation based on axiomatic derivations is defined as follows: $\Gamma \vdash \varphi$ iff there is a derivation with the sentence $\varphi$ as its last formula (and $\Gamma$ is taken as the set of sentences in that derivation which are justified by (2) above). $\varphi$ is a theorem if $\varphi$ has a derivation where $\Gamma$ is empty, i.e., every sentence in the derivation is justfied either by (1) or (3). For instance, here is a derivation that shows that $\vdash \varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi))$:

1. $\psi \rightarrow (\psi \vee \varphi)$
2. $(\psi \rightarrow (\psi \vee \varphi)) \rightarrow (\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi)))$
3. $\varphi \rightarrow (\psi \rightarrow (\psi \vee \varphi))$

The sentence on line 1 is of the form of the axiom $\varphi \rightarrow (\varphi \vee \psi)$ (with the roles of $\varphi$ and $\psi$ reversed). The sentence on line 2 is of the form of the axiom $\varphi \rightarrow (\psi \rightarrow \varphi)$. Thus, both lines are justified. Line 3 is justified by modus ponens: if we abbreviate it as $\theta$, then line 2 has the form $\chi \rightarrow \theta$, where $\chi$ is $\psi \rightarrow (\psi \vee \varphi)$, i.e., line 1.

A set $\Gamma$ is inconsistent if $\Gamma \vdash \bot$. A complete axiom system will also prove that $\bot \rightarrow \varphi$ for any $\varphi$, and so if $\Gamma$ is inconsistent, then $\Gamma \vdash \varphi$ for any $\varphi$.

Systems of axiomatic derivations for logic were first given by Gottlob Frege in his 1879 *Begriffsschrift*, which for this reason is often considered the first work of modern logic. They were perfected in Alfred North Whitehead and Bertrand Russell's *Principia Mathematica* and by David Hilbert and his students in the 1920s. They are thus often called "Frege systems" or "Hilbert systems." They are very versatile in that it is often easy to find an axiomatic system for a logic. Because derivations have a very simple structure and only one or two inference rules, it is also relatively easy to prove things *about* them. However, they are very hard to use in practice, i.e., it is difficult to find and write proofs.

# Chapter 15

# The Sequent Calculus

This chapter presents Gentzen's standard sequent calculus LK for classical first-order logic. It could use more examples and exercises. To include or exclude material relevant to the sequent calculus as a proof system, use the "prfLK" tag.

## 15.1 Rules and Derivations

For the following, let $\Gamma, \Delta, \Pi, \Lambda$ represent finite sequences of sentences.

**Definition 15.1** (Sequent)**.** A *sequent* is an expression of the form

$$\Gamma \Rightarrow \Delta$$

where $\Gamma$ and $\Delta$ are finite (possibly empty) sequences of sentences of the language $\mathcal{L}$. $\Gamma$ is called the *antecedent*, while $\Delta$ is the *succedent*.

The intuitive idea behind a sequent is: if all of the sentences in the antecedent hold, then at least one of the sentences in the succedent holds. That is, if $\Gamma = \langle \varphi_1, \ldots, \varphi_m \rangle$ and $\Delta = \langle \psi_1, \ldots, \psi_n \rangle$, then $\Gamma \Rightarrow \Delta$ holds iff

$$(\varphi_1 \wedge \cdots \wedge \varphi_m) \rightarrow (\psi_1 \vee \cdots \vee \psi_n)$$

holds. There are two special cases: where $\Gamma$ is empty and when $\Delta$ is empty. When $\Gamma$ is empty, i.e., $m = 0$, $\Rightarrow \Delta$ holds iff $\psi_1 \vee \cdots \vee \psi_n$ holds. When $\Delta$ is empty, i.e., $n = 0$, $\Gamma \Rightarrow$ holds iff $\neg(\varphi_1 \wedge \cdots \wedge \varphi_m)$ does. We say a sequent is valid iff the corresponding sentence is valid.

If $\Gamma$ is a sequence of sentences, we write $\Gamma, \varphi$ for the result of appending $\varphi$ to the right end of $\Gamma$ (and $\varphi, \Gamma$ for the result of appending $\varphi$ to the left end of $\Gamma$). If $\Delta$ is a sequence of sentences also, then $\Gamma, \Delta$ is the concatenation of the two sequences.

**Definition 15.2** (Initial Sequent)**.** An *initial sequent* is a sequent of one of the following forms:

1. $\varphi \Rightarrow \varphi$
2. $\bot \Rightarrow$

for any sentence $\varphi$ in the language.

Derivations in the sequent calculus are certain trees of sequents, where the topmost sequents are initial sequents, and if a sequent stands below one or two other sequents, it must follow correctly by a rule of inference. The rules for **LK** are divided into two main types: *logical* rules and *structural* rules. The logical rules are named for the main operator of the sentence containing $\varphi$ and/or $\psi$ in the lower sequent. Each one comes in two versions, one for inferring a sequent with the sentence containg the logical operator on the left, and one with the sentence on the right.

## 15.2 Propositional Rules

**Rules for $\neg$**

$$\frac{\Gamma \Rightarrow \Delta, \varphi}{\neg\varphi, \Gamma \Rightarrow \Delta}\,\neg\mathrm{L} \qquad \frac{\varphi, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg\varphi}\,\neg\mathrm{R}$$

**Rules for $\wedge$**

$$\frac{\varphi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\,\wedge\mathrm{L} \qquad \frac{\psi, \Gamma \Rightarrow \Delta}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\,\wedge\mathrm{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}\,\wedge\mathrm{R}$$

**Rules for $\vee$**

$$\frac{\varphi, \Gamma \Rightarrow \Delta \qquad \psi, \Gamma \Rightarrow \Delta}{\varphi \vee \psi, \Gamma \Rightarrow \Delta}\,\vee\mathrm{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\,\vee\mathrm{R} \qquad \frac{\Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \vee \psi}\,\vee\mathrm{R}$$

### Rules for $\rightarrow$

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \psi, \Pi \Rightarrow \Lambda}{\varphi \rightarrow \psi, \Gamma, \Pi \Rightarrow \Delta, \Lambda} \rightarrow\text{L} \qquad \frac{\varphi, \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi} \rightarrow\text{R}$$

## 15.3 Quantifier Rules

### Rules for $\forall$

$$\frac{\varphi(t), \Gamma \Rightarrow \Delta}{\forall x\, \varphi(x), \Gamma \Rightarrow \Delta} \forall\text{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi(a)}{\Gamma \Rightarrow \Delta, \forall x\, \varphi(x)} \forall\text{R}$$

In $\forall$L, $t$ is a closed term (i.e., one without variables). In $\forall$R, $a$ is a constant symbol which must not occur anywhere in the lower sequent of the $\forall$R rule. We call $a$ the *eigenvariable* of the $\forall$R inference.

### Rules for $\exists$

$$\frac{\varphi(a), \Gamma \Rightarrow \Delta}{\exists x\, \varphi(x), \Gamma \Rightarrow \Delta} \exists\text{L} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi(t)}{\Gamma \Rightarrow \Delta, \exists x\, \varphi(x)} \exists\text{R}$$

Again, $t$ is a closed term, and $a$ is a constant symbol which does not occur in the lower sequent of the $\exists$L rule. We call $a$ the *eigenvariable* of the $\exists$L inference.

The condition that an eigenvariable not occur in the lower sequent of the $\forall$R or $\exists$L inference is called the *eigenvariable condition*.

We use the term "eigenvariable" even though $a$ in the above rules is a constant symbol. This has historical reasons.

In $\exists$R and $\forall$L there are no restrictions on the term $t$. On the other hand, in the $\exists$L and $\forall$R rules, the eigenvariable condition requires that the constant symbol $a$ does not occur anywhere outside of $\varphi(a)$ in the upper sequent. It is necessary to ensure that the system is sound, i.e., only derives sequents that are valid. Without this condition, the following would be allowed:

$$\dfrac{\dfrac{\varphi(a) \Rightarrow \varphi(a)}{\exists x\, \varphi(x) \Rightarrow \varphi(a)}\,{}^*\exists\text{L}}{\exists x\, \varphi(x) \Rightarrow \forall x\, \varphi(x)} \forall\text{R} \qquad \dfrac{\dfrac{\varphi(a) \Rightarrow \varphi(a)}{\varphi(a) \Rightarrow \forall x\, \varphi(x)}\,{}^*\forall\text{R}}{\exists x\, \varphi(x) \Rightarrow \forall x\, \varphi(x)} \exists\text{L}$$

However, $\exists x\, \varphi(x) \Rightarrow \forall x\, \varphi(x)$ is not valid.

## 15.4 Structural Rules

We also need a few rules that allow us to rearrange sentences in the left and right side of a sequent. Since the logical rules require that the sentences in the premise which the rule acts upon stand either to the far left or to the far right, we need an "exchange" rule that allows us to move sentences to the right position. It's also important sometimes to be able to combine two identical sentences into one, and to add a sentence on either side.

### Weakening

$$\frac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\,\text{WL} \qquad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \varphi}\,\text{WR}$$

### Contraction

$$\frac{\varphi, \varphi, \Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\,\text{CL} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi}\,\text{CR}$$

### Exchange

$$\frac{\Gamma, \varphi, \psi, \Pi \Rightarrow \Delta}{\Gamma, \psi, \varphi, \Pi \Rightarrow \Delta}\,\text{XL} \qquad \frac{\Gamma \Rightarrow \Delta, \varphi, \psi, \Lambda}{\Gamma \Rightarrow \Delta, \psi, \varphi, \Lambda}\,\text{XR}$$

A series of weakening, contraction, and exchange inferences will often be indicated by double inference lines.

The following rule, called "cut," is not strictly speaking necessary, but makes it a lot easier to reuse and combine derivations.

$$\frac{\Gamma \Rightarrow \Delta, \varphi \qquad \varphi, \Pi \Rightarrow \Lambda}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\,\text{Cut}$$

## 15.5 Derivations

We've said what an initial sequent looks like, and we've given the rules of inference. Derivations in the sequent calculus are inductively generated from

these: each derivation either is an initial sequent on its own, or consists of one or two derivations followed by an inference.

**Definition 15.3** (**LK** derivation)**.** An **LK**-*derivation* of a sequent $S$ is a tree of sequents satisfying the following conditions:

1. The topmost sequents of the tree are initial sequents.
2. The bottommost sequent of the tree is $S$.
3. Every sequent in the tree except $S$ is a premise of a correct application of an inference rule whose conclusion stands directly below that sequent in the tree.

We then say that $S$ is the *end-sequent* of the derivation and that $S$ is *derivable in* **LK** (or **LK**-derivable).

**Example 15.4.** Every initial sequent, e.g., $\chi \Rightarrow \chi$ is a derivation. We can obtain a new derivation from this by applying, say, the WL rule,

$$\frac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta}\,\text{WL}$$

The rule, however, is meant to be general: we can replace the $\varphi$ in the rule with any sentence, e.g., also with $\theta$. If the premise matches our initial sequent $\chi \Rightarrow \chi$, that means that both $\Gamma$ and $\Delta$ are just $\chi$, and the conclusion would then be $\theta, \chi \Rightarrow \chi$. So, the following is a derivation:

$$\frac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\text{WL}$$

We can now apply another rule, say XL, which allows us to switch two sentences on the left. So, the following is also a correct derivation:

$$\dfrac{\dfrac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\text{WL}}{\chi, \theta \Rightarrow \chi}\,\text{XL}$$

In this application of the rule, which was given as

$$\frac{\Gamma, \varphi, \psi, \Pi \Rightarrow \Delta}{\Gamma, \psi, \varphi, \Pi \Rightarrow \Delta,}\,\text{XL}$$

both $\Gamma$ and $\Pi$ were empty, $\Delta$ is $\chi$, and the roles of $\varphi$ and $\psi$ are played by $\theta$ and $\chi$, respectively. In much the same way, we also see that

$$\frac{\theta \Rightarrow \theta}{\chi, \theta \Rightarrow \theta}\,\text{WL}$$

is a derivation. Now we can take these two derivations, and combine them using $\wedge$R. That rule was

$$\dfrac{\Gamma \Rightarrow \Delta, \varphi \qquad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi} \wedge\text{R}$$

In our case, the premises must match the last sequents of the derivations ending in the premises. That means that $\Gamma$ is $\chi, \theta, \Delta$ is empty, $\varphi$ is $\chi$ and $\psi$ is $\theta$. So the conclusion, if the inference should be correct, is $\chi, \theta \Rightarrow \chi \wedge \theta$. Of course, we can also reverse the premises, then $\varphi$ would be $\theta$ and $\psi$ would be $\chi$. So both of the following are correct derivations.

$$\dfrac{\dfrac{\dfrac{\dfrac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\text{WL}}{\chi, \theta \Rightarrow \chi}\,\text{XL} \qquad \dfrac{\theta \Rightarrow \theta}{\chi, \theta \Rightarrow \theta}\,\text{WL}}{\chi, \theta \Rightarrow \chi \wedge \theta}\,\wedge\text{R} \qquad \dfrac{\dfrac{\dfrac{\theta \Rightarrow \theta}{\chi, \theta \Rightarrow \theta}\,\text{WL} \qquad \dfrac{\dfrac{\chi \Rightarrow \chi}{\theta, \chi \Rightarrow \chi}\,\text{WL}}{\chi, \theta \Rightarrow \chi}\,\text{XL}}{\chi, \theta \Rightarrow \theta \wedge \chi}\,\wedge\text{R}$$

## 15.6 Examples of Derivations

**Example 15.5.** Give an **LK**-derivation for the sequent $\varphi \wedge \psi \Rightarrow \varphi$.

We begin by writing the desired end-sequent at the bottom of the derivation.

$$\dfrac{}{\varphi \wedge \psi \Rightarrow \varphi}$$

Next, we need to figure out what kind of inference could have a lower sequent of this form. This could be a structural rule, but it is a good idea to start by looking for a logical rule. The only logical connective occurring in the lower sequent is $\wedge$, so we're looking for an $\wedge$ rule, and since the $\wedge$ symbol occurs in the antecedent, we're looking at the $\wedge$L rule.

$$\dfrac{}{\varphi \wedge \psi \Rightarrow \varphi}\,\wedge\text{L}$$

There are two options for what could have been the upper sequent of the $\wedge$L inference: we could have an upper sequent of $\varphi \Rightarrow \varphi$, or of $\psi \Rightarrow \varphi$. Clearly, $\varphi \Rightarrow \varphi$ is an initial sequent (which is a good thing), while $\psi \Rightarrow \varphi$ is not derivable in general. We fill in the upper sequent:

$$\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi}\,\wedge\text{L}$$

We now have a correct **LK**-derivation of the sequent $\varphi \wedge \psi \Rightarrow \varphi$.

**Example 15.6.** Give an **LK**-derivation for the sequent $\neg\varphi \vee \psi \Rightarrow \varphi \to \psi$.

Begin by writing the desired end-sequent at the bottom of the derivation.

$$\dfrac{}{\neg\varphi \vee \psi \Rightarrow \varphi \to \psi}$$

To find a logical rule that could give us this end-sequent, we look at the logical connectives in the end-sequent: $\neg$, $\vee$, and $\rightarrow$. We only care at the moment about $\vee$ and $\rightarrow$ because they are main operators of sentences in the end-sequent, while $\neg$ is inside the scope of another connective, so we will take care of it later. Our options for logical rules for the final inference are therefore the $\vee$L rule and the $\rightarrow$R rule. We could pick either rule, really, but let's pick the $\rightarrow$R rule (if for no reason other than it allows us to put off splitting into two branches). According to the form of $\rightarrow$R inferences which can yield the lower sequent, this must look like:

$$\dfrac{\dfrac{}{\varphi, \neg\varphi \vee \psi \Rightarrow \psi}}{\neg\varphi \vee \psi \Rightarrow \varphi \rightarrow \psi} \rightarrow\text{R}$$

If we move $\neg\varphi \vee \psi$ to the outside of the antecedent, we can apply the $\vee$L rule. According to the schema, this must split into two upper sequents as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\varphi, \varphi \Rightarrow \psi} \quad \dfrac{}{\psi, \varphi \Rightarrow \psi}}{\neg\varphi \vee \psi, \varphi \Rightarrow \psi} \vee\text{L}}{\varphi, \neg\varphi \vee \psi \Rightarrow \psi} \text{XR}}{\neg\varphi \vee \psi \Rightarrow \varphi \rightarrow \psi} \rightarrow\text{R}$$

Remember that we are trying to wind our way up to initial sequents; we seem to be pretty close! The right branch is just one weakening and one exchange away from an initial sequent and then it is done:

$$\dfrac{\dfrac{\dfrac{\dfrac{}{\neg\varphi, \varphi \Rightarrow \psi} \quad \dfrac{\dfrac{\dfrac{}{\psi \Rightarrow \psi}}{\varphi, \psi \Rightarrow \psi} \text{WL}}{\psi, \varphi \Rightarrow \psi} \text{XL}}{\neg\varphi \vee \psi, \varphi \Rightarrow \psi} \vee\text{L}}{\varphi, \neg\varphi \vee \psi \Rightarrow \psi} \text{XR}}{\neg\varphi \vee \psi \Rightarrow \varphi \rightarrow \psi} \rightarrow\text{R}$$

Now looking at the left branch, the only logical connective in any sentence is the $\neg$ symbol in the antecedent sentences, so we're looking at an instance of the $\neg$L rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow \psi, \varphi}}{\neg\varphi, \varphi \Rightarrow \psi} \neg\text{L} \quad \dfrac{\dfrac{\dfrac{}{\psi \Rightarrow \psi}}{\varphi, \psi \Rightarrow \psi} \text{WL}}{\psi, \varphi \Rightarrow \psi} \text{XL}}{\neg\varphi \vee \psi, \varphi \Rightarrow \psi} \vee\text{L}}{\varphi, \neg\varphi \vee \psi \Rightarrow \psi} \text{XR}}{\neg\varphi \vee \psi \Rightarrow \varphi \rightarrow \psi} \rightarrow\text{R}$$

Similarly to how we finished off the right branch, we are just one weakening and one exchange away from finishing off this left branch as well.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\varphi \Rightarrow \varphi, \psi}\,\mathrm{WR}}{\varphi \Rightarrow \psi, \varphi}\,\mathrm{XR}}{\neg\varphi, \varphi \Rightarrow \psi}\,\neg\mathrm{L} \qquad \dfrac{\dfrac{\dfrac{\psi \Rightarrow \psi}{\varphi, \psi \Rightarrow \psi}\,\mathrm{WL}}{\psi, \varphi \Rightarrow \psi}\,\mathrm{XL}}{\neg\varphi \vee \psi, \varphi \Rightarrow \psi}\,\vee\mathrm{L}}{\varphi, \neg\varphi \vee \psi \Rightarrow \psi}\,\mathrm{XR}}{\neg\varphi \vee \psi \Rightarrow \varphi \to \psi}\,\to\mathrm{R}$$

**Example 15.7.** Give an **LK**-derivation of the sequent $\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)$

Using the techniques from above, we start by writing the desired end-sequent at the bottom.

$$\dfrac{}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}$$

The available main connectives of sentences in the end-sequent are the $\vee$ symbol and the $\neg$ symbol. It would work to apply either the $\vee$L or the $\neg$R rule here, but we start with the $\neg$R rule because it avoids splitting up into two branches for a moment:

$$\dfrac{\dfrac{}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}\,\neg\mathrm{R}$$

Now we have a choice of whether to look at the $\wedge$L or the $\vee$L rule. Let's see what happens when we apply the $\wedge$L rule: we have a choice to start with either the sequent $\varphi, \neg\varphi \vee \psi \Rightarrow$ or the sequent $\psi, \neg\varphi \vee \psi \Rightarrow$ . Since the proof is symmetric with regards to $\varphi$ and $\psi$, let's go with the former:

$$\dfrac{\dfrac{\dfrac{}{\varphi, \neg\varphi \vee \neg\psi \Rightarrow}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}\,\wedge\mathrm{L}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}\,\neg\mathrm{R}$$

Continuing to fill in the derivation, we see that we run into a problem:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\,\neg\mathrm{L} \qquad \dfrac{\dfrac{}{\varphi \Rightarrow \psi}\,?}{\neg\psi, \varphi \Rightarrow}\,\neg\mathrm{L}}{\neg\varphi \vee \neg\psi, \varphi \Rightarrow}\,\vee\mathrm{L}}{\varphi, \neg\varphi \vee \neg\psi \Rightarrow}\,\mathrm{XL}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}\,\wedge\mathrm{L}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)}\,\neg\mathrm{R}$$

The top of the right branch cannot be reduced any further, and it cannot be brought by way of structural inferences to an initial sequent, so this is not the right path to take. So clearly, it was a mistake to apply the $\wedge$L rule above. Going back to what we had before and carrying out the $\vee$L rule instead, we get

$$\dfrac{\dfrac{\dfrac{}{\neg\varphi, \varphi \wedge \psi \Rightarrow} \quad \dfrac{}{\neg\psi, \varphi \wedge \psi \Rightarrow}}{\neg\varphi \vee \neg\psi, \varphi \wedge \psi \Rightarrow} \vee\mathrm{L}}{\dfrac{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)} \neg\mathrm{R}} \mathrm{XL}$$

Completing each branch as we've done before, we get

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi} \wedge\mathrm{L}}{\neg\varphi, \varphi \wedge \psi \Rightarrow} \neg\mathrm{L} \quad \dfrac{\dfrac{\dfrac{\psi \Rightarrow \psi}{\varphi \wedge \psi \Rightarrow \psi} \wedge\mathrm{L}}{\neg\psi, \varphi \wedge \psi \Rightarrow} \neg\mathrm{L}}{\neg\varphi \vee \neg\psi, \varphi \wedge \psi \Rightarrow} \vee\mathrm{L}}{\varphi \wedge \psi, \neg\varphi \vee \neg\psi \Rightarrow} \mathrm{XL}}{\neg\varphi \vee \neg\psi \Rightarrow \neg(\varphi \wedge \psi)} \neg\mathrm{R}$$

(We could have carried out the $\wedge$ rules lower than the $\neg$ rules in these steps and still obtained a correct derivation).

**Example 15.8.** So far we haven't used the contraction rule, but it is sometimes required. Here's an example where that happens. Suppose we want to prove $\Rightarrow A \vee \neg\varphi$. Applying $\vee$R backwards would give us one of these two derivations:

$$\dfrac{\dfrac{}{\Rightarrow \varphi}}{\Rightarrow \varphi \vee \neg\varphi} \vee\mathrm{R} \qquad \dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow}}{\Rightarrow \neg\varphi} \neg\mathrm{R}}{\Rightarrow \varphi \vee \neg\varphi} \vee\mathrm{R}$$

Neither of these of course ends in an initial sequent. The trick is to realize that the contraction rule allows us to combine two copies of a sentence into one—and when we're searching for a proof, i.e., going from bottom to top, we can keep a copy of $\varphi \vee \neg\varphi$ in the premise, e.g.,

$$\dfrac{\dfrac{\dfrac{}{\Rightarrow \varphi \vee \neg\varphi, \varphi}}{\Rightarrow \varphi \vee \neg\varphi, \varphi \vee \neg\varphi} \vee\mathrm{R}}{\Rightarrow \varphi \vee \neg\varphi} \mathrm{CR}$$

Now we can apply $\vee$R a second time, and also get $\neg\varphi$, which leads to a complete derivation.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\Rightarrow \varphi, \neg\varphi} \neg\mathrm{R}}{\Rightarrow \varphi, \varphi \vee \neg\varphi} \vee\mathrm{R}}{\Rightarrow \varphi \vee \neg\varphi, \varphi} \mathrm{XR}}{\Rightarrow \varphi \vee \neg\varphi, \varphi \vee \neg\varphi} \vee\mathrm{R}}{\Rightarrow \varphi \vee \neg\varphi} \mathrm{CR}$$

## 15.7 Derivations with Quantifiers

**Example 15.9.** Give an **LK**-derivation of the sequent $\exists x\, \neg\varphi(x) \Rightarrow \neg\forall x\, \varphi(x)$.

When dealing with quantifiers, we have to make sure not to violate the eigenvariable condition, and sometimes this requires us to play around with the order of carrying out certain inferences. In general, it helps to try and take care of rules subject to the eigenvariable condition first (they will be lower down in the finished proof). Also, it is a good idea to try and look ahead and try to guess what the initial sequent might look like. In our case, it will have to be something like $\varphi(a) \Rightarrow \varphi(a)$. That means that when we are "reversing" the quantifier rules, we will have to pick the same term—what we will call $a$—for both the $\forall$ and the $\exists$ rule. If we picked different terms for each rule, we would end up with something like $\varphi(a) \Rightarrow \varphi(b)$, which, of course, is not derivable.

Starting as usual, we write

$$\dfrac{}{\exists x\, \neg\varphi(x) \;\Rightarrow\; \neg\forall x\, \varphi(x)}$$

We could either carry out the $\exists$L rule or the $\neg$R rule. Since the $\exists$L rule is subject to the eigenvariable condition, it's a good idea to take care of it sooner rather than later, so we'll do that one first.

$$\dfrac{\dfrac{}{\neg\varphi(a) \;\Rightarrow\; \neg\forall x\, \varphi(x)}}{\exists x\, \neg\varphi(x) \;\Rightarrow\; \neg\forall x\, \varphi(x)}\,\exists\text{L}$$

Applying the $\neg$L and $\neg$R rules backwards, we get

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\forall x\, \varphi(x) \;\Rightarrow\; \varphi(a)}}{\neg\varphi(a), \forall x\, \varphi(x) \;\Rightarrow}\,\neg\text{L}}{\forall x\, \varphi(x), \neg\varphi(a) \;\Rightarrow}\,\text{XL}}{\neg\varphi(a) \;\Rightarrow\; \neg\forall x \varphi(x)}\,\neg\text{R}}{\exists x \neg\varphi(x) \;\Rightarrow\; \neg\forall x \varphi(x)}\,\exists\text{L}$$

At this point, our only option is to carry out the $\forall$L rule. Since this rule is not subject to the eigenvariable restriction, we're in the clear. Remember, we want to try and obtain an initial sequent (of the form $\varphi(a) \Rightarrow \varphi(a)$), so we should choose $a$ as our argument for $\varphi$ when we apply the rule.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\varphi(a) \;\Rightarrow\; \varphi(a)}{\forall x\, \varphi(x) \;\Rightarrow\; \varphi(a)}\,\forall\text{L}}{\neg\varphi(a), \forall x\, \varphi(x) \;\Rightarrow}\,\neg\text{L}}{\forall x\, \varphi(x), \neg\varphi(a) \;\Rightarrow}\,\text{XL}}{\neg\varphi(a) \;\Rightarrow\; \neg\forall x\, \varphi(x)}\,\neg\text{R}}{\exists x\, \neg\varphi(x) \;\Rightarrow\; \neg\forall x\, \varphi(x)}\,\exists\text{L}$$

It is important, especially when dealing with quantifiers, to double check at this point that the eigenvariable condition has not been violated. Since the only rule we applied that is subject to the eigenvariable condition was $\exists$L, and the eigenvariable $a$ does not occur in its lower sequent (the end-sequent), this is a correct derivation.

This section collects the definitions of the provability relation and consistency for natural deduction.

## 15.8 Proof-Theoretic Notions

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the derivability or non-derivability of certain sequents. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorem*.

**Definition 15.10** (Theorems)**.** A sentence $\varphi$ is a *theorem* if there is a derivation in **LK** of the sequent $\Rightarrow \varphi$. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 15.11** (Derivability)**.** A sentence $\varphi$ is *derivable from* a set of sentences $\Gamma$, $\Gamma \vdash \varphi$, iff there is a finite subset $\Gamma_0 \subseteq \Gamma$ and a sequence $\Gamma_0'$ of the sentences in $\Gamma_0$ such that **LK** derives $\Gamma_0' \Rightarrow \varphi$. If $\varphi$ is not derivable from $\Gamma$ we write $\Gamma \nvdash \varphi$.

Because of the contraction, weakening, and exchange rules, the order and number of sentences in $\Gamma_0'$ does not matter: if a sequent $\Gamma_0' \Rightarrow \varphi$ is derivable, then so is $\Gamma_0'' \Rightarrow \varphi$ for any $\Gamma_0''$ that contains the same sentences as $\Gamma_0'$. For instance, if $\Gamma_0 = \{\psi, \chi\}$ then both $\Gamma_0' = \langle \psi, \psi, \chi \rangle$ and $\Gamma_0'' = \langle \chi, \chi, \psi \rangle$ are sequences containing just the sentences in $\Gamma_0$. If a sequent containing one is derivable, so is the other, e.g.:

$$\dfrac{\dfrac{\dfrac{\vdots}{\psi, \psi, \chi \Rightarrow \varphi}}{\psi, \chi \Rightarrow \varphi}\,\mathrm{CL}}{\chi, \psi \Rightarrow \varphi}\,\mathrm{XL}}{\chi, \chi, \psi \Rightarrow \varphi}\,\mathrm{WL}$$

From now on we'll say that if $\Gamma_0$ is a finite set of sentences then $\Gamma_0 \Rightarrow \varphi$ is any sequent where the antecedent is a sequence of sentences in $\Gamma_0$ and tacitly include contractions, exchanges, and weakenings if necessary.

**Definition 15.12** (Consistency)**.** A set of sentences $\Gamma$ is *inconsistent* iff there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that **LK** derives $\Gamma_0 \Rightarrow$ . If $\Gamma$ is not inconsistent, i.e., if for every finite $\Gamma_0 \subseteq \Gamma$, **LK** does not derive $\Gamma_0 \Rightarrow$ , we say it is *consistent*.

**Proposition 15.13** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* The initial sequent $\varphi \Rightarrow \varphi$ is derivable, and $\{\varphi\} \subseteq \Gamma$. □

**Proposition 15.14** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Suppose $\Gamma \vdash \varphi$, i.e., there is a finite $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \Rightarrow \varphi$ is derivable. Since $\Gamma \subseteq \Delta$, then $\Gamma_0$ is also a finite subset of $\Delta$. The derivation of $\Gamma_0 \Rightarrow \varphi$ thus also shows $\Delta \vdash \varphi$. □

**Proposition 15.15** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* If $\Gamma \vdash \varphi$, there is a finite $\Gamma_0 \subseteq \Gamma$ and a derivation $\pi_0$ of $\Gamma_0 \Rightarrow \varphi$. If $\{\varphi\} \cup \Delta \vdash \psi$, then for some finite subset $\Delta_0 \subseteq \Delta$, there is a derivation $\pi_1$ of $\varphi, \Delta_0 \Rightarrow \psi$. Consider the following derivation:

$$\dfrac{\begin{array}{cc}\vdots\,\pi_0 & \vdots\,\pi_1\\ \Gamma_0 \Rightarrow \varphi & \varphi, \Delta_0 \Rightarrow \psi\end{array}}{\Gamma_0, \Delta_0 \Rightarrow \psi}\ \text{Cut}$$

Since $\Gamma_0 \cup \Delta_0 \subseteq \Gamma \cup \Delta$, this shows $\Gamma \cup \Delta \vdash \psi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 15.16.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every sentence $\varphi$.*

*Proof.* Exercise. □

**Proposition 15.17** (Compactness)**.** 1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*

2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that the sequent $\Gamma_0 \Rightarrow \varphi$ has a derivation. Consequently, $\Gamma_0 \vdash \varphi$.

2. If $\Gamma$ is inconsistent, there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that **LK** derives $\Gamma_0 \Rightarrow$ . But then $\Gamma_0$ is a finite subset of $\Gamma$ that is inconsistent.

□

## 15.9 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 15.18.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are finite $\Gamma_0$ and $\Gamma_1 \subseteq \Gamma$ such that **LK** derives $\Gamma_0 \Rightarrow \varphi$ and $\varphi, \Gamma_1 \Rightarrow$ . Let the **LK**-derivation of $\Gamma_0 \Rightarrow \varphi$ be $\pi_0$ and the **LK**-derivation of $\Gamma_1, \varphi \Rightarrow$ be $\pi_1$. We can then derive

$$\dfrac{\begin{array}{cc}\vdots\,\pi_0 & \vdots\,\pi_1\\ \Gamma_0 \Rightarrow \varphi & \varphi, \Gamma_1 \Rightarrow\end{array}}{\Gamma_0, \Gamma_1 \Rightarrow}\,\text{Cut}$$

Since $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$, hence $\Gamma$ is inconsistent. □

**Proposition 15.19.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$, i.e., there is a derivation $\pi_0$ of $\Gamma \Rightarrow \varphi$. By adding a $\neg$L rule, we obtain a derivation of $\neg\varphi, \Gamma \Rightarrow$ , i.e., $\Gamma \cup \{\neg\varphi\}$ is inconsistent.

If $\Gamma \cup \{\neg A\}$ is inconsistent, there is a derivation $\pi_1$ of $\neg\varphi, \Gamma \Rightarrow$ . The following is a derivation of $\Gamma \Rightarrow \varphi$:

$$\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\Rightarrow \varphi, \neg\varphi}\,\neg\text{R} \qquad \begin{array}{c}\vdots\,\pi_1\\ \neg\varphi, \Gamma \Rightarrow\end{array}}{\Gamma \Rightarrow \varphi}\,\text{Cut}$$

□

**Proposition 15.20.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* Suppose $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$. Then there is a derivation $\pi$ of a sequent $\Gamma_0 \Rightarrow \varphi$. The sequent $\neg\varphi, \Gamma_0 \Rightarrow$ is also derivable:

$$\dfrac{\begin{array}{cc}\vdots\,\pi & \dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\,\neg\text{L}}{\varphi, \neg\varphi \Rightarrow}\,\text{XL}\\ \Gamma_0 \Rightarrow \varphi & \end{array}}{\Gamma, \neg\varphi \Rightarrow}\,\text{Cut}$$

Since $\neg\varphi \in \Gamma$ and $\Gamma_0 \subseteq \Gamma$, this shows that $\Gamma$ is inconsistent. □

**Proposition 15.21.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are finite sets $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$ and **LK**-derivations $\pi_0$ and $\pi_1$ of $\varphi, \Gamma_0 \Rightarrow$ and $\neg\varphi, \Gamma_1 \Rightarrow$, respectively. We can then derive

$$\dfrac{\dfrac{\begin{array}{c}\vdots\\ \pi_0\end{array}}{\varphi, \Gamma_0 \Rightarrow}}{\Gamma_0 \Rightarrow \neg\varphi}\,\neg\mathrm{R} \qquad \dfrac{\begin{array}{c}\vdots\\ \pi_1\end{array}}{\neg\varphi, \Gamma_1 \Rightarrow}}{\Gamma_0, \Gamma_1 \Rightarrow}\,\mathrm{Cut}$$

Since $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$. Hence $\Gamma$ is inconsistent. □

## 15.10 Derivability and the Propositional Connectives

**Proposition 15.22.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$.*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. Both sequents $\varphi \wedge \psi \Rightarrow \varphi$ and $\varphi \wedge \psi \Rightarrow \psi$ are derivable:

$$\dfrac{\varphi \Rightarrow \varphi}{\varphi \wedge \psi \Rightarrow \varphi}\,\wedge\mathrm{L} \qquad \dfrac{\psi \Rightarrow \psi}{\varphi \wedge \psi \Rightarrow \psi}\,\wedge\mathrm{L}$$

2. Here is a derivation of the sequent $\varphi, \psi \Rightarrow \varphi \wedge \psi$:

$$\dfrac{\varphi \Rightarrow \varphi \qquad \psi \Rightarrow \psi}{\varphi, \psi \Rightarrow \varphi \wedge \psi}\,\wedge\mathrm{R}$$

□

**Proposition 15.23.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. We give a derivation of the sequent $\varphi \vee \psi, \neg\varphi, \neg\psi \Rightarrow$:

$$\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\,\neg\mathrm{L}}{\varphi, \neg\varphi, \neg\psi \Rightarrow} \qquad \dfrac{\dfrac{\psi \Rightarrow \psi}{\neg\psi, \psi \Rightarrow}\,\neg\mathrm{L}}{\psi, \neg\varphi, \neg\psi \Rightarrow}}{\varphi \vee \psi, \neg\varphi, \neg\psi \Rightarrow}\,\vee\mathrm{L}$$

(Recall that double inference lines indicate several weakening, contraction, and exchange inferences.)

2. Both sequents $\varphi \Rightarrow \varphi \vee \psi$ and $\psi \Rightarrow \varphi \vee \psi$ have derivations:

$$\frac{\varphi \Rightarrow \varphi}{\varphi \Rightarrow \varphi \vee \psi} \vee\text{R} \qquad \frac{\psi \Rightarrow \psi}{\psi \Rightarrow \varphi \vee \psi} \vee\text{R}$$

□

**Proposition 15.24.** *1. $\varphi, \varphi \to \psi \vdash \psi$.*

*2. Both $\neg\varphi \vdash \varphi \to \psi$ and $\psi \vdash \varphi \to \psi$.*

*Proof.* 1. The sequent $\varphi \to \psi, \varphi \Rightarrow \psi$ is derivable:

$$\frac{\varphi \Rightarrow \varphi \qquad \psi \Rightarrow \psi}{\varphi \to \psi, \varphi \Rightarrow \psi} \to\text{L}$$

2. Both sequents $\neg\varphi \Rightarrow \varphi \to \psi$ and $\psi \Rightarrow \varphi \to \psi$ are derivable:

$$\dfrac{\dfrac{\dfrac{\dfrac{\varphi \Rightarrow \varphi}{\neg\varphi, \varphi \Rightarrow}\neg\text{L}}{\varphi, \neg\varphi \Rightarrow}\text{XL}}{\varphi, \neg\varphi \Rightarrow \psi}\text{WR}}{\neg\varphi \Rightarrow \varphi \to \psi} \to\text{R} \qquad \dfrac{\dfrac{\psi \Rightarrow \psi}{\varphi, \psi \Rightarrow \psi}\text{WL}}{\psi \Rightarrow \varphi \to \psi} \to\text{R}$$

□

## 15.11 Derivability and the Quantifiers

**Theorem 15.25.** *If $c$ is a constant not occurring in $\Gamma$ or $\varphi(x)$ and $\Gamma \vdash \varphi(c)$, then $\Gamma \vdash \forall x\, \varphi(x)$.*

*Proof.* Let $\pi_0$ be an **LK**-derivation of $\Gamma_0 \Rightarrow \varphi(c)$ for some finite $\Gamma_0 \subseteq \Gamma$. By adding a $\forall$R inference, we obtain a proof of $\Gamma_0 \Rightarrow \forall x\, \varphi(x)$, since $c$ does not occur in $\Gamma$ or $\varphi(x)$ and thus the eigenvariable condition is satisfied. □

**Proposition 15.26.** *1. $\varphi(t) \vdash \exists x\, \varphi(x)$.*

*2. $\forall x\, \varphi(x) \vdash \varphi(t)$.*

*Proof.* 1. The sequent $\varphi(t) \Rightarrow \exists x\, \varphi(x)$ is derivable:

$$\frac{\varphi(t) \Rightarrow \varphi(t)}{\varphi(t) \Rightarrow \exists x\, \varphi(x)} \exists\text{R}$$

2. The sequent $\forall x\, \varphi(x) \Rightarrow \varphi(t)$ is derivable:

$$\frac{\varphi(t) \Rightarrow \varphi(t)}{\forall x\, \varphi(x) \Rightarrow \varphi(t)} \forall\text{L}$$

□

## 15.12 Soundness

A derivation system, such as the sequent calculus, is *sound* if it cannot derive things that do not actually hold. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable $\varphi$ is valid;
2. if a sentence is derivable from some others, it is also a consequence of them;
3. if a set of sentences is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

Because all these proof-theoretic properties are defined via derivability in the sequent calculus of certain sequents, proving (1)–(3) above requires proving something about the semantic properties of derivable sequents. We will first define what it means for a sequent to be *valid*, and then show that every derivable sequent is valid. (1)–(3) then follow as corollaries from this result.

**Definition 15.27.** A structure $\mathfrak{M}$ *satisfies* a sequent $\Gamma \Rightarrow \Delta$ iff either $\mathfrak{M} \nvDash \varphi$ for some $\varphi \in \Gamma$ or $\mathfrak{M} \vDash \varphi$ for some $\varphi \in \Delta$.

A sequent is *valid* iff every structure $\mathfrak{M}$ satisfies it.

**Theorem 15.28** (Soundness)**.** *If* **LK** *derives* $\Theta \Rightarrow \Xi$*, then* $\Theta \Rightarrow \Xi$ *is valid.*

*Proof.* Let $\pi$ be a derivation of $\Theta \Rightarrow \Xi$. We proceed by induction on the number of inferences $n$ in $\pi$.

If the number of inferences is 0, then $\pi$ consists only of an initial sequent. Every initial sequent $\varphi \Rightarrow \varphi$ is obviously valid, since for every $\mathfrak{M}$, either $\mathfrak{M} \nvDash \varphi$ or $\mathfrak{M} \vDash \varphi$.

If the number of inferences is greater than 0, we distinguish cases according to the type of the lowermost inference. By induction hypothesis, we can assume that the premises of that inference are valid, since the number of inferences in the proof of any premise is smaller than $n$.

First, we consider the possible inferences with only one premise.

1. The last inference is a weakening. Then $\Theta \Rightarrow \Xi$ is either $A, \Gamma \Rightarrow \Delta$ (if the last inference is WL) or $\Gamma \Rightarrow \Delta, \varphi$ (if it's WR), and the derivation ends in one of

$$\frac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta\end{array}}{\varphi, \Gamma \Rightarrow \Delta}\,\mathrm{WL} \qquad \frac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta\end{array}}{\Gamma \Rightarrow \Delta, \varphi}\,\mathrm{WR}$$

By induction hypothesis, $\Gamma \Rightarrow \Delta$ is valid, i.e., for every structure $\mathfrak{M}$, either there is some $\chi \in \Gamma$ such that $\mathfrak{M} \nvDash \chi$ or there is some $\chi \in \Delta$ such that $\mathfrak{M} \vDash \chi$.

If $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Gamma$, then $\chi \in \Theta$ as well since $\Theta = \varphi, \Gamma$, and so $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Theta$. Similarly, if $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$, as $\chi \in \Xi$, $\mathfrak{M} \vDash \chi$ for some $\chi \in \Xi$. Consequently, $\Theta \Rightarrow \Xi$ is valid.

2. The last inference is ¬L: Then the premise of the last inference is $\Gamma \Rightarrow \Delta, \varphi$ and the conclusion is $\neg\varphi, \Gamma \Rightarrow \Delta$, i.e., the derivation ends in

$$\frac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi\end{array}}{\neg\varphi, \Gamma \Rightarrow \Delta}\,\neg\mathrm{L}$$

and $\Theta = \neg\varphi, \Gamma$ while $\Xi = \Delta$.

The induction hypothesis tells us that $\Gamma \Rightarrow \Delta, \varphi$ is valid, i.e., for every $\mathfrak{M}$, either (a) for some $\chi \in \Gamma$, $\mathfrak{M} \nvDash \chi$, or (b) for some $\chi \in \Delta$, $\mathfrak{M} \vDash \chi$, or (c) $\mathfrak{M} \vDash \varphi$. We want to show that $\Theta \Rightarrow \Xi$ is also valid. Let $\mathfrak{M}$ be a structure. If (a) holds, then there is $\chi \in \Gamma$ so that $\mathfrak{M} \nvDash \varphi$, but $\varphi \in \Theta$ as well. If (b) holds, there is $\chi \in \Delta$ such that $\mathfrak{M} \vDash \chi$, but $\chi \in \Xi$ as well. Finally, if $\mathfrak{M} \vDash \varphi$, then $\mathfrak{M} \nvDash \neg\varphi$. Since $\neg\varphi \in \Theta$, there is $\chi \in \Theta$ such that $\mathfrak{M} \nvDash \chi$. Consequently, $\Theta \Rightarrow \Xi$ is valid.

3. The last inference is ¬R: Exercise.

4. The last inference is ∧L: There are two variants: $\varphi \wedge \psi$ may be inferred on the left from $\varphi$ or from $\psi$ on the left side of the premise. In the first case, the $\pi$ ends in

$$\frac{\begin{array}{c}\vdots\\ \varphi, \Gamma \Rightarrow \Delta\end{array}}{\varphi \wedge \psi, \Gamma \Rightarrow \Delta}\,\wedge\mathrm{L}$$

and $\Theta = \varphi \wedge \psi, \Gamma$ while $\Xi = \Delta$. Consider a structure $\mathfrak{M}$. Since by induction hypothesis, $\varphi, \Gamma \Rightarrow \Delta$ is valid, (a) $\mathfrak{M} \nvDash \varphi$, (b) $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$. In case (a), $\mathfrak{M} \nvDash \varphi \wedge \psi$, so there

is $\chi \in \Theta$ (namely, $\varphi \wedge \psi$) such that $\mathfrak{M} \nvDash \chi$. In case (b), there is $\chi \in \Gamma$ such that $\mathfrak{M} \nvDash \chi$, and $\chi \in \Theta$ as well. In case (c), there is $\chi \in \Delta$ such that $\mathfrak{M} \vDash \chi$, and $\chi \in \Xi$ as well since $\Xi = \Delta$. So in each case, $\mathfrak{M}$ satisfies $\varphi \wedge \psi, \Gamma \Rightarrow \Delta$. Since $\mathfrak{M}$ was arbitrary, $\Gamma \Rightarrow \Delta$ is valid. The case where $\varphi \wedge \psi$ is inferred from $\psi$ is handled the same, changing $\varphi$ to $\psi$.

5. The last inference is $\vee$R: There are two variants: $\varphi \vee \psi$ may be inferred on the right from $\varphi$ or from $\psi$ on the right side of the premise. In the first case, $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi\end{array}}{\Gamma \Rightarrow \Delta, \varphi \vee \psi} \vee\text{R}$$

Now $\Theta = \Gamma$ and $\Xi = \Delta, \varphi \vee \psi$. Consider a structure $\mathfrak{M}$. Since $\Gamma \Rightarrow \Delta, \varphi$ is valid, (a) $\mathfrak{M} \vDash \varphi$, (b) $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$. In case (a), $\mathfrak{M} \vDash \varphi \vee \psi$. In case (b), there is $\chi \in \Gamma$ such that $\mathfrak{M} \nvDash \chi$. In case (c), there is $\chi \in \Delta$ such that $\mathfrak{M} \vDash \chi$. So in each case, $\mathfrak{M}$ satisfies $\Gamma \Rightarrow \Delta, \varphi \vee \psi$, i.e., $\Theta \Rightarrow \Xi$. Since $\mathfrak{M}$ was arbitrary, $\Theta \Rightarrow \Xi$ is valid. The case where $\varphi \vee \psi$ is inferred from $\psi$ is handled the same, changing $\varphi$ to $\psi$.

6. The last inference is $\rightarrow$R: Then $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \varphi, \Gamma \Rightarrow \Delta, \varphi\end{array}}{\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi} \rightarrow\text{R}$$

Again, the induction hypothesis says that the premise is valid; we want to show that the conclusion is valid as well. Let $\mathfrak{M}$ be arbitrary. Since $\varphi, \Gamma \Rightarrow \Delta, \psi$ is valid, at least one of the following cases obtains: (a) $\mathfrak{M} \nvDash \varphi$, (b) $\mathfrak{M} \vDash \psi$, (c) $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$. In cases (a) and (b), $\mathfrak{M} \vDash \varphi \rightarrow \psi$ and so there is a $\chi \in \Delta, \varphi \rightarrow \psi$ such that $\mathfrak{M} \vDash \chi$. In case (c), for some $\chi \in \Gamma$, $\mathfrak{M} \nvDash \chi$. In case (d), for some $\chi \in \Delta$, $\mathfrak{M} \vDash \chi$. In each case, $\mathfrak{M}$ satisfies $\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi$. Since $\mathfrak{M}$ was arbitrary, $\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi$ is valid.

7. The last inference is $\forall$L: Then there is a formula $\varphi(x)$ and a closed term $t$ such that $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \varphi(t), \Gamma \Rightarrow \Delta\end{array}}{\forall x\, \varphi(x), \Gamma \Rightarrow \Delta}\,\forall\text{L}$$

We want to show that the conclusion $\forall x\, \varphi(x), \Gamma \Rightarrow \Delta$ is valid. Consider a structure $\mathfrak{M}$. Since the premise $\varphi(t), \Gamma \Rightarrow \Delta$ is valid, (a) $\mathfrak{M} \nvDash \varphi(t)$, (b) $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$. In case (a), by Proposition 12.54, if $\mathfrak{M} \vDash \forall x\, \varphi(x)$, then $\mathfrak{M} \vDash \varphi(t)$. Since $\mathfrak{M} \nvDash \varphi(t)$, $\mathfrak{M} \nvDash \forall x\, \varphi(x)$ . In case (b) and (c), $\mathfrak{M}$ also satisfies $\forall x\, \varphi(x), \Gamma \Rightarrow \Delta$. Since $\mathfrak{M}$ was arbitrary, $\forall x\, \varphi(x), \Gamma \Rightarrow \Delta$ is valid.

8. The last inference is $\exists$R: Exercise.

9. The last inference is $\forall$R: Then there is a formula $\varphi(x)$ and a constant symbol $a$ such that $\pi$ ends in

$$\dfrac{\begin{array}{c}\vdots\\ \Gamma \Rightarrow \Delta, \varphi(a)\end{array}}{\Gamma \Rightarrow \Delta, \forall x\, \varphi(x)}\,\forall\text{R}$$

where the eigenvariable condition is satisfied, i.e., $a$ does not occur in $\varphi(x)$, $\Gamma$, or $\Delta$. By induction hypothesis, the premise of the last inference is valid. We have to show that the conclusion is valid as well, i.e., that for any structure $\mathfrak{M}$, (a) $\mathfrak{M} \vDash \forall x\, \varphi(x)$, (b) $\mathfrak{M} \nvDash \chi$ for some $\chi \in \Gamma$, or (c) $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$.

Suppose $\mathfrak{M}$ is an arbitrary structure. If (b) or (c) holds, we are done, so suppose neither holds: for all $\chi \in \Gamma$, $\mathfrak{M} \vDash \chi$, and for all $\chi \in \Delta$, $\mathfrak{M} \nvDash \chi$. We have to show that (a) holds, i.e., $\mathfrak{M} \vDash \forall x\, \varphi(x)$. By Proposition 12.42, if suffices to show that $\mathfrak{M}, s \vDash \varphi(x)$ for all variable assignments $s$. So let $s$ be an arbitrary variable assignment. Consider the structure $\mathfrak{M}'$ which is just like $\mathfrak{M}$ except $a^{\mathfrak{M}'} = s(x)$. By Corollary 12.44, for any $\chi \in \Gamma$, $\mathfrak{M}' \vDash \chi$ since $a$ does not occur in $\Gamma$, and for any $\chi \in \Delta$, $\mathfrak{M}' \nvDash \chi$. But the premise is valid, so $\mathfrak{M}' \vDash \varphi(a)$. By Proposition 12.41, $\mathfrak{M}', s \vDash \varphi(a)$, since $\varphi(a)$ is a sentence. Now $s \sim_x s$ with $s(x) = \mathrm{Val}_s^{\mathfrak{M}'}(a)$, since we've defined $\mathfrak{M}'$ in just this way. So Proposition 12.46 applies, and we get $\mathfrak{M}', s \vDash \varphi(x)$. Since $a$ does not occur in $\varphi(x)$, by Proposition 12.43, $\mathfrak{M}, s \vDash \varphi(x)$. Since $s$ was arbitrary, we've completed the proof that $\mathfrak{M}, s \vDash \varphi(x)$ for all variable assignments.

10. The last inference is $\exists$L: Exercise.

Now let's consider the possible inferences with two premises.

1. The last inference is a cut: then $\pi$ ends in

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \Rightarrow \Delta, \varphi & \varphi, \Pi \Rightarrow \Lambda\end{array}}{\Gamma, \Pi \Rightarrow \Delta, \Lambda}\ \text{Cut}$$

   Let $\mathfrak{M}$ be a structure. By induction hypothesis, the premises are valid, so $\mathfrak{M}$ satisfies both premises. We distinguish two cases: (a) $\mathfrak{M} \nvDash \varphi$ and (b) $\mathfrak{M} \vDash \varphi$. In case (a), in order for $\mathfrak{M}$ to satisfy the left premise, it must satisfy $\Gamma \Rightarrow \Delta$. But then it also satisfies the conclusion. In case (b), in order for $\mathfrak{M}$ to satisfy the right premise, it must satisfy $\Pi \setminus \Lambda$. Again, $\mathfrak{M}$ satisfies the conclusion.

2. The last inference is $\wedge$R. Then $\pi$ ends in

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \Rightarrow \Delta, \varphi & \Gamma \Rightarrow \Delta, \psi\end{array}}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}\ \wedge\text{R}$$

   Consider a structure $\mathfrak{M}$. If $\mathfrak{M}$ satisfies $\Gamma \Rightarrow \Delta$, we are done. So suppose it doesn't. Since $\Gamma \Rightarrow \Delta, \varphi$ is valid by induction hypothesis, $\mathfrak{M} \vDash \varphi$. Similarly, since $\Gamma \Rightarrow \Delta, \psi$ is valid, $\mathfrak{M} \vDash \psi$. But then $\mathfrak{M} \vDash \varphi \wedge \psi$.

3. The last inference is $\vee$L: Exercise.

4. The last inference is $\rightarrow$L. Then $\pi$ ends in

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ \Gamma \Rightarrow \Delta, \varphi & \psi, \Pi \Rightarrow \Lambda\end{array}}{\varphi \rightarrow \psi, \Gamma, \Pi \Rightarrow \Delta, \Lambda}\ \rightarrow\text{L}$$

   Again, consider a structure $\mathfrak{M}$ and suppose $\mathfrak{M}$ doesn't satisfy $\Gamma, \Pi \Rightarrow \Lambda, \Pi$. We have to show that $\mathfrak{M} \nvDash \varphi \rightarrow \psi$. If $\mathfrak{M}$ doesn't satisfy $\Gamma, \Pi \Rightarrow \Lambda, \Pi$, it satisfies neither $\Gamma \Rightarrow \Delta$ nor $\Pi \Rightarrow \Lambda$. Since, $\Gamma \Rightarrow \Delta, \varphi$ is valid, we have $\mathfrak{M} \vDash \varphi$. Since $\psi, \Pi \Rightarrow \Lambda$ is valid, we have $\mathfrak{M} \nvDash \psi$. But then $\mathfrak{M} \nvDash \varphi \rightarrow \psi$, which is what we wanted to show.

□

**Corollary 15.29.** *If $\vdash \varphi$ then $\varphi$ is valid.*

**Corollary 15.30.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* If $\Gamma \vdash \varphi$ then for some finite subset $\Gamma_0 \subseteq \Gamma$, there is a derivation of $\Gamma_0 \Rightarrow \varphi$. By Theorem 15.28, every structure $\mathfrak{M}$ either makes some $\psi \in \Gamma_0$ false or makes $\varphi$ true. Hence, if $\mathfrak{M} \vDash \Gamma$ then also $\mathfrak{M} \vDash \varphi$. □

**Corollary 15.31.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then there is a finite $\Gamma_0 \subseteq \Gamma$ and a derivation of $\Gamma_0 \Rightarrow$ . By Theorem 15.28, $\Gamma_0 \Rightarrow$ is valid. In other words, for every structure $\mathfrak{M}$, there is $\chi \in \Gamma_0$ so that $\mathfrak{M} \nvDash \chi$, and since $\Gamma_0 \subseteq \Gamma$, that $\chi$ is also in $\Gamma$. Thus, no $\mathfrak{M}$ satisfies $\Gamma$, and $\Gamma$ is not satisfiable. □

## 15.13 Derivations with Identity predicate

Derivations with identity predicate require additional initial sequents and inference rules.

**Definition 15.32** (Initial sequents for $=$)**.** If $t$ is a closed term, then $\Rightarrow t = t$ is an initial sequent.

The rules for $=$ are ($t_1$ and $t_2$ are closed terms):

$$\frac{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_1)}{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_2)} = \qquad \frac{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_2)}{t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_1)} =$$

**Example 15.33.** If $s$ and $t$ are closed terms, then $s = t, \varphi(s) \vdash \varphi(t)$:

$$\dfrac{\dfrac{\varphi(s) \Rightarrow \varphi(s)}{s = t, \varphi(s) \Rightarrow \varphi(s)}\,\mathrm{WL}}{s = t, \varphi(s) \Rightarrow \varphi(t)}\,=$$

This may be familiar as the principle of substitutability of identicals, or Leibniz' Law.

**LK** proves that $=$ is symmetric and transitive:

$$\dfrac{\dfrac{\Rightarrow t_1 = t_1}{t_1 = t_2 \Rightarrow t_1 = t_1}\,\mathrm{WL}}{t_1 = t_2 \Rightarrow t_2 = t_1}\,= \qquad \dfrac{\dfrac{\dfrac{t_1 = t_2 \Rightarrow t_1 = t_2}{t_2 = t_3, t_1 = t_2 \Rightarrow t_1 = t_2}\,\mathrm{WL}}{t_2 = t_3, t_1 = t_2 \Rightarrow t_1 = t_3}\,=}{t_1 = t_2, t_2 = t_3 \Rightarrow t_1 = t_3}\,\mathrm{XL}$$

In the proof on the left, the formula $x = t_1$ is our $\varphi(x)$. On the right, we take $\varphi(x)$ to be $t_1 = x$.

## 15.14 Soundness with Identity predicate

**Proposition 15.34. LK** *with initial sequents and rules for identity is sound.*

*Proof.* Initial sequents of the form $\Rightarrow t = t$ are valid, since for every structure $\mathfrak{M}$, $\mathfrak{M} \vDash t = t$. (Note that we assume the term $t$ to be closed, i.e., it contains no variables, so variable assignments are irrelevant).

Suppose the last inference in a derivation is $=$. Then the premise is $t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_1)$ and the conclusion is $t_1 = t_2, \Gamma \Rightarrow \Delta, \varphi(t_2)$. Consider a structure $\mathfrak{M}$. We need to show that the conclusion is valid, i.e., if $\mathfrak{M} \vDash t_1 = t_2$ and $\mathfrak{M} \vDash \Gamma$, then either $\mathfrak{M} \vDash \chi$ for some $\chi \in \Delta$ or $\mathfrak{M} \vDash \varphi(t_2)$.

By induction hypothesis, the premise is valid. This means that if $\mathfrak{M} \vDash t_1 = t_2$ and $\mathfrak{M} \vDash \Gamma$ either (a) for some $\chi \in \Delta$, $\mathfrak{M} \vDash \chi$ or (b) $\mathfrak{M} \vDash \varphi(t_1)$. In case (a) we are done. Consider case (b). Let $s$ be a variable assignment with $s(x) = \mathrm{Val}^{\mathfrak{M}}(t_1)$. By Proposition 12.41, $\mathfrak{M}, s \vDash \varphi(t_1)$. Since $s \sim_x s$, by Proposition 12.46, $\mathfrak{M}, s \vDash \varphi(x)$. since $\mathfrak{M} \vDash t_1 = t_2$, we have $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$, and hence $s(x) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. By applying Proposition 12.46 again, we also have $\mathfrak{M}, s \vDash \varphi(t_2)$. By Proposition 12.41, $\mathfrak{M} \vDash \varphi(t_2)$. □

## Problems

**Problem 15.1.** Give derivations of the following sequents:

1. $\Rightarrow \neg(\varphi \to \psi) \to (\varphi \wedge \neg\psi)$
2. $(\varphi \wedge \psi) \to \chi \Rightarrow (\varphi \to \chi) \vee (\psi \to \chi)$

**Problem 15.2.** Give derivations of the following sequents:

1. $\forall x\,(\varphi(x) \to \psi) \Rightarrow (\exists y\,\varphi(y) \to \psi)$
2. $\exists x\,(\varphi(x) \to \forall y\,\varphi(y))$

**Problem 15.3.** Prove Proposition 15.16

**Problem 15.4.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 15.5.** Complete the proof of Theorem 15.28.

**Problem 15.6.** Give derivations of the following sequents:

1. $\Rightarrow \forall x\,\forall y\,((x = y \wedge \varphi(x)) \to \varphi(y))$
2. $\exists x\,\varphi(x) \wedge \forall y\,\forall z\,((\varphi(y) \wedge \varphi(z)) \to y = z) \Rightarrow \exists x\,(\varphi(x) \wedge \forall y\,(\varphi(y) \to y = x))$

# Chapter 16

# Natural Deduction

This chapter presents a natural deduction system in the style of Gentzen/Prawitz.

To include or exclude material relevant to natural deduction as a proof system, use the "prfND" tag.

## 16.1 Rules and Derivations

Natural deduction systems are meant to closely parallel the informal reasoning used in mathematical proof (hence it is somewhat "natural"). Natural deduction proofs begin with assumptions. Inference rules are then applied. Assumptions are "discharged" by the $\neg$Intro, $\rightarrow$Intro, $\vee$Elim and $\exists$Elim inference rules, and the label of the discharged assumption is placed beside the inference for clarity.

**Definition 16.1** (Initial Formula)**.** An *initial formula* or *assumption* is any formula in the topmost position of any branch.

Derivations in natural deduction are certain trees of sentences, where the topmost sentences are assumptions, and if a sentence stands below one, two, or three other sequents, it must follow correctly by a rule of inference. The sentences at the top of the inference are called the *premises* and the sentence below the *conclusion* of the inference. The rules come in pairs, an introduction and an elimination rule for each logical operator. They introduce a logical operator in the conclusion or remove a logical operator from a premise of the rule. Some of the rules allow an assumption of a certain type to be *discharged*. To indicate which assumption is discharged by which inference, we also assign labels to both the assumption and the inference. This is indicated by writing the assumption as "$[\varphi]^n$".

It is customary to consider rules for all logical operators, even for those (if any) that we consider as defined.

## 16.2 Propositional Rules

**Rules for $\wedge$**

$$\frac{\varphi \qquad \psi}{\varphi \wedge \psi}\ \wedge\text{Intro} \qquad \frac{\varphi \wedge \psi}{\varphi}\ \wedge\text{Elim} \qquad \frac{\varphi \wedge \psi}{\psi}\ \wedge\text{Elim}$$

**Rules for $\vee$**

$$\frac{\varphi}{\varphi \vee \psi}\ \vee\text{Intro} \qquad \frac{\psi}{\varphi \vee \psi}\ \vee\text{Intro} \qquad {}^{n}\frac{\varphi \vee \psi \qquad \begin{array}{c}[\varphi]^n \\ \vdots \\ \chi\end{array} \qquad \begin{array}{c}[\psi]^n \\ \vdots \\ \chi\end{array}}{\chi}\ \vee\text{Elim}$$

**Rules for $\rightarrow$**

$${}^{n}\frac{\begin{array}{c}[\varphi]^n \\ \vdots \\ \psi\end{array}}{\varphi \rightarrow \psi}\ \rightarrow\text{Intro} \qquad \frac{\varphi \rightarrow \psi \qquad \varphi}{\psi}\ \rightarrow\text{Elim}$$

**Rules for $\neg$**

$${}^{n}\frac{\begin{array}{c}[\varphi]^n \\ \vdots \\ \bot\end{array}}{\neg\varphi}\ \neg\text{Intro} \qquad \frac{\neg\varphi \qquad \varphi}{\bot}\ \neg\text{Elim}$$

**Rules for $\bot$**

$$\frac{\bot}{\varphi}\,\bot_I \qquad\qquad \begin{array}{c}[\neg\varphi]^n\\ \vdots\\ \bot\end{array}\;{}^{n}\,\bot_C\;\varphi$$

Note that $\neg$Intro and $\bot_C$ are very similar: The difference is that $\neg$Intro derives a negated sentence $\neg\varphi$ but $\bot_C$ a positive sentence $\varphi$.

## 16.3 Quantifier Rules

**Rules for $\forall$**

$$\frac{\varphi(a)}{\forall x\,\varphi(x)}\,\forall\text{Intro} \qquad\qquad \frac{\forall x\,\varphi(x)}{\varphi(t)}\,\forall\text{Elim}$$

In the rules for $\forall$, $t$ is a ground term (a term that does not contain any variables), and $a$ is a constant symbol which does not occur in the conclusion $\forall x\,\varphi(x)$, or in any assumption which is undischarged in the derivation ending with the premise $\varphi(a)$. We call $a$ the *eigenvariable* of the $\forall$Intro inference.

**Rules for $\exists$**

$$\frac{\varphi(t)}{\exists x\,\varphi(x)}\,\exists\text{Intro} \qquad\qquad {}^{n}\,\frac{\exists x\,\varphi(x) \qquad \begin{array}{c}[\varphi(a)]^n\\ \vdots\\ \chi\end{array}}{\chi}\,\exists\text{Elim}$$

Again, $t$ is a ground term, and $a$ is a constant which does not occur in the premise $\exists x\,\varphi(x)$, in the conclusion $\chi$, or any assumption which is undischarged in the derivations ending with the two premises (other than the assumptions $\varphi(a)$). We call $a$ the *eigenvariable* of the $\exists$Elim inference.

The condition that an eigenvariable neither occur in the premises nor in any assumption that is undischarged in the derivations leading to the premises for the $\forall$Intro or $\exists$Elim inference is called the *eigenvariable condition*.

We use the term "eigenvariable" even though $a$ in the above rules is a constant. This has historical reasons.

In $\exists$Intro and $\forall$Elim there are no restrictions, and the term $t$ can be anything, so we do not have to worry about any conditions. On the other hand, in the $\exists$Elim and $\forall$Intro rules, the eigenvariable condition requires that the constant symbol $a$ does not occur anywhere in the conclusion or in an undischarged assumption. The condition is necessary to ensure that the system is sound, i.e., only derives sentences from undischarged assumptions from which they follow. Without this condition, the following would be allowed:

$$\dfrac{\exists x\, \varphi(x) \qquad \dfrac{[\varphi(a)]^1}{\forall x\, \varphi(x)}\,{*\forall\text{Intro}}}{\forall x\, \varphi(x)}\,\exists\text{Elim}$$

However, $\exists x\, \varphi(x) \nvDash \forall x\, \varphi(x)$.

## 16.4 Derivations

We've said what an assumption is, and we've given the rules of inference. Derivations in natural deduction are inductively generated from these: each derivation either is an assumption on its own, or consists of one, two, or three derivations followed by a correct inference.

**Definition 16.2** (Derivation)**.** A *derivation* of a sentence $\varphi$ from assumptions $\Gamma$ is a tree of sentences satisfying the following conditions:

1. The topmost sentences of the tree are either in $\Gamma$ or are discharged by an inference in the tree.
2. The bottommost sentence of the tree is $\varphi$.
3. Every sentence in the tree except $\varphi$ is a premise of a correct application of an inference rule whose conclusion stands directly below that sentence in the tree.

We then say that $\varphi$ is the *conclusion* of the derivation and that $\varphi$ is *derivable* from $\Gamma$.

**Example 16.3.** Every assumption on its own is a derivation. So, e.g., $\chi$ by itself is a derivation, and so is $\theta$ by itself. We can obtain a new derivation from these by applying, say, the $\wedge$Intro rule,

$$\dfrac{\varphi \qquad \psi}{\varphi \wedge \psi}\,\wedge\text{Intro}$$

These rules are meant to be general: we can replace the $\varphi$ and $\psi$ in it with any sentences, e.g., by $\chi$ and $\theta$. Then the conclusion would be $\chi \wedge \theta$, and so

$$\dfrac{\chi \qquad \theta}{\chi \wedge \theta}\,\wedge\text{Intro}$$

is a correct derivation. Of course, we can also switch the assumptions, so that $\theta$ plays the role of $\varphi$ and $\chi$ that of $\psi$. Thus,

$$\dfrac{\theta \qquad \chi}{\theta \wedge \chi}\,\wedge\text{Intro}$$

is also a correct derivation.

We can now apply another rule, say, $\rightarrow$Intro, which allows us to conclude a conditional and allows us to discharge any assumption that is identical to the conclusion of that conditional. So both of the following would be correct derivations:

$$1\,\dfrac{\dfrac{[\chi]^1 \qquad \theta}{\chi \wedge \theta}\,\wedge\text{Intro}}{\chi \rightarrow (\chi \wedge \theta)}\,\rightarrow\text{Intro} \qquad 1\,\dfrac{\dfrac{\chi \qquad [\theta]^1}{\chi \wedge \theta}\,\wedge\text{Intro}}{\theta \rightarrow (\chi \wedge \theta)}\,\rightarrow\text{Intro}$$

## 16.5 Examples of Derivations

**Example 16.4.** Let's give a derivation of the sentence $(\varphi \wedge \psi) \rightarrow \varphi$.

We begin by writing the desired conclusion at the bottom of the derivation.

$$\dfrac{}{(\varphi \wedge \psi) \rightarrow \varphi}$$

Next, we need to figure out what kind of inference could result in a sentence of this form. The main operator of the conclusion is $\rightarrow$, so we'll try to arrive at the conclusion using the $\rightarrow$Intro rule. It is best to write down the assumptions involved and label the inference rules as you progress, so it is easy to see whether all assumptions have been discharged at the end of the proof.

$$1\,\dfrac{\begin{array}{c}[\varphi \wedge \psi]^1 \\ \vdots \\ \varphi\end{array}}{(\varphi \wedge \psi) \rightarrow \varphi}\,\rightarrow\text{Intro}$$

We now need to fill in the steps from the assumption $\varphi \wedge \psi$ to $\varphi$. Since we only have one connective to deal with, $\wedge$, we must use the $\wedge$ elim rule. This gives us the following proof:

$$1\,\dfrac{\dfrac{[\varphi \wedge \psi]^1}{\varphi}\,\wedge\text{Elim}}{(\varphi \wedge \psi) \rightarrow \varphi}\,\rightarrow\text{Intro}$$

We now have a correct derivation of $(\varphi \wedge \psi) \rightarrow \varphi$.

**Example 16.5.** Now let's give a derivation of $(\neg\varphi \vee \psi) \to (\varphi \to \psi)$.

We begin by writing the desired conclusion at the bottom of the derivation.

$$\frac{}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}$$

To find a logical rule that could give us this conclusion, we look at the logical connectives in the conclusion: $\neg$, $\vee$, and $\to$. We only care at the moment about the first occurence of $\to$ because it is the main operator of the sentence in the end-sequent, while $\neg$, $\vee$ and the second occurence of $\to$ are inside the scope of another connective, so we will take care of those later. We therefore start with the $\to$Intro rule. A correct application must look like this:

$$\dfrac{\begin{array}{c}[\neg\varphi \vee \psi]^1 \\ \vdots \\ \varphi \to \psi\end{array}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,{\scriptstyle 1}\ \to\text{Intro}$$

This leaves us with two possibilities to continue. Either we can keep working from the bottom up and look for another application of the $\to$Intro rule, or we can work from the top down and apply a $\vee$Elim rule. Let us apply the latter. We will use the assumption $\neg\varphi \vee \psi$ as the leftmost premise of $\vee$Elim. For a valid application of $\vee$Elim, the other two premises must be identical to the conclusion $\varphi \to \psi$, but each may be derived in turn from another assumption, namely the two disjuncts of $\neg\varphi \vee \psi$. So our derivation will look like this:

$$\dfrac{\dfrac{\begin{array}{ccc}[\neg\varphi \vee \psi]^1 & \begin{array}{c}[\neg\varphi]^2 \\ \vdots \\ \varphi \to \psi\end{array} & \begin{array}{c}[\psi]^2 \\ \vdots \\ \varphi \to \psi\end{array}\end{array}}{\varphi \to \psi}\,{\scriptstyle 2}\ \vee\text{Elim}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,{\scriptstyle 1}\ \to\text{Intro}$$

In each of the two branches on the right, we want to derive $\varphi \to \psi$, which is best done using $\to$Intro.

$$\dfrac{\dfrac{\begin{array}{ccc}[\neg\varphi \vee \psi]^1 & \dfrac{\begin{array}{c}[\neg\varphi]^2, [\varphi]^3 \\ \vdots \\ \psi\end{array}}{\varphi \to \psi}\,{\scriptstyle 3}\ \to\text{Intro} & \dfrac{\begin{array}{c}[\psi]^2, [\varphi]^4 \\ \vdots \\ \psi\end{array}}{\varphi \to \psi}\,{\scriptstyle 4}\ \to\text{Intro}\end{array}}{\varphi \to \psi}\,{\scriptstyle 2}\ \vee\text{Elim}}{(\neg\varphi \vee \psi) \to (\varphi \to \psi)}\,{\scriptstyle 1}\ \to\text{Intro}$$

For the two missing parts of the derivation, we need derivations of $\psi$ from $\neg\varphi$ and $\varphi$ in the middle, and from $\varphi$ and $\psi$ on the left. Let's take the former first. $\neg\varphi$ and $\varphi$ are the two premises of $\neg$Elim:

$$\dfrac{\dfrac{[\neg\varphi]^2 \qquad [\varphi]^3}{\bot}\,\neg\text{Elim}}{\begin{array}{c}\vdots\\ \psi\end{array}}$$

By using $\bot_I$, we can obtain $\psi$ as a conclusion and complete the branch.

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg\varphi]^2 \qquad [\varphi]^3}{\bot}\,\bot\text{Intro}}{\psi}\,\bot_I}{\varphi\to\psi}\,{}^{3}\!\to\text{Intro} \qquad \dfrac{\begin{array}{c}[\psi]^2, [\varphi]^4\\ \vdots\\ \psi\end{array}}{\varphi\to\psi}\,{}^{4}\!\to\text{Intro}}{\varphi\to\psi}\,{}^{2}\!\vee\text{Elim}}{(\neg\varphi\vee\psi)\to(\varphi\to\psi)}\,{}^{1}\!\to\text{Intro}$$

Let's now look at the rightmost branch. Here it's important to realize that the definition of derivation *allows assumptions to be discharged* but *does not require* them to be. In other words, if we can derive $\psi$ from one of the assumptions $\varphi$ and $\psi$ without using the other, that's ok. And to derive $\psi$ from $\psi$ is trivial: $\psi$ by itself is such a derivation, and no inferences are needed. So we can simply delete the assumtion $\varphi$.

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg\varphi]^2 \qquad [\varphi]^3}{\bot}\,\neg\text{Elim}}{\psi}\,\bot_I}{\varphi\to\psi}\,{}^{3}\!\to\text{Intro} \qquad \dfrac{[\psi]^2}{\varphi\to\psi}\,\to\text{Intro}}{\varphi\to\psi}\,{}^{2}\!\vee\text{Elim}}{(\neg\varphi\vee\psi)\to(\varphi\to\psi)}\,{}^{1}\!\to\text{Intro}$$

Note that in the finished derivation, the rightmost $\to$Intro inference does not actually discharge any assumptions.

**Example 16.6.** So far we have not needed the $\bot_C$ rule. It is special in that it allows us to discharge an assumption that isn't a sub-formula of the conclusion of the rule. It is closely related to the $\bot_I$ rule. In fact, the $\bot_I$ rule is a special case of the $\bot_C$ rule—there is a logic called "intuitionistic logic" in which only $\bot_I$ is allowed. The $\bot_C$ rule is a last resort when nothing else works. For instance, suppose we want to derive $\varphi \vee \neg\varphi$. Our usual strategy would be to attempt to derive $\varphi \vee \neg\varphi$ using $\vee$Intro. But this would require us to derive either $\varphi$ or $\neg\varphi$ from no assumptions, and this can't be done. $\bot_C$ to the rescue!

$$\dfrac{\dfrac{\dfrac{[\neg(\varphi \vee \neg\varphi)]^1}{\vdots}}{\bot}\,1}{\varphi \vee \neg\varphi}\,\bot_C$$

Now we're looking for a derivation of $\bot$ from $\neg(\varphi \vee \neg\varphi)$. Since $\bot$ is the conclusion of $\neg$Elim we might try that:

$$\dfrac{\dfrac{\dfrac{[\neg(\varphi \vee \neg\varphi)]^1}{\vdots}}{\neg\varphi} \qquad \dfrac{[\neg(\varphi \vee \neg\varphi)]^1}{\vdots}}{\bot}\,\neg\text{Elim}}{\varphi}\,1}{\varphi \vee \neg\varphi}\,\bot_C$$

Our strategy for finding a derivation of $\neg\varphi$ calls for an application of $\neg$Intro:

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg(\varphi \vee \neg\varphi)]^1, [\varphi]^2}{\vdots}}{\bot}\,2}{\neg\varphi}\,\neg\text{Intro} \qquad \dfrac{[\neg(\varphi \vee \neg\varphi)]^1}{\vdots}}{\bot}\,\neg\text{Elim}}{\varphi}\,1}{\varphi \vee \neg\varphi}\,\bot_C$$

Here, we can get $\bot$ easily by applying $\neg$Elim to the assumption $\neg(\varphi \vee \neg\varphi)$ and $\varphi \vee \neg\varphi$ which follows from our new assumption $\varphi$ by $\vee$Intro:

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg(\varphi \vee \neg\varphi)]^1 \qquad \dfrac{[\varphi]^2}{\varphi \vee \neg\varphi}\,\vee\text{Intro}}{\bot}\,\neg\text{Elim}}{\neg\varphi}\,2\,\neg\text{Intro} \qquad \dfrac{[\neg(\varphi \vee \neg\varphi)]^1}{\vdots}}{\bot}\,\neg\text{Elim}}{\varphi}\,1}{\varphi \vee \neg\varphi}\,\bot_C$$

On the right side we use the same strategy, except we get $\varphi$ by $\bot_C$:

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg(\varphi \vee \neg\varphi)]^1 \qquad \dfrac{[\varphi]^2}{\varphi \vee \neg\varphi}\,\vee\text{Intro}}{\bot}\,\neg\text{Elim}}{\neg\varphi}\,2\,\neg\text{Intro} \qquad \dfrac{\dfrac{[\neg(\varphi \vee \neg\varphi)]^1 \qquad \dfrac{[\neg\varphi]^3}{\varphi \vee \neg\varphi}\,\vee\text{Intro}}{\bot}\,\neg\text{Elim}}{\varphi}\,3\,\bot_C}{\bot}\,\neg\text{Elim}}{\varphi}\,1}{\varphi \vee \neg\varphi}\,\bot_C$$

## 16.6 Derivations with Quantifiers

**Example 16.7.** When dealing with quantifiers, we have to make sure not to violate the eigenvariable condition, and sometimes this requires us to play around with the order of carrying out certain inferences. In general, it helps to try and take care of rules subject to the eigenvariable condition first (they will be lower down in the finished proof).

Let's see how we'd give a derivation of the formula $\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)$. Starting as usual, we write

$$\frac{}{\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)}$$

We start by writing down what it would take to justify that last step using the $\to$Intro rule.

$$\dfrac{\begin{array}{c}[\exists x\, \neg\varphi(x)]^1 \\ \vdots \\ \neg\forall x\, \varphi(x)\end{array}}{\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)}\ \to\text{Intro}$$

Since there is no obvious rule to apply to $\neg\forall x\, \varphi(x)$, we will proceed by setting up the derivation so we can use the $\exists$Elim rule. Here we must pay attention to the eigenvariable condition, and choose a constant that does not appear in $\exists x\, \varphi(x)$ or any assumptions that it depends on. (Since no constant symbols appear, however, any choice will do fine.)

$$\dfrac{\dfrac{\begin{array}{c}[\exists x\, \neg\varphi(x)]^1 \\ \vdots \\ \neg\forall x\, \varphi(x)\end{array}}{\neg\forall x\, \varphi(x)}\ \exists\text{Elim}^{\,2}}{\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)}\ \to\text{Intro}$$

In order to derive $\neg\forall x\, \varphi(x)$, we will attempt to use the $\neg$Intro rule: this requires that we derive a contradiction, possibly using $\forall x\, \varphi(x)$ as an additional assumption. Of course, this contradiction may involve the assumption $\neg\varphi(a)$ which will be discharged by the $\to$Intro inference. We can set it up as follows:

$$\dfrac{\dfrac{\begin{array}{c}[\exists x\, \neg\varphi(x)]^1 \\ \vdots \\ \neg\forall x\, \varphi(x)\end{array}}{\neg\forall x\, \varphi(x)}\ \exists\text{Elim}^{\,2}}{\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)}\ \to\text{Intro}$$

$$\dfrac{\begin{array}{c}[\neg\varphi(a)]^2, [\forall x\, \varphi(x)]^3 \\ \vdots \\ \bot\end{array}}{\neg\forall x\, \varphi(x)}\ \neg\text{Intro}^{\,3}$$

It looks like we are close to getting a contradiction. The easiest rule to apply is the ∀Elim, which has no eigenvariable conditions. Since we can use any term we want to replace the universally quantified $x$, it makes the most sense to continue using $a$ so we can reach a contradiction.

$$\dfrac{\dfrac{\dfrac{[\exists x\, \neg\varphi(x)]^1 \qquad \dfrac{\dfrac{[\forall x\, \varphi(x)]^3}{\varphi(a)}\,\forall\text{Elim} \qquad [\neg\varphi(a)]^2}{\bot}\,\neg\text{Elim}}{\neg\forall x\, \varphi(x)}\,{}^{3}\neg\text{Intro}}{\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)}\,{}^{2}\exists\text{Elim}\,{}^{1}\to\text{Intro}$$

It is important, especially when dealing with quantifiers, to double check at this point that the eigenvariable condition has not been violated. Since the only rule we applied that is subject to the eigenvariable condition was ∃Elim, and the eigenvariable $a$ does not occur in any assumptions it depends on, this is a correct derivation.

**Example 16.8.** Sometimes we may derive a formula from other formulas. In these cases, we may have undischarged assumptions. It is important to keep track of our assumptions as well as the end goal.

Let's see how we'd give a derivation of the formula $\exists x\, \chi(x, b)$ from the assumptions $\exists x\, (\varphi(x) \wedge \psi(x))$ and $\forall x\, (\psi(x) \to \chi(x, b))$. Starting as usual, we write the conclusion at the bottom.

$$\dfrac{}{\exists x\, \chi(x, b)}$$

We have two premises to work with. To use the first, i.e., try to find a derivation of $\exists x\, \chi(x, b)$ from $\exists x\, (\varphi(x) \wedge \psi(x))$ we would use the ∃Elim rule. Since it has an eigenvariable condition, we will apply that rule first. We get the following:

$$\dfrac{\exists x\, (\varphi(x) \wedge \psi(x)) \qquad \begin{array}{c}[\varphi(a) \wedge \psi(a)]^1 \\ \vdots \\ \exists x\, \chi(x, b)\end{array}}{\exists x\, \chi(x, b)}\,{}^{1}\exists\text{Elim}$$

The two assumptions we are working with share $\psi$. It may be useful at this

point to apply ∧Elim to separate out $\psi(a)$.

$$\dfrac{\dfrac{\exists x\,(\varphi(x) \wedge \psi(x))}{\exists x\,\chi(x,b)}\,1 \qquad \begin{array}{c}[\varphi(a) \wedge \psi(a)]^1 \\ \hline \psi(a) \\ \vdots \\ \exists x\,\chi(x,b)\end{array}}{\exists x\,\chi(x,b)}\,\exists\text{Elim}\;\wedge\text{Elim}$$

The second assumption we have to work with is $\forall x\,(\psi(x) \to \chi(x,b))$. Since there is no eigenvariable condition we can instantiate $x$ with the constant symbol $a$ using ∀Elim to get $\psi(a) \to \chi(a,b)$. We now have both $\psi(a) \to \chi(a,b)$ and $\psi(a)$. Our next move should be a straightforward application of the →Elim rule.

$$\dfrac{\dfrac{\exists x\,(\varphi(x) \wedge \psi(x))}{\exists x\,\chi(x,b)}\,1 \qquad \begin{array}{c}\dfrac{\dfrac{\forall x\,(\psi(x) \to \chi(x,b))}{\psi(a) \to \chi(a,b)}\,\forall\text{Elim} \qquad \dfrac{[\varphi(a) \wedge \psi(a)]^1}{\psi(a)}\,\wedge\text{Elim}}{\chi(a,b)}\,\to\text{Elim} \\ \vdots \\ \exists x\,\chi(x,b)\end{array}}{\exists x\,\chi(x,b)}\,\exists\text{Elim}$$

We are so close! One application of ∃Intro and we have reached our goal.

$$\dfrac{\dfrac{\exists x\,(\varphi(x) \wedge \psi(x))}{\exists x\,\chi(x,b)}\,1 \qquad \dfrac{\dfrac{\dfrac{\forall x\,(\psi(x) \to \chi(x,b))}{\psi(a) \to \chi(a,b)}\,\forall\text{Elim} \qquad \dfrac{[\varphi(a) \wedge \psi(a)]^1}{\psi(a)}\,\wedge\text{Elim}}{\chi(a,b)}\,\to\text{Elim}}{\exists x\,\chi(x,b)}\,\exists\text{Intro}}{\exists x\,\chi(x,b)}\,\exists\text{Elim}$$

Since we ensured at each step that the eigenvariable conditions were not violated, we can be confident that this is a correct derivation.

**Example 16.9.** Give a derivation of the formula $\neg\forall x\,\varphi(x)$ from the assumptions $\forall x\,\varphi(x) \to \exists y\,\psi(y)$ and $\neg\exists y\,\psi(y)$. Starting as usual, we write the target formula at the bottom.

$$\dfrac{}{\neg\forall x\,\varphi(x)}$$

The last line of the derivation is a negation, so let's try using ¬Intro. This will

require that we figure out how to derive a contradiction.

$$\dfrac{\dfrac{\begin{array}{c}[\forall x\, \varphi(x)]^1\\ \vdots\\ \bot\end{array}}{\neg\forall x\, \varphi(x)}\,\neg\text{Intro}}{1}$$

So far so good. We can use ∀Elim but it's not obvious if that will help us get to our goal. Instead, let's use one of our assumptions. $\forall x\, \varphi(x) \to \exists y\, \psi(y)$ together with $\forall x\, \varphi(x)$ will allow us to use the →Elim rule.

$$\dfrac{\dfrac{\begin{array}{c}\forall x\, \varphi(x) \to \exists y\, \psi(y) \qquad [\forall x\, \varphi(x)]^1\\ \exists y\, \psi(y)\\ \vdots\\ \bot\end{array}}{\neg\forall x\, \varphi(x)}\,\neg\text{Intro}}{1}\,\to\text{Elim}$$

We now have one final assumption to work with, and it looks like this will help us reach a contradiction by using ¬Elim.

$$\dfrac{\dfrac{\neg\exists y\, \psi(y) \qquad \dfrac{\forall x\, \varphi(x) \to \exists y\, \psi(y) \qquad [\forall x\, \varphi(x)]^1}{\exists y\, \psi(y)}\,\to\text{Elim}}{\bot}\,\neg\text{Elim}}{\neg\forall x\, \varphi(x)}\,\neg\text{Intro}\;1$$

## 16.7 Proof-Theoretic Notions

This section collects the definitions the provability relation and consistency for natural deduction.

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the derivability or non-derivability of certain sentences from others. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorems*.

**Definition 16.10** (Theorems)**.** A sentence $\varphi$ is a *theorem* if there is a derivation of $\varphi$ in natural deduction in which all assumptions are discharged. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 16.11** (Derivability)**.** A sentence $\varphi$ is *derivable from* a set of sentences $\Gamma$, $\Gamma \vdash \varphi$, if there is a derivation with conclusion $\varphi$ and in which every assumption is either discharged or is in $\Gamma$. If $\varphi$ is not derivable from $\Gamma$ we write $\Gamma \nvdash \varphi$.

**Definition 16.12** (Consistency)**.** A set of sentences $\Gamma$ is *inconsistent* iff $\Gamma \vdash \bot$. If $\Gamma$ is not inconsistent, i.e., if $\Gamma \nvdash \bot$, we say it is *consistent*.

**Proposition 16.13** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* The assumption $\varphi$ by itself is a derivation of $\varphi$ where every undischarged assumption (i.e., $\varphi$) is in $\Gamma$. □

**Proposition 16.14** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Any derivation of $\varphi$ from $\Gamma$ is also a derivation of $\varphi$ from $\Delta$. □

**Proposition 16.15** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* If $\Gamma \vdash \varphi$, there is a derivation $\delta_0$ of $\varphi$ with all undischarged assumptions in $\Gamma$. If $\{\varphi\} \cup \Delta \vdash \psi$, then there is a derivation $\delta_1$ of $\psi$ with all undischarged assumptions in $\{\varphi\} \cup \Delta$. Now consider:

$$\dfrac{\dfrac{\begin{array}{c}\Delta, [\varphi]^1 \\ \vdots\,\delta_1 \\ \psi\end{array}}{\varphi \to \psi}\,1 \to\text{Intro} \qquad \begin{array}{c}\Gamma \\ \vdots\,\delta_0 \\ \varphi\end{array}}{\psi} \to\text{Elim}$$

The undischarged assumptions are now all among $\Gamma \cup \Delta$, so this shows $\Gamma \cup \Delta \vdash \psi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 16.16.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every sentence $\varphi$.*

*Proof.* Exercise. □

**Proposition 16.17** (Compactness)**.** *1. If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*

*2. If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a derivation $\delta$ of $\varphi$ from $\Gamma$. Let $\Gamma_0$ be the set of undischarged assumptions of $\delta$. Since any derivation is finite, $\Gamma_0$ can only contain finitely many sentences. So, $\delta$ is a derivation of $\varphi$ from a finite $\Gamma_0 \subseteq \Gamma$.

2. This is the contrapositive of (1) for the special case $\varphi \equiv \bot$.

□

## 16.8 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 16.18.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* Let the derivation of $\varphi$ from $\Gamma$ be $\delta_1$ and the derivation of $\bot$ from $\Gamma \cup \{\varphi\}$ be $\delta_2$. We can then derive:

$$\dfrac{\dfrac{\Gamma, [\varphi]^1 \quad \vdots\ \delta_2}{\dfrac{\bot}{\neg\varphi}\ 1\ \neg\text{Intro}} \qquad \begin{matrix}\Gamma \\ \vdots\ \delta_1 \\ \varphi\end{matrix}}{\bot}\ \neg\text{Elim}$$

In the new derivation, the assumption $\varphi$ is discharged, so it is a derivation from $\Gamma$. □

**Proposition 16.19.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$, i.e., there is a derivation $\delta_0$ of $\varphi$ from undischarged assumptions $\Gamma$. We obtain a derivation of $\bot$ from $\Gamma \cup \{\neg\varphi\}$ as follows:

$$\dfrac{\neg\varphi \qquad \begin{matrix}\Gamma \\ \vdots\ \delta_0 \\ \varphi\end{matrix}}{\bot}\ \neg\text{Elim}$$

Now assume $\Gamma \cup \{\neg\varphi\}$ is inconsistent, and let $\delta_1$ be the corresponding derivation of $\bot$ from undischarged assumptions in $\Gamma \cup \{\neg\varphi\}$. We obtain a derivation of $\varphi$ from $\Gamma$ alone by using $\bot_C$:

$$\dfrac{\Gamma, [\neg\varphi]^1 \quad \vdots\ \delta_1}{\dfrac{\bot}{\varphi}\ \bot_C}$$

□

**Proposition 16.20.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* Suppose $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$. Then there is a derivation $\delta$ of $\varphi$ from $\Gamma$. Consider this simple application of the $\neg$Elim rule:

$$\dfrac{\neg\varphi \qquad \begin{array}{c}\Gamma\\ \vdots\,\delta\\ \varphi\end{array}}{\bot}\ \neg\text{Elim}$$

Since $\neg\varphi \in \Gamma$, all undischarged assumptions are in $\Gamma$, this shows that $\Gamma \vdash \bot$. □

**Proposition 16.21.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are derivations $\delta_1$ and $\delta_2$ of $\bot$ from $\Gamma \cup \{\varphi\}$ and $\bot$ from $\Gamma \cup \{\neg\varphi\}$, respectively. We can then derive

$$\dfrac{\dfrac{\begin{array}{c}\Gamma, [\neg\varphi]^2\\ \vdots\,\delta_2\\ \bot\end{array}}{\neg\neg\varphi}\ \neg\text{Intro}^{\,2} \qquad \dfrac{\begin{array}{c}\Gamma, [\varphi]^1\\ \vdots\,\delta_1\\ \bot\end{array}}{\neg\varphi}\ \neg\text{Intro}^{\,1}}{\bot}\ \neg\text{Elim}$$

Since the assumptions $\varphi$ and $\neg\varphi$ are discharged, this is a derivation of $\bot$ from $\Gamma$ alone. Hence $\Gamma$ is inconsistent. □

## 16.9 Derivability and the Propositional Connectives

**Proposition 16.22.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. We can derive both

$$\dfrac{\varphi \wedge \psi}{\varphi}\ \wedge\text{Elim} \qquad \dfrac{\varphi \wedge \psi}{\psi}\ \wedge\text{Elim}$$

2. We can derive:

$$\dfrac{\varphi \qquad \psi}{\varphi \wedge \psi}\ \wedge\text{Intro}$$

□

**Proposition 16.23.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

2. *Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. Consider the following derivation:

$$\dfrac{\varphi \vee \psi \qquad \dfrac{\dfrac{\neg\varphi \qquad [\varphi]^1}{\bot}\neg\text{Elim} \qquad \dfrac{\neg\psi \qquad [\psi]^1}{\bot}\neg\text{Elim}}{\bot}\vee\text{Elim}}{1}$$

This is a derivation of $\bot$ from undischarged assumptions $\varphi \vee \psi$, $\neg\varphi$, and $\neg\psi$.

2. We can derive both

$$\dfrac{\varphi}{\varphi \vee \psi}\vee\text{Intro} \qquad \dfrac{\psi}{\varphi \vee \psi}\vee\text{Intro}$$

□

**Proposition 16.24.** *1. $\varphi, \varphi \to \psi \vdash \psi$.*

2. *Both $\neg\varphi \vdash \varphi \to \psi$ and $\psi \vdash \varphi \to \psi$.*

*Proof.* 1. We can derive:

$$\dfrac{\varphi \to \psi \qquad \psi}{\psi}\to\text{Elim}$$

2. This is shown by the following two derivations:

$$\dfrac{\dfrac{\dfrac{\neg\varphi \qquad [\varphi]^1}{\bot}\neg\text{Elim}}{\psi}\bot_I}{\varphi \to \psi}{\scriptstyle 1}\to\text{Intro} \qquad \dfrac{\psi}{\varphi \to \psi}\to\text{Intro}$$

Note that $\to$Intro may, but does not have to, discharge the assumption $\varphi$.

□

## 16.10 Derivability and the Quantifiers

**Theorem 16.25.** *If $c$ is a constant not occurring in $\Gamma$ or $\varphi(x)$ and $\Gamma \vdash \varphi(c)$, then $\Gamma \vdash \forall x\, \varphi(x)$.*

*Proof.* Let $\delta$ be a derivation of $\varphi(c)$ from $\Gamma$. By adding a $\forall$Intro inference, we obtain a proof of $\forall x\, \varphi(x)$. Since $c$ does not occur in $\Gamma$ or $\varphi(x)$, the eigenvariable condition is satisfied. □

**Proposition 16.26.** *1.* $\varphi(t) \vdash \exists x\, \varphi(x)$.

2. $\forall x\, \varphi(x) \vdash \varphi(t)$.

*Proof.* 1. The following is a derivation of $\exists x\, \varphi(x)$ from $\varphi(t)$:

$$\dfrac{\varphi(t)}{\exists x\, \varphi(x)}\ \exists\text{Intro}$$

2. The following is a derivation of $\varphi(t)$ from $\forall x\, \varphi(x)$:

$$\dfrac{\forall x\, \varphi(x)}{\varphi(t)}\ \forall\text{Elim}$$

□

## 16.11 Soundness

A derivation system, such as natural deduction, is *sound* if it cannot derive things that do not actually follow. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable sentence is valid;
2. if a sentence is derivable from some others, it is also a consequence of them;
3. if a set of sentences is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

**Theorem 16.27** (Soundness)**.** *If $\varphi$ is derivable from the undischarged assumptions $\Gamma$, then $\Gamma \vDash \varphi$.*

*Proof.* Let $\delta$ be a derivation of $\varphi$. We proceed by induction on the number of inferences in $\delta$.

For the induction basis we show the claim if the number of inferences is 0. In this case, $\delta$ consists only of an initial formula. Every initial formula $\varphi$ is an undischarged assumption, and as such, any structure $\mathfrak{M}$ that satisfies all of the undischarged assumptions of the proof also satisfies $\varphi$.

Now for the inductive step. Suppose that $\delta$ contains $n$ inferences. The premise(s) of the lowermost inference are derived using sub-derivations, each of which contains fewer than $n$ inferences. We assume the induction hypothesis: The premises of the last inference follow from the undischarged assumptions of the sub-derivations ending in those premises. We have to show that $\varphi$ follows from the undischarged assumptions of the entire proof.

We distinguish cases according to the type of the lowermost inference. First, we consider the possible inferences with only one premise.

1. Suppose that the last inference is ¬Intro: The derivation has the form

$$\dfrac{\begin{array}{c}\Gamma, [\varphi]^n \\ \vdots\,\delta_1 \\ \bot\end{array}}{\neg\varphi}\ {}^{n}\neg\text{Intro}$$

   By inductive hypothesis, $\bot$ follows from the undischarged assumptions $\Gamma \cup \{\varphi\}$ of $\delta_1$. Consider a structure $\mathfrak{M}$. We need to show that, if $\mathfrak{M} \vDash \Gamma$, then $\mathfrak{M} \vDash \neg\varphi$. Suppose for reductio that $\mathfrak{M} \vDash \Gamma$, but $\mathfrak{M} \nvDash \neg\varphi$, i.e., $\mathfrak{M} \vDash \varphi$. This would mean that $\mathfrak{M} \vDash \Gamma \cup \{\varphi\}$. This is contrary to our inductive hypothesis. So, $\mathfrak{M} \vDash \neg\varphi$.

2. The last inference is ∧Elim: There are two variants: $\varphi$ or $\psi$ may be inferred from the premise $\varphi \wedge \psi$. Consider the first case. The derivation $\delta$ looks like this:

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots\,\delta_1 \\ \varphi \wedge \psi\end{array}}{\varphi}\ \wedge\text{Elim}$$

   By inductive hypothesis, $\varphi \wedge \psi$ follows from the undischarged assumptions $\Gamma$ of $\delta_1$. Consider a structure $\mathfrak{M}$. We need to show that, if $\mathfrak{M} \vDash \Gamma$, then $\mathfrak{M} \vDash \varphi$. Suppose $\mathfrak{M} \vDash \Gamma$. By our inductive hypothesis ($\Gamma \vDash \varphi \vee \psi$), we know that $\mathfrak{M} \vDash \varphi \wedge \psi$. By definition, $\mathfrak{M} \vDash \varphi \wedge \psi$ iff $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \vDash \psi$. (The case where $\psi$ is inferred from $\varphi \wedge \psi$ is handled similarly.)

3. The last inference is $\vee$Intro: There are two variants: $\varphi \vee \psi$ may be inferred from the premise $\varphi$ or the premise $\psi$. Consider the first case. The derivation has the form

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \delta_1 \\ \vdots \\ \varphi\end{array}}{\varphi \vee \psi}\ \vee\text{Intro}$$

   By inductive hypothesis, $\varphi$ follows from the undischarged assumptions $\Gamma$ of $\delta_1$. Consider a structure $\mathfrak{M}$. We need to show that, if $\mathfrak{M} \vDash \Gamma$, then $\mathfrak{M} \vDash \varphi \vee \psi$. Suppose $\mathfrak{M} \vDash \Gamma$; then $\mathfrak{M} \vDash \varphi$ since $\Gamma \vDash \varphi$ (the inductive hypothesis). So it must also be the case that $\mathfrak{M} \vDash \varphi \vee \psi$. (The case where $\varphi \vee \psi$ is inferred from $\psi$ is handled similarly.)

4. The last inference is $\to$Intro: $\varphi \to \psi$ is inferred from a subproof with assumption $\varphi$ and conclusion $\psi$, i.e.,

$$\dfrac{\begin{array}{c}\Gamma, [\varphi]^n \\ \vdots \\ \delta_1 \\ \vdots \\ \psi\end{array}}{\varphi \to \psi}\ {}^{n}\ \to\text{Intro}$$

   By inductive hypothesis, $\psi$ follows from the undischarged assumptions of $\delta_1$, i.e., $\Gamma \cup \{\varphi\} \vDash \psi$. Consider a structure $\mathfrak{M}$. The undischarged assumptions of $\delta$ are just $\Gamma$, since $\varphi$ is discharged at the last inference. So we need to show that $\Gamma \vDash \varphi \to \psi$. For reductio, suppose that for some structure $\mathfrak{M}$, $\mathfrak{M} \vDash \Gamma$ but $\mathfrak{M} \nvDash \varphi \to \psi$. So, $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \nvDash \psi$. But by hypothesis, $\psi$ is a consequence of $\Gamma \cup \{\varphi\}$, i.e., $\mathfrak{M} \vDash \psi$, which is a contradiction. So, $\Gamma \vDash \varphi \to \psi$.

5. The last inference is $\bot_I$: Here, $\delta$ ends in

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \delta_1 \\ \vdots \\ \bot\end{array}}{\varphi}\ \bot_I$$

   By induction hypothesis, $\Gamma \vDash \bot$. We have to show that $\Gamma \vDash \varphi$. Suppose not; then for some $\mathfrak{M}$ we have $\mathfrak{M} \vDash \Gamma$ and $\mathfrak{M} \nvDash \varphi$. But we always have $\mathfrak{M} \nvDash \bot$, so this would mean that $\Gamma \nvDash \bot$, contrary to the induction hypothesis.

6. The last inference is $\bot_C$: Exercise.

7. The last inference is $\forall$Intro: Then $\delta$ has the form

$$\dfrac{\begin{array}{c}\Gamma\\ \vdots\,\delta_1\\ \vdots\\ \varphi(a)\end{array}}{\forall x\,\varphi(x)}\,\forall\text{Intro}$$

   The premise $\varphi(a)$ is a consequence of the undischarged assumptions $\Gamma$ by induction hypothesis. Consider some structure, $\mathfrak{M}$, such that $\mathfrak{M} \vDash \Gamma$. We need to show that $\mathfrak{M} \vDash \forall x\,\varphi(x)$. Since $\forall x\,\varphi(x)$ is a sentence, this means we have to show that for every variable assignment $s$, $\mathfrak{M}, s \vDash \varphi(x)$ (Proposition 12.42). Since $\Gamma$ consists entirely of sentences, $\mathfrak{M}, s \vDash \psi$ for all $\psi \in \Gamma$ by Definition 12.35. Let $\mathfrak{M}'$ be like $\mathfrak{M}$ except that $a^{\mathfrak{M}'} = s(x)$. Since $a$ does not occur in $\Gamma$, $\mathfrak{M}' \vDash \Gamma$ by Corollary 12.44. Since $\Gamma \vDash A(a)$, $\mathfrak{M}' \vDash A(a)$. Since $\varphi(a)$ is a sentence, $\mathfrak{M}, s \vDash \varphi(a)$ by Proposition 12.41. $\mathfrak{M}', s \vDash \varphi(x)$ iff $\mathfrak{M}' \vDash \varphi(a)$ by Proposition 12.46 (recall that $\varphi(a)$ is just $\varphi(x)[a/x]$). So, $\mathfrak{M}', s \vDash \varphi(x)$. Since $a$ does not occur in $\varphi(x)$, by Proposition 12.43, $\mathfrak{M}, s \vDash \varphi(x)$. But $s$ was an arbitrary variable assignment, so $\mathfrak{M} \vDash \forall x\,\varphi(x)$.

8. The last inference is $\exists$Intro: Exercise.

9. The last inference is $\forall$Elim: Exercise.

Now let's consider the possible inferences with several premises: $\vee$Elim, $\wedge$Intro, $\to$Elim, and $\exists$Elim.

1. The last inference is $\wedge$Intro. $\varphi \wedge \psi$ is inferred from the premises $\varphi$ and $\psi$ and $\delta$ has the form

$$\dfrac{\begin{array}{cc}\Gamma_1 & \Gamma_2\\ \vdots\,\delta_1 & \vdots\,\delta_2\\ \varphi & \psi\end{array}}{\varphi \wedge \psi}\,\wedge\text{Intro}$$

   By induction hypothesis, $\varphi$ follows from the undischarged assumptions $\Gamma_1$ of $\delta_1$ and $\psi$ follows from the undischarged assumptions $\Gamma_2$ of $\delta_2$. The undischarged assumptions of $\delta$ are $\Gamma_1 \cup \gamma_2$, so we have to show that $\Gamma_1 \cup \Gamma_2 \vDash \varphi \wedge \psi$. Consider a structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$. Since $\mathfrak{M} \vDash \Gamma_1$, it must be the case that $\mathfrak{M} \vDash \varphi$ as $\Gamma_1 \vDash \varphi$, and since $\mathfrak{M} \vDash \Gamma_2$, $\mathfrak{M} \vDash \psi$ since $\Gamma_2 \vDash \psi$. Together, $\mathfrak{M} \vDash \varphi \wedge \psi$.

2. The last inference is $\vee$Elim: Exercise.

3. The last inference is $\rightarrow$Elim. $\psi$ is inferred from the premises $\varphi \rightarrow \psi$ and $\varphi$. The derivation $\delta$ looks like this:

$$\dfrac{\begin{array}{cc}\Gamma_1 & \Gamma_2 \\ \vdots & \vdots \\ \delta_1 & \delta_2 \\ \vdots & \vdots \\ \varphi \rightarrow \psi & \varphi\end{array}}{\psi} \rightarrow\text{Elim}$$

   By induction hypothesis, $\varphi \rightarrow \psi$ follows from the undischarged assumptions $\Gamma_1$ of $\delta_1$ and $\varphi$ follows from the undischarged assumptions $\Gamma_2$ of $\delta_2$. Consider a structure $\mathfrak{M}$. We need to show that, if $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$, then $\mathfrak{M} \vDash \psi$. Suppose $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$. Since $\Gamma_1 \vDash \varphi \rightarrow \psi$, $\mathfrak{M} \vDash \varphi \rightarrow \psi$. Since $\Gamma_2 \vDash \varphi$, we have $\mathfrak{M} \vDash \varphi$. This means that $\mathfrak{M} \vDash \psi$ (For if $\mathfrak{M} \nvDash \psi$, since $\mathfrak{M} \vDash \varphi$, we'd have $\mathfrak{M} \nvDash \varphi \rightarrow \psi$, contradicting $\mathfrak{M} \vDash \varphi \rightarrow \psi$).

4. The last inference is $\neg$Elim: Exercise.

5. The last inference is $\exists$Elim: Exercise.

□

**Corollary 16.28.** *If $\vdash \varphi$, then $\varphi$ is valid.*

**Corollary 16.29.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then $\Gamma \vdash \bot$, i.e., there is a derivation of $\bot$ from undischarged assumptions in $\Gamma$. By Theorem 16.27, any structure $\mathfrak{M}$ that satisfies $\Gamma$ must satisfy $\bot$. Since $\mathfrak{M} \nvDash \bot$ for every structure $\mathfrak{M}$, no $\mathfrak{M}$ can satisfy $\Gamma$, i.e., $\Gamma$ is not satisfiable. □

## 16.12 Derivations with Identity predicate

Derivations with identity predicate require additional inference rules.

$$\dfrac{}{t = t} =\text{Intro} \qquad \dfrac{t_1 = t_2 \qquad \varphi(t_1)}{\varphi(t_2)} =\text{Elim} \qquad \dfrac{t_1 = t_2 \qquad \varphi(t_2)}{\varphi(t_1)} =\text{Elim}$$

In the above rules, $t$, $t_1$, and $t_2$ are closed terms. The $=$Intro rule allows us to derive any identity statement of the form $t = t$ outright, from no assumptions.

**Example 16.30.** If $s$ and $t$ are closed terms, then $\varphi(s), s = t \vdash \varphi(t)$:

$$\dfrac{s = t \qquad \varphi(s)}{\varphi(t)} \,{=}\text{Elim}$$

This may be familiar as the "principle of substitutability of identicals," or Leibniz' Law.

**Example 16.31.** We derive the sentence

$$\forall x\, \forall y\, ((\varphi(x) \wedge \varphi(y)) \to x = y)$$

from the sentence

$$\exists x\, \forall y\, (\varphi(y) \to y = x)$$

We develop the derivation backwards:

$$\dfrac{\dfrac{\dfrac{\exists x\, \forall y\, (\varphi(y) \to y = x) \quad [\varphi(a) \wedge \varphi(b)]^1}{\vdots}}{a = b}}{\dfrac{\dfrac{\dfrac{}{((\varphi(a) \wedge \varphi(b)) \to a = b)}\,{\to}\text{Intro}}{\forall y\, ((\varphi(a) \wedge \varphi(y)) \to a = y)}\,\forall\text{Intro}}{\forall x\, \forall y\, ((\varphi(x) \wedge \varphi(y)) \to x = y)}\,\forall\text{Intro}}\,{}^{1}$$

We'll now have to use the main assumption: since it is an existential formula, we use $\exists$Elim to derive the intermediary conclusion $a = b$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\exists x\, \forall y\, (\varphi(y) \to y = x) \qquad \dfrac{\dfrac{[\forall y\, (\varphi(y) \to y = c)]^2}{[\varphi(a) \wedge \varphi(b)]^1}}{\vdots}}{a = b}}{a = b}\,\exists\text{Elim}}{((\varphi(a) \wedge \varphi(b)) \to a = b)}\,{\to}\text{Intro}}{\forall y\, ((\varphi(a) \wedge \varphi(y)) \to a = y)}\,\forall\text{Intro}}{\forall x\, \forall y\, ((\varphi(x) \wedge \varphi(y)) \to x = y)}\,\forall\text{Intro}\,{}^{2}\,{}^{1}$$

The sub-derivation on the top right is completed by using its assumptions to show that $a = c$ and $b = c$. This requies two separate derivations. The derivation for $a = c$ is as follows:

$$\dfrac{\dfrac{[\forall y\, (\varphi(y) \to y = c)]^2}{\varphi(a) \to a = c}\,\forall\text{Elim} \qquad \dfrac{[\varphi(a) \wedge \varphi(b)]^1}{\varphi(a)}\,\wedge\text{Elim}}{a = c}\,{\to}\text{Elim}$$

From $a = c$ and $b = c$ we derive $a = b$ by $=$Elim.

## 16.13 Soundness with Identity predicate

**Proposition 16.32.** *Natural deduction with rules for $=$ is sound.*

*Proof.* Any formula of the form $t = t$ is valid, since for every structure $\mathfrak{M}$, $\mathfrak{M} \vDash t = t$. (Note that we assume the term $t$ to be ground, i.e., it contains no variables, so variable assignments are irrelevant).

Suppose the last inference in a derivation is =Elim, i.e., the derivation has the following form:

$$\dfrac{\begin{array}{cc} \Gamma_1 & \Gamma_2 \\ \vdots & \vdots \\ \delta_1 & \delta_2 \\ \vdots & \vdots \\ t_1 = t_2 & \varphi(t_1) \end{array}}{\varphi(t_2)} =\text{Elim}$$

The premises $t_1 = t_2$ and $\varphi(t_1)$ are derived from undischarged assumptions $\Gamma_1$ and $\Gamma_2$, respectively. We want to show that $\varphi(t_2)$ follows from $\Gamma_1 \cup \Gamma_2$. Consider a structure $\mathfrak{M}$ with $\mathfrak{M} \vDash \Gamma_1 \cup \Gamma_2$. By induction hypothesis, $\mathfrak{M} \vDash \varphi(t_1)$ and $\mathfrak{M} \vDash t_1 = t_2$. Therefore, $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. Let $s$ be any variable assignment, and $s'$ be the $x$-variant given by $s'(x) = \mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. By Proposition 12.46, $\mathfrak{M}, s \vDash \varphi(t_1)$ iff $\mathfrak{M}, s' \vDash \varphi(x)$ iff $\mathfrak{M}, s \vDash \varphi(t_2)$. Since $\mathfrak{M} \vDash \varphi(t_1)$, we have $\mathfrak{M} \vDash \varphi(t_2)$. □

## Problems

**Problem 16.1.** Give derivations of the following:

1. $\neg(\varphi \to \psi) \to (\varphi \wedge \neg\psi)$
2. $(\varphi \to \chi) \vee (\psi \to \chi)$ from the assumption $(\varphi \wedge \psi) \to \chi$

**Problem 16.2.** Give derivations of the following:

1. $\exists y\, \varphi(y) \to \psi$ from the assumption $\forall x\, (\varphi(x) \to \psi)$
2. $\exists x\, (\varphi(x) \to \forall y\, \varphi(y))$

**Problem 16.3.** Prove Proposition 16.16

**Problem 16.4.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 16.5.** Complete the proof of Theorem 16.27.

**Problem 16.6.** Prove that $=$ is both symmetric and transitive, i.e., give derivations of $\forall x\, \forall y\, (x = y \to y = x)$ and $\forall x\, \forall y\, \forall z((x = y \wedge y = z) \to x = z)$

**Problem 16.7.** Give derivations of the following formulas:

1. $\forall x \, \forall y \, ((x = y \wedge \varphi(x)) \rightarrow \varphi(y))$
2. $\exists x \, \varphi(x) \wedge \forall y \, \forall z \, ((\varphi(y) \wedge \varphi(z)) \rightarrow y = z) \rightarrow \exists x \, (\varphi(x) \wedge \forall y \, (\varphi(y) \rightarrow y = x))$

# Chapter 17

# Tableaux

This chapter presents a signed analytic tableaux system.
To include or exclude material relevant to natural deduction as a proof system, use the "prfTab" tag.

## 17.1 Rules and Tableaux

A tableau is a systematic survey of the possible ways a sentence can be true or false in a structure. The bulding blocks of a tableau are signed formulas: sentences plus a truth value "sign," either $\mathbb{T}$ or $\mathbb{F}$. These signed formulas are arranged in a (downward growing) tree.

**Definition 17.1.** A *signed formula* is a pair consisting of a truth value and a sentence, i.e., either:

$$\mathbb{T}\varphi \text{ or } \mathbb{F}\varphi.$$

Intuitively, we might read $\mathbb{T}\varphi$ as "$\varphi$ might be true" and $\mathbb{F}\varphi$ as "$\varphi$ might be false" (in some structure).

Each signed formula in the tree is either an *assumption* (which are listed at the very top of the tree), or it is obtained from a signed formula above it by one of a number of rules of inference. There are two rules for each possible main operator of the preceding formula, one for the case when the sign is $\mathbb{T}$, and one for the case where the sign is $\mathbb{F}$. Some rules allow the tree to branch, and some only add signed formulas to the branch. A rule may be (and often must be) applied not to the immediately preceding signed formula, but to any signed formula in the branch from the root to the place the rule is applied.

A branch is *closed* when it contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$. A closed tableau is one where every branch is closed. Under the intuitive interpretation, any branch describes a joint possibility, but $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$ are not jointly possible. In other words, if a branch is closed, the possibility it describes has been ruled

out. In particular, that means that a closed tableau rules out all possibilities of simultaneously making every assumption of the form $\mathbb{T}\varphi$ true and every assumption of the form $\mathbb{F}\varphi$ false.

A closed tableau *for* $\varphi$ is a closed tableau with root $\mathbb{F}\varphi$. If such a closed tableau exists, all possibilities for $\varphi$ being false have been ruled out; i.e., $\varphi$ must be true in every structure.

## 17.2 Propositional Rules

**Rules for $\neg$**

$$\frac{\mathbb{T}\neg\varphi}{\mathbb{F}\varphi}\ \neg\mathbb{T} \qquad\qquad \frac{\mathbb{F}\neg\varphi}{\mathbb{T}\varphi}\ \neg\mathbb{F}$$

**Rules for $\wedge$**

$$\frac{\mathbb{T}\varphi\wedge\psi}{\begin{array}{c}\mathbb{T}\varphi\\ \mathbb{T}\psi\end{array}}\ \wedge\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\wedge\psi}{\mathbb{F}\varphi \quad | \quad \mathbb{F}\psi}\ \wedge\mathbb{F}$$

**Rules for $\vee$**

$$\frac{\mathbb{T}\varphi\vee\psi}{\mathbb{T}\varphi \quad | \quad \mathbb{T}\psi}\ \vee\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\vee\psi}{\begin{array}{c}\mathbb{F}\varphi\\ \mathbb{F}\psi\end{array}}\ \vee\mathbb{F}$$

**Rules for $\rightarrow$**

$$\frac{\mathbb{T}\varphi\rightarrow\psi}{\mathbb{F}\varphi \quad | \quad \mathbb{T}\psi}\ \rightarrow\mathbb{T} \qquad\qquad \frac{\mathbb{F}\varphi\rightarrow\psi}{\begin{array}{c}\mathbb{T}\varphi\\ \mathbb{F}\psi\end{array}}\ \rightarrow\mathbb{F}$$

### The Cut Rule

$$\frac{}{\mathbb{T}\varphi \quad | \quad \mathbb{F}\varphi}\ \text{Cut}$$

The Cut rule is not applied "to" a previous signed formula; rather, it allows every branch in a tableau to be split in two, one branch containing $\mathbb{T}\varphi$, the other $\mathbb{F}\varphi$. It is not necessary—any set of signed formulas with a closed tableau has one not using Cut—but it allows us to combine tableaux in a convenient way.

## 17.3 Quantifier Rules

### Rules for $\forall$

$$\frac{\mathbb{T}\forall x\, \varphi(x)}{\mathbb{T}\varphi(t)}\ \forall\mathbb{T} \qquad\qquad \frac{\mathbb{F}\forall x\, \varphi(x)}{\mathbb{F}\varphi(a)}\ \forall\mathbb{F}$$

In $\forall\mathbb{T}$, $t$ is a closed term (i.e., one without variables). In $\forall\mathbb{F}$, $a$ is a constant symbol which must not occur anywhere in the branch above $\forall\mathbb{F}$rule. We call $a$ the *eigenvariable* of the $\forall\mathbb{F}$inference.

### Rules for $\exists$

$$\frac{\mathbb{T}\exists x\, \varphi(x)}{\mathbb{T}\varphi(a)}\ \exists\mathbb{T} \qquad\qquad \frac{\mathbb{F}\exists x\, \varphi(x)}{\mathbb{F}\varphi(t)}\ \exists\mathbb{F}$$

Again, $t$ is a closed term, and $a$ is a constant symbol which does not occur in the branch above the $\exists\mathbb{F}$rule. We call $a$ the *eigenvariable* of the $\exists\mathbb{F}$inference.

The condition that an eigenvariable not occur in the branch above the $\forall\mathbb{F}$or $\exists\mathbb{T}$inference is called the *eigenvariable condition*.

We use the term "eigenvariable" even though $a$ in the above rules is a constant symbol. This has historical reasons.

In $\forall\mathbb{T}$and $\exists\mathbb{F}$there are no restrictions on the term $t$. On the other hand, in the $\exists\mathbb{T}$and $\forall\mathbb{F}$rules, the eigenvariable condition requires that the constant symbol $a$ does not occur anywhere in the branches above the respective inference. It is necessary to ensure that the system is sound. Without this condition, the following would be a closed tableau for $\exists x\, \varphi(x) \to \forall x\, \varphi(x)$:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\,\varphi(x) \to \forall x\,\varphi(x)$ | Assumption |
| 2. | $\mathbb{T}\,\exists x\,\varphi(x)$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,\forall x\,\varphi(x)$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\,\varphi(a)$ | $\exists\mathbb{T}\,2$ |
| 5. | $\mathbb{F}\,\varphi(a)$ | $\forall\mathbb{F}\,3$ |
| | $\otimes$ | |

However, $\exists x\,\varphi(x) \to \forall x\,\varphi(x)$ is not valid.

## 17.4 Tableaux

We've said what an assumption is, and we've given the rules of inference. Tableaux are inductively generated from these: each tableau either is a single branch consisting of one or more assumptions, or it results from a tableau by applying one of the rules of inference on a branch.

**Definition 17.2** (Tableau)**.** A tableau for assumptions $S_1\varphi_1, \ldots, S_n\varphi_n$ (where each $S_i$ is either $\mathbb{T}$ or $\mathbb{F}$) is a tree of signed formulas satisfying the following conditions:

1. The $n$ topmost signed formulas of the tree are $S_i\varphi_i$, one below the other.
2. Every signed formula in the tree that is not one of the assumptions results from a correct application of an inference rule to a signed formula in the branch above it.

A branch of a tableau is *closed* iff it contains both $\mathbb{T}\,\varphi$ and $\mathbb{F}\,\varphi$, and *open* otherwise. A tableau in which every branch is closed is a *closed tableau* (for its set of assumptions). If a tableau is not closed, i.e., if it contains at least one open branch, it is *open*.

**Example 17.3.** Every set of assumptions on its own is a tableau, but it will generally not be closed. (Obviously, it is closed only if the assumptions already contain a pair of signed formulas $\mathbb{T}\,\varphi$ and $\mathbb{F}\,\varphi$.)

From a tableau (open or closed) we can obtain a new, larger one by applying one of the rules of inference to a signed formula $\varphi$ in it. The rule will append one or more signed formulas to the end of any branch containing the occurrence of $\varphi$ to which we apply the rule.

For instance, consider the assumption $\mathbb{T}\,\varphi \wedge \neg\varphi$. Here is the (open) tableau consisting of just that assumption:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\,\varphi \wedge \neg\varphi$ | Assumption |

We obtain a new tableau from it by applying the $\wedge\mathbb{T}$ rule to the assumption. That rule allows us to add two new lines to the tableau, $\mathbb{T}\,\varphi$ and $\mathbb{T}\,\neg\varphi$:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\varphi \wedge \neg\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,1$ |
| 3. | $\mathbb{T}\neg\varphi$ | $\wedge\mathbb{T}\,1$ |

When we write down tableaux, we record the rules we've applied on the right (e.g., $\wedge\mathbb{T}1$ means that the signed formula on that line is the result of applying the $\wedge\mathbb{T}$ rule to the signed formula on line 1). This new tableau now contains additional signed formulas, but to only one ($\mathbb{T}\neg\varphi$) can we apply a rule (in this case, the $\neg\mathbb{T}$ rule). This results in the closed tableau

| | | |
|---|---|---|
| 1. | $\mathbb{T}\varphi \wedge \neg\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,1$ |
| 3. | $\mathbb{T}\neg\varphi$ | $\wedge\mathbb{T}\,1$ |
| 4. | $\mathbb{F}\varphi$ | $\neg\mathbb{T}\,3$ |
| | $\otimes$ | |

## 17.5 Examples of Tableaux

**Example 17.4.** Let's find a closed tableau for the sentence $(\varphi \wedge \psi) \to \varphi$.

We begin by writing the corresponding assumption at the top of the tableau.

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ | Assumption |

There is only one assumption, so only one signed formula to which we can apply a rule. (For every signed formula, there is always at most one rule that can be applied: it's the rule for the corresponding sign and main operator of the sentence.) In this case, this means, we must apply $\to\mathbb{F}$.

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ ✓ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\varphi$ | $\to\mathbb{F}\,1$ |

To keep track of which signed formulas we have applied their corresponding rules to, we write a checkmark next to the sentence. However, *only* write a checkmark if the rule has been applied to all open branches. Once a signed formula has had the corresponding rule applied in every open branch, we will not have to return to it and apply the rule again. In this case, there is only one branch, so the rule only has to be applied once. (Note that checkmarks are only a convenience for constructing tableaux and are not officially part of the syntax of tableaux.)

There is one new signed formula to which we can apply a rule: the $\mathbb{T}\varphi \wedge \psi$ on line 3. Applying the $\wedge\mathbb{T}$ rule results in:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\varphi \wedge \psi) \to \varphi$ ✓ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\psi$ | $\wedge\mathbb{T}\,2$ |
| | $\otimes$ | |

Since the branch now contains both $\mathbb{T}\varphi$ (on line 4) and $\mathbb{F}\varphi$ (on line 3), the branch is closed. Since it is the only branch, the tableau is closed. We have found a closed tableau for $(\varphi \wedge \psi) \to \varphi$.

**Example 17.5.** Now let's find a closed tableau for $(\neg\varphi \vee \psi) \to (\varphi \to \psi)$.

We begin with the corresponding assumption:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ | Assumption |

The one signed formula in this tableau has main operator $\to$ and sign $\mathbb{F}$, so we apply the $\to\mathbb{F}$ rule to it to obtain:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | Assumption |
| 2. | $\mathbb{T}\neg\varphi \vee \psi$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}(\varphi \to \psi)$ | $\to\mathbb{F}\,1$ |

We now have a choice as to whether to apply $\vee\mathbb{T}$ to line 2 or $\to\mathbb{F}$ to line 3. It actually doesn't matter which order we pick, as long as each signed formula has its corresponding rule applied in every branch. So let's pick the first one. The $\vee\mathbb{T}$ rule allows the tableau to branch, and the two conclusions of the rule will be the new signed formulas added to the two new branches. This results in:

| | | |
|---|---|---|
| 1. | $\mathbb{F}(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | Assumption |
| 2. | $\mathbb{T}\neg\varphi \vee \psi$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}(\varphi \to \psi)$ | $\to\mathbb{F}\,1$ |
| | ◿ ◺ | |
| 4. | $\mathbb{T}\neg\varphi$ $\quad$ $\mathbb{T}\psi$ | $\vee\mathbb{T}\,2$ |

We have not applied the $\to\mathbb{F}$ rule to line 3 yet: let's do that now. To save time, we apply it to both branches. Recall that we write a checkmark next to a signed formula only if we have applied the corresponding rule in every open branch. So it's a good idea to apply a rule at the end of every branch that contains the signed formula the rule applies to. That way we won't have to return to that signed formula lower down in the various branches.

| | | | |
|---|---|---|---|
| 1. | $\mathbb{F}\,(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | | Assumption |
| 2. | $\mathbb{T}\neg\varphi \vee \psi$ ✓ | | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,(\varphi \to \psi)$ ✓ | | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\neg\varphi$ | $\mathbb{T}\psi$ | $\vee\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\varphi$ | $\mathbb{T}\varphi$ | $\to\mathbb{F}\,3$ |
| 6. | $\mathbb{F}\psi$ | $\mathbb{F}\psi$ | $\to\mathbb{F}\,3$ |
| | | $\otimes$ | |

The right branch is now closed. On the left branch, we can still apply the $\neg\mathbb{T}$ rule to line 4. This results in $\mathbb{F}\,\varphi$ and closes the left branch:

| | | | |
|---|---|---|---|
| 1. | $\mathbb{F}\,(\neg\varphi \vee \psi) \to (\varphi \to \psi)$ ✓ | | Assumption |
| 2. | $\mathbb{T}\neg\varphi \vee \psi$ ✓ | | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,(\varphi \to \psi)$ ✓ | | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\neg\varphi$ | $\mathbb{T}\psi$ | $\vee\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\varphi$ | $\mathbb{T}\varphi$ | $\to\mathbb{F}\,3$ |
| 6. | $\mathbb{F}\psi$ | $\mathbb{F}\psi$ | $\to\mathbb{F}\,3$ |
| 7. | $\mathbb{F}\varphi$ | $\otimes$ | $\neg\mathbb{T}\,4$ |
| | $\otimes$ | | |

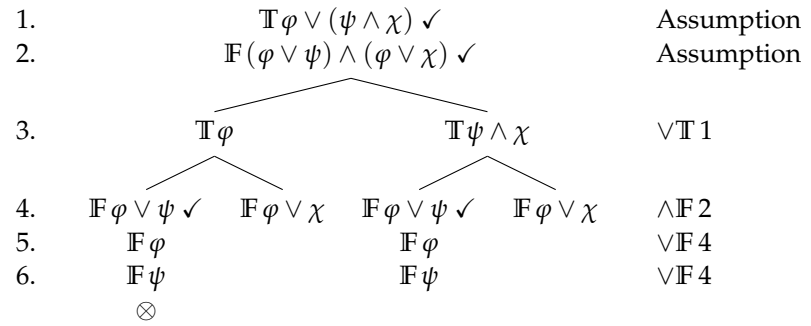
**Example 17.6.** We can give tableaux for any number of signed formulas as assumptions. Often it is also necessary to apply more than one rule that allows branching; and in general a tableau can have any number of branches. For instance, consider a tableau for $\{\mathbb{T}\,\varphi \vee (\psi \wedge \chi), \mathbb{F}\,(\varphi \vee \psi) \wedge (\varphi \vee \chi)\}$. We start by applying the $\vee\mathbb{T}$ to the first assumption:

| | | | |
|---|---|---|---|
| 1. | $\mathbb{T}\,\varphi \vee (\psi \wedge \chi)$ ✓ | | Assumption |
| 2. | $\mathbb{F}\,(\varphi \vee \psi) \wedge (\varphi \vee \chi)$ | | Assumption |
| 3. | $\mathbb{T}\,\varphi$ | $\mathbb{T}\,\psi \wedge \chi$ | $\vee\mathbb{T}\,1$ |

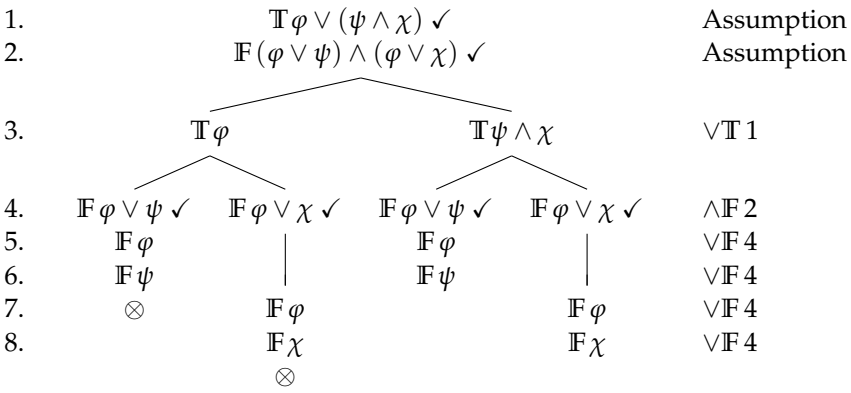
Now we can apply the $\wedge\mathbb{F}$ rule to line 2. We do this on both branches simultaneously, and can therefore check off line 2:

| | | | | | |
|---|---|---|---|---|---|
| 1. | $\mathbb{T}\,\varphi \vee (\psi \wedge \chi)$ ✓ | | | | Assumption |
| 2. | $\mathbb{F}\,(\varphi \vee \psi) \wedge (\varphi \vee \chi)$ ✓ | | | | Assumption |
| 3. | $\mathbb{T}\,\varphi$ | | $\mathbb{T}\,\psi \wedge \chi$ | | $\vee\mathbb{T}\,1$ |
| 4. | $\mathbb{F}\,\varphi \vee \psi$ | $\mathbb{F}\,\varphi \vee \chi$ | $\mathbb{F}\,\varphi \vee \psi$ | $\mathbb{F}\,\varphi \vee \chi$ | $\wedge\mathbb{F}\,2$ |

Now we can apply $\vee\mathbb{F}$ to all the branches containing $\varphi \vee \psi$:

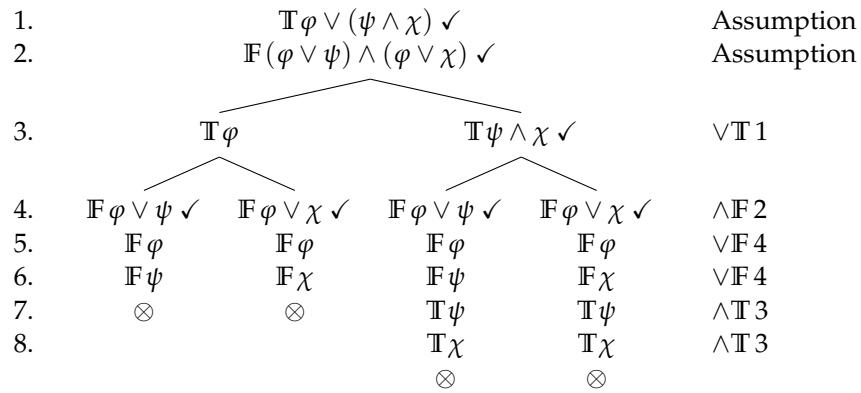



The leftmost branch is now closed. Let's now apply $\vee\mathbb{F}$ to $\varphi \vee \chi$:

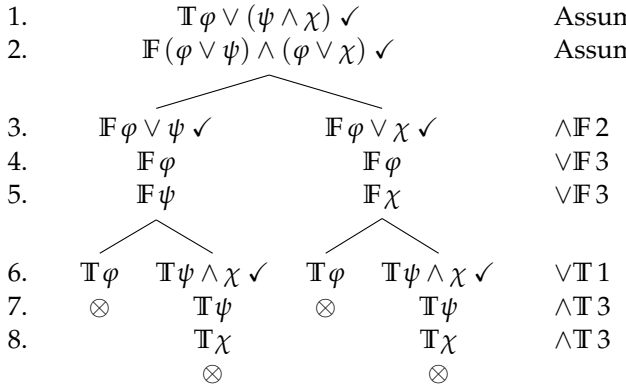



Note that we moved the result of applying $\vee\mathbb{F}$ a second time below for clarity. In this instance it would not have been needed, since the justifications would have been the same.

Two branches remain open, and $\mathbb{T}\psi \wedge \chi$ on line 3 remains unchecked. We apply $\wedge\mathbb{T}$ to it to obtain a closed tableau:




For comparison, here's a closed tableau for the same set of assumptions in which the rules are applied in a different order:




## 17.6 Tableaux with Quantifiers

**Example 17.7.** When dealing with quantifiers, we have to make sure not to violate the eigenvariable condition, and sometimes this requires us to play around with the order of carrying out certain inferences. In general, it helps to try and take care of rules subject to the eigenvariable condition first (they will be higher up in the finished tableau).

Let's see how we'd give a tableau for the sentence $\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)$. Starting as usual, we start by recording the assumption,

1. $\mathbb{F}\exists x\, \neg\varphi(x) \to \neg\forall x\, \varphi(x)$ Assumption

Since the main operator is $\to$, we apply the $\to\mathbb{F}$:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\,\neg\varphi(x) \to \neg\forall x\,\varphi(x)$ ✓ | Assumption |
| 2. | $\mathbb{F}\,\exists x\,\neg\varphi(x)$ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,\neg\forall x\,\varphi(x)$ | $\to\mathbb{F}\,1$ |

The next line to deal with is 2. We use $\exists\mathbb{T}$. This requires a new constant symbol; since no constant symbols yet occur, we can pick any one, say, $a$.

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\,\neg\varphi(x) \to \neg\forall x\,\varphi(x)$ ✓ | Assumption |
| 2. | $\mathbb{T}\,\exists x\,\neg\varphi(x)$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,\neg\forall x\,\varphi(x)$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\,\neg\varphi(a)$ | $\exists\mathbb{T}\,2$ |

Now we apply $\neg\mathbb{F}$ to line 3:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\,\neg\varphi(x) \to \neg\forall x\,\varphi(x)$ ✓ | Assumption |
| 2. | $\mathbb{T}\,\exists x\,\neg\varphi(x)$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,\neg\forall x\,\varphi(x)$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\,\neg\varphi(a)$ | $\exists\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\,\forall x\,\varphi(x)$ | $\neg\mathbb{F}\,3$ |

We obtain a closed tableau by applying $\neg\mathbb{T}$ to line 4, followed by $\forall\mathbb{T}$ to line 5.

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\,\neg\varphi(x) \to \neg\forall x\,\varphi(x)$ ✓ | Assumption |
| 2. | $\mathbb{T}\,\exists x\,\neg\varphi(x)$ ✓ | $\to\mathbb{F}\,1$ |
| 3. | $\mathbb{F}\,\neg\forall x\,\varphi(x)$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{T}\,\neg\varphi(a)$ | $\exists\mathbb{T}\,2$ |
| 5. | $\mathbb{T}\,\forall x\,\varphi(x)$ | $\neg\mathbb{F}\,3$ |
| 6. | $\mathbb{F}\,\varphi(a)$ | $\neg\mathbb{T}\,4$ |
| 7. | $\mathbb{T}\,\varphi(a)$ | $\forall\mathbb{T}\,5$ |
| | $\otimes$ | |

**Example 17.8.** Let's see how we'd give a tableau for the set

$$\mathbb{F}\,\exists x\,\chi(x,b), \mathbb{T}\,\exists x\,(\varphi(x) \wedge \psi(x)), \mathbb{T}\forall x\,(\psi(x) \to \chi(x,b).$$

Starting as usual, we start with the assumptions:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\,\chi(x,b)$ | Assumption |
| 2. | $\mathbb{T}\,\exists x\,(\varphi(x) \wedge \psi(x))$ | Assumption |
| 3. | $\mathbb{T}\forall x\,(\psi(x) \to \chi(x,b)$ | Assumption |

We should always apply a rule with the eigenvariable condition first; in this case that would be $\exists\mathbb{T}$ to line 2. Since the assumptions contain the constant symbol $b$, we have to use a different one; let's pick $a$ again.

| | | |
|---|---|---|
| 1. | $\mathbb{F}\exists x\,\chi(x,b)$ | Assumption |
| 2. | $\mathbb{T}\exists x\,(\varphi(x)\wedge\psi(x))$ ✓ | Assumption |
| 3. | $\mathbb{T}\forall x\,(\psi(x)\to\chi(x,b)$ | Assumption |
| 4. | $\mathbb{T}\varphi(a)\wedge\psi(a)$ | $\exists\mathbb{T}\,2$ |

If we now apply $\exists\mathbb{F}$ to line 1 or $\forall\mathbb{T}$ to line 3, we have to decide with term $t$ to substitute for $x$. Since there is no eigenvariable condition for these rules, we can pick any term we like. In some cases we may even have to apply the rule several times with different $t$s. But as a general rule, it pays to pick one of the terms already occuring in the tableau—in this case, $a$ and $b$—and in this case we can guess that $a$ will be more likely to result in a closed branch.

| | | |
|---|---|---|
| 1. | $\mathbb{F}\exists x\,\chi(x,b)$ | Assumption |
| 2. | $\mathbb{T}\exists x\,(\varphi(x)\wedge\psi(x))$ ✓ | Assumption |
| 3. | $\mathbb{T}\forall x\,(\psi(x)\to\chi(x,b)$ | Assumption |
| 4. | $\mathbb{T}\varphi(a)\wedge\psi(a)$ | $\exists\mathbb{T}\,2$ |
| 5. | $\mathbb{F}\chi(a,b)$ | $\exists\mathbb{F}\,1$ |
| 6. | $\mathbb{T}\psi(a)\to\chi(a,b)$ | $\forall\mathbb{T}\,1$ |

We don't check the signed formulas in lines 1 and 3, since we may have to use them again. Now apply $\wedge\mathbb{T}$ to line 4:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\exists x\,\chi(x,b)$ | Assumption |
| 2. | $\mathbb{T}\exists x\,(\varphi(x)\wedge\psi(x))$ ✓ | Assumption |
| 3. | $\mathbb{T}\forall x\,(\psi(x)\to\chi(x,b)$ | Assumption |
| 4. | $\mathbb{T}\varphi(a)\wedge\psi(a)$ ✓ | $\exists\mathbb{T}\,2$ |
| 5. | $\mathbb{F}\chi(a,b)$ | $\exists\mathbb{F}\,1$ |
| 6. | $\mathbb{T}\psi(a)\to\chi(a,b)$ | $\forall\mathbb{T}\,1$ |
| 7. | $\mathbb{T}\varphi(a)$ | $\wedge\mathbb{T}\,4$ |
| 8. | $\mathbb{T}\psi(a)$ | $\wedge\mathbb{T}\,4$ |

If we now apply $\to\mathbb{T}$ to line 5, the tableau closes:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\exists x\,\chi(x,b)$ | Assumption |
| 2. | $\mathbb{T}\exists x\,(\varphi(x)\wedge\psi(x))$ ✓ | Assumption |
| 3. | $\mathbb{T}\forall x\,(\psi(x)\to\chi(x,b)$ | Assumption |
| 4. | $\mathbb{T}\varphi(a)\wedge\psi(a)$ ✓ | $\exists\mathbb{T}\,2$ |
| 5. | $\mathbb{F}\chi(a,b)$ | $\exists\mathbb{F}\,1$ |
| 6. | $\mathbb{T}\psi(a)\to\chi(a,b)$ ✓ | $\forall\mathbb{T}\,1$ |
| 7. | $\mathbb{T}\varphi(a)$ | $\wedge\mathbb{T}\,4$ |
| 8. | $\mathbb{T}\psi(a)$ | $\wedge\mathbb{T}\,4$ |
| 9. | $\mathbb{F}\psi(a)$ ⊗ \| $\mathbb{T}\chi(a,b)$ ⊗ | $\to\mathbb{T}\,6$ |

**Example 17.9.** We construct a tableau for the set

$$\mathbb{T}\forall x\,\varphi(x), \mathbb{T}\forall x\,\varphi(x) \to \exists y\,\psi(y), \mathbb{T}\neg\exists y\,\psi(y).$$

Starting as usual, we write down the assumptions:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\forall x\,\varphi(x)$ | Assumption |
| 2. | $\mathbb{T}\forall x\,\varphi(x) \to \exists y\,\psi(y)$ | Assumption |
| 3. | $\mathbb{T}\neg\exists y\,\psi(y)$ | Assumption |

We begin by applying the $\neg\mathbb{T}$ rule to line 3. A corollary to the rule "always apply rules with eigenvariable conditions first" is "defer applying quantifier rules without eigenvariable conditions until needed." Also, defer rules that result in a split.

| | | |
|---|---|---|
| 1. | $\mathbb{T}\forall x\,\varphi(x)$ | Assumption |
| 2. | $\mathbb{T}\forall x\,\varphi(x) \to \exists y\,\psi(y)$ | Assumption |
| 3. | $\mathbb{T}\neg\exists y\,\psi(y)$ ✓ | Assumption |
| 4. | $\mathbb{F}\exists y\,\psi(y)$ | $\neg\mathbb{T}\,3$ |

The new line 4 requires $\exists\mathbb{F}$, a quantifier rule without the eigenvariable condition. So we defer this in favor of using $\to\mathbb{T}$ on line 2.

| | | | |
|---|---|---|---|
| 1. | $\mathbb{T}\forall x\,\varphi(x)$ | | Assumption |
| 2. | $\mathbb{T}\forall x\,\varphi(x) \to \exists y\,\psi(y)$ ✓ | | Assumption |
| 3. | $\mathbb{T}\neg\exists y\,\psi(y)$ ✓ | | Assumption |
| 4. | $\mathbb{F}\exists y\,\psi(y)$ | | $\neg\mathbb{T}\,3$ |
| 5. | $\mathbb{F}\forall x\,\varphi(x)$ | $\mathbb{T}\exists y\,\psi(y)$ | $\to\mathbb{T}\,2$ |

Both new signed formulas require rules with eigenvariable conditions, so these should be next:

| | | | |
|---|---|---|---|
| 1. | $\mathbb{T}\forall x\,\varphi(x)$ | | Assumption |
| 2. | $\mathbb{T}\forall x\,\varphi(x) \to \exists y\,\psi(y)$ ✓ | | Assumption |
| 3. | $\mathbb{T}\neg\exists y\,\psi(y)$ ✓ | | Assumption |
| 4. | $\mathbb{F}\exists y\,\psi(y)$ ✓ | | $\neg\mathbb{T}\,3$ |
| 5. | $\mathbb{F}\forall x\,\varphi(x)$ | $\mathbb{T}\exists y\,\psi(y)$ | $\to\mathbb{T}\,2$ |
| 6. | $\mathbb{F}\,\varphi(b)$ | $\mathbb{T}\,\varphi(c)$ | $\forall\mathbb{F}\,5$ |

To close the branches, we have to use the signed formulas on lines 1 and 3. The corresponding rules ($\forall\mathbb{T}$and $\exists\mathbb{F}$) don't have eigenvariable conditions, so we are free to pick whichever terms are suitable. In this case, that's $b$ and $c$, respectively.

| | | | |
|---|---|---|---|
| 1. | $\mathbb{T}\forall x\,\varphi(x)$ | | Assumption |
| 2. | $\mathbb{T}\forall x\,\varphi(x) \to \exists y\,\psi(y)$ ✓ | | Assumption |
| 3. | $\mathbb{T}\neg\exists y\,\psi(y)$ ✓ | | Assumption |
| 4. | $\mathbb{F}\,\exists y\,\psi(y)$ ✓ | | $\neg\mathbb{T}\,3$ |
| 5. | $\mathbb{F}\,\forall x\,\varphi(x)$ | $\mathbb{T}\exists y\,\psi(y)$ | $\to\mathbb{T}\,2$ |
| 6. | $\mathbb{F}\,\varphi(b)$ | $\mathbb{T}\,\psi(c)$ | $\forall\mathbb{F}\,5$; $\exists\mathbb{T}\,5$ |
| 7. | $\mathbb{T}\,\varphi(b)$ | $\mathbb{F}\,\psi(c)$ | $\forall\mathbb{T}\,1$; $\exists\mathbb{F}\,4$ |

## 17.7 Proof-Theoretic Notions

This section collects the definitions of the provability relation and consistency for tableaux.

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the existence of certain closed tableaux. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorems*.

**Definition 17.10** (Theorems)**.** A sentence $\varphi$ is a *theorem* if there is a closed tableau for $\mathbb{F}\,\varphi$. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 17.11** (Derivability)**.** A sentence $\varphi$ is *derivable from* a set of sentences $\Gamma$, $\Gamma \vdash \varphi$, iff there is a finite set $\{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ and a closed tableau for the set

$$\{\mathbb{F}\,\varphi, \mathbb{T}\,\psi_1, \ldots, \mathbb{T}\,\psi_n, \}$$

If $\varphi$ is not derivable from $\Gamma$ we write $\Gamma \nvdash \varphi$.

**Definition 17.12** (Consistency)**.** A set of sentences $\Gamma$ is *inconsistent* iff there is a finite set $\{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ and a closed tableau for the set

$$\{\mathbb{T}\,\psi_1, \ldots, \mathbb{T}\,\psi_n, \}.$$

If $\Gamma$ is not inconsistent, we say it is *consistent*.

**Proposition 17.13** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* If $\varphi \in \Gamma$, $\{\varphi\}$ is a finite subset of $\Gamma$ and the tableau

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\varphi$ | Assumption |
| 2. | $\mathbb{T}\,\varphi$ | Assumption |
| | $\otimes$ | |

is closed. □

**Proposition 17.14** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Any finite subset of $\Gamma$ is also a finite subset of $\Delta$. □

**Proposition 17.15** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* If $\{\varphi\} \cup \Delta \vdash \psi$, then there is a finite subset $\Delta_0 = \{\chi_1, \ldots, \chi_n\} \subseteq \Delta$ such that

$$\{\mathbb{F}\psi, \mathbb{T}\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_n\}$$

has a closed tableau. If $\Gamma \vdash \varphi$ then there are $\theta_1, \ldots, \theta_m$ such that

$$\{\mathbb{F}\varphi, \mathbb{T}\theta_1, \ldots, \mathbb{T}\theta_m\}$$

has a closed tableau.

Now consider the tableau with assumptions

$$\mathbb{F}\psi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_n, \mathbb{T}\theta_1, \ldots, \mathbb{T}\theta_m.$$

Apply the Cut rule on $\varphi$. This generates two branches, one has $\mathbb{T}\varphi$ in it, the other $\mathbb{F}\varphi$. Thus, on the one branch, all of

$$\{\mathbb{F}\psi, \mathbb{T}\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_n\}$$

are available. Since there is a closed tableau for these assumptions, we can attach it to that branch; every branch through $\mathbb{T}\varphi_1$ closes. On the other branch, all of

$$\{\mathbb{F}\varphi, \mathbb{T}\theta_1, \ldots, \mathbb{T}\theta_m\}$$

are available, so we can also complete the other side to obtain a closed tableau. This shows $\Gamma \cup \Delta \vdash \psi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 17.16.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every sentence $\varphi$.*

*Proof.* Exercise. □

**Proposition 17.17** (Compactness)**.**

1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*
2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a finite subset $\Gamma_0 = \{\psi_1, \ldots, \psi_n\}$ and a closed tableau for

$$\mathbb{F}\varphi, \mathbb{T}\psi_1, \cdots \mathbb{T}\psi_n$$

This tableau also shows $\Gamma_0 \vdash \varphi$.

2. If $\Gamma$ is inconsistent, then for some finite subset $\Gamma_0 = \{\psi_1, \ldots, \psi_n\}$ there is a closed tableau for

$$\mathbb{T}\psi_1, \cdots \mathbb{T}\psi_n$$

This closed tableau shows that $\Gamma_0$ is inconsistent.

□

## 17.8 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 17.18.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* There are finite $\Gamma_0 = \{\psi_1, \ldots, \psi_n\}$ and $\Gamma_1 = \{\chi_1, \ldots, \chi_n\} \subseteq \Gamma$ such that

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$
$$\{\mathbb{T}\neg\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_m\}$$

have closed tableaux. Using the Cut rule on $\varphi$ we can combine these into a single closed tableau that shows $\Gamma_0 \cup \Gamma_1$ is inconsistent. Since $\Gamma_0 \subseteq \Gamma$ and $\Gamma_1 \subseteq \Gamma$, $\Gamma_0 \cup \Gamma_1 \subseteq \Gamma$, hence $\Gamma$ is inconsistent. □

**Proposition 17.19.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$, i.e., there is a closed tableau for

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

Using the $\neg\mathbb{T}$ rule, this can be turned into a closed tableau for

$$\{\mathbb{T}\neg\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}.$$

On the other hand, if there is a closed tableau for the latter, we can turn it into a closed tableau of the former by removing every formula that results from $\neg\mathbb{T}$applied to the first assumption $\mathbb{T}\neg\varphi$ as well as that assumption, and adding the assumption $\mathbb{F}\varphi$. For if a branch was closed before because it contained the conclusion of $\neg\mathbb{T}$applied to $\mathbb{T}\neg\varphi$, i.e., $\mathbb{F}\varphi$, the corresponding branch in the new tableau is also closed. If a branch in the old tableau was closed because it contained the assumption $\mathbb{T}\neg\varphi$ as well as $\mathbb{F}\neg\varphi$ we can turn it into a closed branch by applying $\neg\mathbb{F}$ to $\mathbb{F}\neg\varphi$ to obtain $\mathbb{T}\varphi$. This closes the branch since we added $\mathbb{F}\varphi$ as an assumption. □

**Proposition 17.20.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* Suppose $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$. Then there are $\psi_1, \ldots, \psi_n \in \Gamma$ such that

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$

has a closed tableau. Replace the assumption $\mathbb{F}\varphi$by $\mathbb{T}\neg\varphi$, and insert the conclusion of $\neg\mathbb{T}$applied to $\mathbb{F}\varphi$after the assumptions. Any sentence in the tableau justified by appeal to line 1 in the old tableau is now justified by appeal to line $n+1$. So if the old tableau was closed, the new one is. It shows that $\Gamma$ is inconsistent, since all assumptions are in $\Gamma$. □

**Proposition 17.21.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* If there are $\psi_1, \ldots, \psi_n \in \Gamma$ and $\chi_1, \ldots, \chi_m \in \Gamma$ such that

$$\{\mathbb{T}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$$
$$\{\mathbb{T}\neg\varphi, \mathbb{T}\chi_1, \ldots, \mathbb{T}\chi_m\}$$

both have closed tableaux, we can construct a tableau that shows that $\Gamma$ is inconsistent by using as assumptions $\mathbb{T}\psi_1$, …, $\mathbb{T}\psi_n$ together with $\mathbb{T}\chi_1$, …, $\mathbb{T}\chi_m$, followed by an application of the Cut rule, yielding two branches, one starting with $\mathbb{T}\varphi$, the other with $\mathbb{F}\varphi$. Add on the part below the assumptions of the first tableau on the left side. Here, every rule application is still correct, and every branch closes. On the right side, add the part below the assumptions of the seond tableau, with the results of any applications of $\neg\mathbb{T}$ to $\mathbb{T}\neg\varphi$ removed.

For if a branch was closed before because it contained the conclusion of $\neg\mathbb{T}$applied to $\mathbb{T}\neg\varphi$, i.e., $\mathbb{F}\varphi$, as well as $\mathbb{F}\varphi$, the corresponding branch in the new tableau is also closed. If a branch in the old tableau was closed because it contained the assumption $\mathbb{T}\neg\varphi$ as well as $\mathbb{F}\neg\varphi$ we can turn it into a closed branch by applying $\neg\mathbb{F}$ to $\mathbb{F}\neg\varphi$ to obtain $\mathbb{T}\varphi$. □

## 17.9 Derivability and the Propositional Connectives

**Proposition 17.22.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$.*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. Both $\{\mathbb{F}\varphi, \mathbb{T}\varphi \wedge \psi\}$ and $\{\mathbb{F}\psi, \mathbb{T}\varphi \wedge \psi\}$ have closed tableaux

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi$ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}$ 2 |
| 4. | $\mathbb{T}\psi$ | $\wedge\mathbb{T}$ 2 |
| | $\otimes$ | |

| | | |
|---|---|---|
| 1. | $\mathbb{F}\psi$ | Assumption |
| 2. | $\mathbb{T}\varphi \wedge \psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\wedge\mathbb{T}\,2$ |
| 4. | $\mathbb{T}\psi$ | $\wedge\mathbb{T}\,2$ |
| | $\otimes$ | |

2. Here is a closed tableau for $\{\mathbb{T}\varphi, \mathbb{T}\psi, \mathbb{F}\varphi \wedge \psi\}$:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \wedge \psi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | Assumption |
| 3. | $\mathbb{T}\psi$ | Assumption |
| 4. | $\mathbb{F}\varphi$ $\quad$ $\mathbb{F}\psi$ | $\wedge\mathbb{F}\,1$ |
| | $\otimes$ $\quad$ $\otimes$ | |

□

**Proposition 17.23.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. We give a closed tableau of $\{\mathbb{T}\varphi \vee \psi, \mathbb{T}\neg\varphi, \mathbb{T}\neg\psi\}$:

| | | |
|---|---|---|
| 1. | $\mathbb{T}\varphi \vee \psi$ | Assumption |
| 2. | $\mathbb{T}\neg\varphi$ | Assumption |
| 3. | $\mathbb{T}\neg\psi$ | Assumption |
| 4. | $\mathbb{F}\varphi$ | $\neg\mathbb{T}\,2$ |
| 5. | $\mathbb{F}\psi$ | $\neg\mathbb{T}\,3$ |
| 6. | $\mathbb{T}\varphi$ $\quad$ $\mathbb{T}\psi$ | $\vee\mathbb{T}\,1$ |
| | $\otimes$ $\quad$ $\otimes$ | |

2. Both $\{\mathbb{F}\varphi \vee \psi, \mathbb{T}\varphi\}$ and $\{\mathbb{F}\varphi \vee \psi, \mathbb{T}\psi\}$ have closed tableaux:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \vee \psi$ | Assumption |
| 2. | $\mathbb{T}\varphi$ | Assumption |
| 3. | $\mathbb{F}\varphi$ | $\vee\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\vee\mathbb{F}\,1$ |
| | $\otimes$ | |

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \vee \psi$ | Assumption |
| 2. | $\mathbb{T}\psi$ | Assumption |
| 3. | $\mathbb{F}\varphi$ | $\vee\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\vee\mathbb{F}\,1$ |
| | $\otimes$ | |

□

**Proposition 17.24.** *1.* $\varphi, \varphi \to \psi \vdash \psi$.

*2. Both* $\neg\varphi \vdash \varphi \to \psi$ *and* $\psi \vdash \varphi \to \psi$.

*Proof.* 1. $\{\mathbb{F}\psi, \mathbb{T}\varphi \to \psi, \mathbb{T}\varphi\}$ has a closed tableau:

| | | | |
|---|---|---|---|
| 1. | $\mathbb{F}\psi$ | | Assumption |
| 2. | $\mathbb{T}\varphi \to \psi$ | | Assumption |
| 3. | $\mathbb{T}\varphi$ | | Assumption |
| 4. | $\mathbb{F}\varphi$ | $\mathbb{T}\psi$ | $\to\mathbb{T}\,2$ |
| | $\otimes$ | $\otimes$ | |

2. Both s$\{\mathbb{F}\varphi \to \psi, \mathbb{T}\neg\varphi\}$ and $\{\mathbb{F}\varphi \to \psi, \mathbb{T}\neg\psi\}$ have closed tableaux:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \to \psi$ | Assumption |
| 2. | $\mathbb{T}\neg\varphi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\to\mathbb{F}\,1$ |
| 5. | $\mathbb{F}\varphi$ | $\neg\mathbb{T}\,2$ |
| | $\otimes$ | |

| | | |
|---|---|---|
| 1. | $\mathbb{F}\varphi \to \psi$ | Assumption |
| 2. | $\mathbb{T}\neg\psi$ | Assumption |
| 3. | $\mathbb{T}\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $\mathbb{F}\psi$ | $\to\mathbb{F}\,1$ |
| 5. | $\mathbb{F}\psi$ | $\neg\mathbb{T}\,2$ |
| | $\otimes$ | |

□

## 17.10 Derivability and the Quantifiers

**Theorem 17.25.** *If $c$ is a constant not occurring in $\Gamma$ or $\varphi(x)$ and $\Gamma \vdash \varphi(c)$, then $\Gamma \vdash \forall x\, \varphi(x)$.*

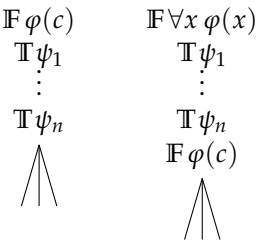
*Proof.* Suppose $\Gamma \vdash \varphi(c)$, i.e., there are $\psi_1, \ldots, \psi_n \in \Gamma$ and a closed tableau for

$$\{\mathbb{F}\,\varphi(c), \mathbb{T}\,\psi_1, \ldots, \mathbb{T}\,\psi_n\}.$$

We have to show that there is also a closed tableau for

$$\{\mathbb{F}\,\forall x\, \varphi(x), \mathbb{T}\,\psi_1, \ldots, \mathbb{T}\,\psi_n\}.$$

Take the closed tableau and replace the first assumption with $\mathbb{F}\,\forall x\, \varphi(x)$, and insert $\mathbb{F}\,\varphi(c)$ after the assumptions.



The tableau is still closed, since all sentences available as assumptions before are still available at the top of the tableau. The inserted line is the result of a correct application of $\forall\mathbb{F}$, since the constant symbol $c$ does not occur in $\psi_1$, $\ldots$, $\psi_n$ of $\forall x\, \varphi(x)$, i.e., it does not occur above the inserted line in the new tableau. □

**Proposition 17.26.** *1.* $\varphi(t) \vdash \exists x\, \varphi(x)$.

2. $\forall x\, \varphi(x) \vdash \varphi(t)$.

*Proof.* 1. A closed tableau for $\mathbb{F}\,\exists x\, \varphi(x), \mathbb{T}\,\varphi(t)$ is:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\exists x\, \varphi(x)$ | Assumption |
| 2. | $\mathbb{T}\,\varphi(t)$ | Assumption |
| 3. | $\mathbb{F}\,\varphi(t)$ | $\exists\mathbb{F}\,1$ |
| | $\otimes$ | |

2. A closed tableau for $\mathbb{F}\,\varphi(t), \mathbb{T}\,\forall x\, \varphi(x)$, is:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\varphi(t)$ | Assumption |
| 2. | $\mathbb{T}\,\forall x\, \varphi(x)$ | Assumption |
| 3. | $\mathbb{T}\,\varphi(t)$ | $\forall\mathbb{T}\,2$ |
| | $\otimes$ | |

□

## 17.11 Soundness

A derivation system, such as tableaux, is *sound* if it cannot derive things that do not actually hold. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable $\varphi$ is valid;
2. if a sentence is derivable from some others, it is also a consequence of them;
3. if a set of sentences is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

Because all these proof-theoretic properties are defined via closed tableaux of some kind or other, proving (1)–(3) above requires proving something about the semantic properties of closed tableaux. We will first define what it means for a signed formula to be satisfied in a structure, and then show that if a tableau is closed, no structure satisfies all its assumptions. (1)–(3) then follow as corollaries from this result.

**Definition 17.27.** A structure $\mathfrak{M}$ *satisfies* a signed formula $\mathbb{T}\varphi$ iff $\mathfrak{M} \vDash \varphi$, and it satisfies $\mathbb{F}\varphi$ iff $\mathfrak{M} \nvDash \varphi$. $\mathfrak{M}$ satisfies a set of signed formulas $\Gamma$ iff it satisfies every $S\,\varphi \in \Gamma$. $\Gamma$ is *satisfiable* if there is a structure that satisfies it, and *unsatisfiable* otherwise.

**Theorem 17.28** (Soundness)**.** *If $\Gamma$ has a closed tableau, $\Gamma$ is unsatisfiable.*

*Proof.* Let's call a branch of a tableau satisfiable iff the set of signed formulas on it is satisfiable, and let's call a tableau satisfiable if it contains at least one satisfiable branch.

We show the following: Extending a satisfiable tableau by one of the rules of inference always results in a satisfiable tableau. This will prove the theorem: any closed tableau results by applying rules of inference to the tableau consisting only of assumptions from $\Gamma$. So if $\Gamma$ were satisfiable, any tableau for it would be satisfiable. A closed tableau, however, is clearly not satisfiable: every branch contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$, and no structure can both satisfy and not satisfy $\varphi$.

Suppose we have a satisfiable tableau, i.e., a tableau with at least one satisfiable branch. Applying a rule of inference either adds signed formulas to a

branch, or splits a branch in two. If the tableau has a satisfiable branch which is not extended by the rule application in question, it remains a satisfiable branch in the extended tableau, so the extended tableau is satisfiable. So we only have to consider the case where a rule is applied to a satisfiable branch.

Let $\Gamma$ be the set of signed formulas on that branch, and let $S\,\varphi \in \Gamma$ be the signed formula to which the rule is applied. If the rule does not result in a split branch, we have to show that the extended branch, i.e., $\Gamma$ together with the conclusions of the rule, is still satisfiable. If the rule results in split branch, we have to show that at least one of the two resulting branches is satisfiable.

First, we consider the possible inferences with only one premise.

1. The branch is expanded by applying $\neg\mathbb{T}$ to $\mathbb{T}\neg\psi \in \Gamma$. Then the extended branch contains the signed formulas $\Gamma \cup \{\mathbb{F}\psi\}$. Suppose $\mathfrak{M} \vDash \Gamma$. In particular, $\mathfrak{M} \vDash \neg\psi$. Thus, $\mathfrak{M} \nvDash \psi$, i.e., $\mathfrak{M}$ satisfies $\mathbb{F}\psi$.

2. The branch is expanded by applying $\neg\mathbb{F}$ to $\mathbb{F}\neg\psi \in \Gamma$: Exercise.

3. The branch is expanded by applying $\wedge\mathbb{T}$ to $\mathbb{T}\psi \wedge \chi \in \Gamma$, which results in two new signed formulas on the branch: $\mathbb{T}\psi$ and $\mathbb{T}\chi$. Suppose $\mathfrak{M} \vDash \Gamma$, in particular $\mathfrak{M} \vDash \psi \wedge \chi$. Then $\mathfrak{M} \vDash \psi$ and $\mathfrak{M} \vDash \chi$. This means that $\mathfrak{M}$ satisfies both $\mathbb{T}\psi$ and $\mathbb{T}\chi$.

4. The branch is expanded by applying $\vee\mathbb{F}$ to $\mathbb{T}\psi \vee \chi \in \Gamma$: Exercise.

5. The branch is expanded by applying $\rightarrow\mathbb{F}$ to $\mathbb{T}\psi \rightarrow \chi \in \Gamma$: This results in two new signed formulas on the branch: $\mathbb{T}\psi$ and $\mathbb{F}\chi$. Suppose $\mathfrak{M} \vDash \Gamma$, in particular $\mathfrak{M} \nvDash \psi \rightarrow \chi$. Then $\mathfrak{M} \vDash \psi$ and $\mathfrak{M} \nvDash \chi$. This means that $\mathfrak{M}$ satisfies both $\mathbb{T}\psi$ and $\mathbb{F}\chi$.

6. The branch is expanded by applying $\forall\mathbb{T}$ to $\mathbb{T}\forall x\,\psi(x) \in \Gamma$: This results in a new signed formula $\mathbb{T}\varphi(t)$ on the branch. Suppose $\mathfrak{M} \vDash \Gamma$, in particular, $\mathfrak{M} \vDash \forall x\,\varphi(x)$. By Proposition 12.54, $\mathfrak{M} \vDash \varphi(t)$. Consequently, $\mathfrak{M}$ satisfies $\mathbb{T}\varphi(t)$.

7. The branch is expanded by applying $\forall\mathbb{F}$ to $\mathbb{F}\forall x\,\psi(x) \in \Gamma$: This results in a new signed formula $\mathbb{F}\varphi(a)$ where $a$ is a constant symbol not occurring in $\Gamma$. Since $\Gamma$ is satisfiable, there is a $\mathfrak{M}$ such that $\mathfrak{M} \vDash \Gamma$, in particular $\mathfrak{M} \nvDash \forall x\,\psi(x)$. We have to show that $\Gamma \cup \{\mathbb{F}\varphi(a)\}$ is satisfiable. To do this, we define a suitable $\mathfrak{M}'$ as follows.

   By Proposition 12.42, $\mathfrak{M} \nvDash \forall x\,\psi(x)$ iff for some $s$, $\mathfrak{M}, s \nvDash \psi(x)$. Now let $\mathfrak{M}'$ be just like $\mathfrak{M}$, except $a^{\mathfrak{M}'} = s(x)$. By Corollary 12.44, for any $\mathbb{T}\chi \in \Gamma$, $\mathfrak{M}' \vDash \chi$, and for any $\mathbb{F}\chi \in \Gamma$, $\mathfrak{M}' \nvDash \chi$, since $a$ does not occur in $\Gamma$.

   By Proposition 12.43, $\mathfrak{M}', s \nvDash \varphi(x)$. By Proposition 12.46, $\mathfrak{M}', s \nvDash \varphi(a)$. Since $\varphi(a)$ is a sentence, by Proposition 12.41, $\mathfrak{M}' \nvDash \varphi(a)$, i.e., $\mathfrak{M}'$ satisfies $\mathbb{F}\varphi(a)$.

8. The branch is expanded by applying $\exists\mathbb{T}$ to $\mathbb{T}\exists x\,\psi(x) \in \Gamma$: Exercise.

9. The branch is expanded by applying $\exists\mathbb{F}$ to $\mathbb{F}\exists x\,\psi(x) \in \Gamma$: Exercise.

Now let's consider the possible inferences with two premises.

1. The branch is expanded by applying $\wedge\mathbb{F}$ to $\mathbb{F}\psi \wedge \chi \in \Gamma$, which results in two branches, a left one continuing through $\mathbb{F}\psi$ and a right one through $\mathbb{F}\chi$. Suppose $\mathfrak{M} \vDash \Gamma$, in particular $\mathfrak{M} \nvDash \psi \wedge \chi$. Then $\mathfrak{M} \nvDash \psi$ or $\mathfrak{M} \nvDash \chi$. In the former case, $\mathfrak{M}$ satisfies $\mathbb{F}\psi$, i.e., $\mathfrak{M}$ satisfies the formulas on the left branch. In the latter, $\mathfrak{M}$ satisfies $\mathbb{F}\chi$, i.e., $\mathfrak{M}$ satisfies the formulas on the right branch.

2. The branch is expanded by applying $\vee\mathbb{T}$ to $\mathbb{T}\psi \vee \chi \in \Gamma$: Exercise.

3. The branch is expanded by applying $\to\mathbb{T}$ to $\mathbb{T}\psi \to \chi \in \Gamma$: Exercise.

4. The branch is expanded by Cut: This results in two branches, one containing $\mathbb{T}\psi$, the other containing $\mathbb{F}\psi$. Since $\mathfrak{M} \vDash \Gamma$ and either $\mathfrak{M} \vDash \psi$ or $\mathfrak{M} \nvDash \psi$, $\mathfrak{M}$ satisfies either the left or the right branch.

□

**Corollary 17.29.** *If $\vdash \varphi$ then $\varphi$ is valid.*

**Corollary 17.30.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* If $\Gamma \vdash \varphi$ then for some $\psi_1, \ldots, \psi_n \in \Gamma$, $\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}$ has a closed tableau. By Theorem 17.28, every structure $\mathfrak{M}$ either makes some $\psi_i$ false or makes $\varphi$ true. Hence, if $\mathfrak{M} \vDash \Gamma$ then also $\mathfrak{M} \vDash \varphi$. □

**Corollary 17.31.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then there are $\psi_1, \ldots, \psi_n \in \Gamma$ and a closed tableau for $\{\mathbb{T}\psi, \ldots, \mathbb{T}\psi\}$. By Theorem 17.28, there is no $\mathfrak{M}$ such that $\mathfrak{M} \vDash \psi_i$ for all $i = 1, \ldots, n$. But then $\Gamma$ is not satisfiable. □

## 17.12 Tableaux with Identity predicate

Tableaux with identity predicate require additional inference rules. The rules for $=$ are ($t$, $t_1$, and $t_2$ are closed terms):

$$\frac{}{\mathbb{T}t = t}\,{=} \qquad \frac{\mathbb{T}t_1 = t_2 \quad \mathbb{T}\varphi(t_1)}{\mathbb{T}\varphi(t_1)}\,{=}\mathbb{T} \qquad \frac{\mathbb{T}t_1 = t_2 \quad \mathbb{F}\varphi(t_1)}{\mathbb{F}\varphi(t_1)}\,{=}\mathbb{T}$$

Note that in contrast to all the other rules, $=\mathbb{T}$ and $=\mathbb{F}$ require that *two* signed formulas already appear on the branch, namely both $\mathbb{T}t_1 = t_2$ and $S\,\varphi(t_1)$.

**Example 17.32.** If $s$ and $t$ are closed terms, then $s = t, \varphi(s) \vdash \varphi(t)$:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,\varphi(t)$ | Assumption |
| 2. | $\mathbb{T}s = t$ | Assumption |
| 3. | $\mathbb{T}\,\varphi(s)$ | Assumption |
| 4. | $\mathbb{T}\,\varphi(t)$ | $=\mathbb{T}\,2,3$ |
| | $\otimes$ | |

This may be familiar as the principle of substitutability of identicals, or Leibniz' Law.

Tableaux prove that $=$ is symmetric:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,t = s$ | Assumption |
| 2. | $\mathbb{T}s = t$ | Assumption |
| 3. | $\mathbb{T}s = s$ | $=$ |
| 4. | $\mathbb{T}t = s$ | $=\mathbb{T}\,2,3$ |
| | $\otimes$ | |

Here, line 2 is the first prerequisite formula $\mathbb{T}s = t$ of $=\mathbb{T}$, and line 3 the second one, $\mathbb{T}\,\varphi(s)$—think of $\varphi(x)$ as $x = s$, then $\varphi(s)$ is $s = s$ and $\varphi(t)$ is $t = s$.

They also prove that $=$ is transitive:

| | | |
|---|---|---|
| 1. | $\mathbb{F}\,t_1 = t_3$ | Assumption |
| 2. | $\mathbb{T}t_1 = t_2$ | Assumption |
| 3. | $\mathbb{T}t_2 = t_3$ | Assumption |
| 4. | $\mathbb{T}t_1 = t_3$ | $=\mathbb{T}\,3,2$ |
| | $\otimes$ | |

In this tableau, the first prerequisite formula of $=\mathbb{T}$ is line 3, $\mathbb{T}t_2 = t_3$. The second one, $\mathbb{T}\,\varphi(t_2)$ is line 2. Think of $\varphi(x)$ as $t_1 = x$; that makes $\varphi(t_2)$ into $t_1 = t_2$ and $\varphi(t_3)$ into $t_1 = t_3$.

## 17.13 Soundness with Identity predicate

**Proposition 17.33.** *Tableaux with rules for identity are sound: no closed tableau is satisfiable.*

*Proof.* We just have to show as before that if a tableau has a satisfiable branch, the branch resulting from applying one of the rules for $=$ to it is also satisfiable. Let $\Gamma$ be the set of signed formulas on the branch, and let $\mathfrak{M}$ be a structure satisfying $\Gamma$.

Suppose the branch is expanded using $=$, i.e., by adding the signed formula $\mathbb{T}t = t$. Trivially, $\mathfrak{M} \vDash t = t$, so $\mathfrak{M}$ also satisfies $\Gamma \cup \{\mathbb{T}t = t\}$.

If the branch is expanded using $=\mathbb{T}$, we add a signed formula $S\,\varphi(t_2)$, but $\Gamma$ contains both $\mathbb{T}t_1 = t_2$ and $\mathbb{T}\varphi(t_1)$. Thus we have $\mathfrak{M} \vDash t_1 = t_2$ and $\mathfrak{M} \vDash \varphi(t_1)$. Let $s$ be a variable assignment with $s(x) = \mathrm{Val}^{\mathfrak{M}}(t_1)$. By Proposition 12.41, $\mathfrak{M}, s \vDash \varphi(t_1)$. Since $s \sim_x s$, by Proposition 12.46, $\mathfrak{M}, s \vDash \varphi(x)$. since $\mathfrak{M} \vDash t_1 = t_2$, we have $\mathrm{Val}^{\mathfrak{M}}(t_1) = \mathrm{Val}^{\mathfrak{M}}(t_2)$, and hence $s(x) = \mathrm{Val}^{\mathfrak{M}}(t_2)$. By applying Proposition 12.46 again, we also have $\mathfrak{M}, s \vDash \varphi(t_2)$. By Proposition 12.41, $\mathfrak{M} \vDash \varphi(t_2)$. The case of $=\mathbb{F}$ is treated similarly. □

## Problems

**Problem 17.1.** Give closed tableaux of the following:

1. $\mathbb{F}\,\neg(\varphi \to \psi) \to (\varphi \wedge \neg\psi)$
2. $\mathbb{F}\,(\varphi \to \chi) \vee (\psi \to \chi), \mathbb{T}\,(\varphi \wedge \psi) \to \chi$

**Problem 17.2.** Give closed tableaux of the following:

1. $\mathbb{F}\,\exists y\,\varphi(y) \to \psi, \mathbb{T}\forall x\,(\varphi(x) \to \psi)$
2. $\mathbb{F}\,\exists x\,(\varphi(x) \to \forall y\,\varphi(y))$

**Problem 17.3.** Prove Proposition 17.16

**Problem 17.4.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 17.5.** Complete the proof of Theorem 17.28.

**Problem 17.6.** Give closed tableaux for the following:

1. $\mathbb{F}\,\forall x\,\forall y\,((x = y \wedge \varphi(x)) \to \varphi(y))$
2. $\mathbb{F}\,\exists x\,(\varphi(x) \wedge \forall y\,(\varphi(y) \to y = x))$,
   $\mathbb{T}\,\exists x\,\varphi(x) \wedge \forall y\,\forall z\,((\varphi(y) \wedge \varphi(z)) \to y = z)$

# Chapter 18

# Axiomatic Derivations

No effort has been made yet to ensure that the material in this chapter respects various tags indicating which connectives and quantifiers are primitive or defined: all are assumed to be primitive. If the FOL tag is true, we produce a version with quantifiers, otherwise without.

## 18.1 Rules and Derivations

Axiomatic derivations are perhaps the simplest proof system for logic. A derivation is just a sequence of formulas. To count as a derivation, every formula in the sequence must either be an instance of an axiom, or must follow from one or more formulas that precede it in the sequence by a rule of inference. A derivation derives its last formula.

**Definition 18.1** (Derivability)**.** If $\Gamma$ is a set of formulas of $\mathcal{L}$ then a *derivation* from $\Gamma$ is a finite sequence $\varphi_1, \ldots, \varphi_n$ of formulas where for each $i \leq n$ one of the following holds:

1. $\varphi_i \in \Gamma$; or
2. $\varphi_i$ is an axiom; or
3. $\varphi_i$ follows from some $\varphi_j$ (and $\varphi_k$) with $j < i$ (and $k < i$) by a rule of inference.

What counts as a correct derivation depends on which inference rules we allow (and of course what we take to be axioms). And an inference rule is an if-then statement that tells us that, under certain conditions, a step $A_i$ in is a correct inference step.

**Definition 18.2** (Rule of inference)**.** A *rule of inference* gives a sufficient condition for what counts as a correct inference step in a derivation from $\Gamma$.

For instance, since any one-element sequence $\varphi$ with $\varphi \in \Gamma$ trivially counts as a derivation, the following might be a very simple rule of inference:

> If $\varphi \in \Gamma$, then $\varphi$ is always a correct inference step in any derivation from $\Gamma$.

Similarly, if $\varphi$ is one of the axioms, then $\varphi$ by itself is a derivation, and so this is also a rule of inference:

> If $\varphi$ is an axiom, then $\varphi$ is a correct inference step.

It gets more interesting if the rule of inference appeals to formulas that appear before the step considered. The following rule is called *modus ponens:*

> If $\psi \to \varphi$ and $\psi$ occur higher up in the derivation, then $\varphi$ is a correct inference step.

If this is the only rule of inference, then our definition of derivation above amounts to this: $\varphi_1, \ldots, \varphi_n$ is a derivation iff for each $i \leq n$ one of the following holds:

1. $\varphi_i \in \Gamma$; or
2. $\varphi_i$ is an axiom; or
3. for some $j < i$, $\varphi_j$ is $\psi \to \varphi_i$, and for some $k < i$, $\varphi_k$ is $\psi$.

The last clause says that $\varphi_i$ follows from $\varphi_j$ ($\psi$) and $\varphi_k$ ($\psi \to \varphi_i$) by modus ponens. If we can go from 1 to $n$, and each time we find a formula $\varphi_i$ that is either in $\Gamma$, an axiom, or which a rule of inference tells us that it is a correct inference step, then the entire sequence counts as a correct derivation.

**Definition 18.3** (Derivability)**.** A formula $\varphi$ is *derivable* from $\Gamma$, written $\Gamma \vdash \varphi$, if there is a derivation from $\Gamma$ ending in $\varphi$.

**Definition 18.4** (Theorems)**.** A formula $\varphi$ is a *theorem* if there is a derivation of $\varphi$ from the empty set. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

## 18.2 Axiom and Rules for the Propositional Connectives

**Definition 18.5** (Axioms)**.** The set of $\mathrm{Ax}_0$ of *axioms* for the propositional connectives comprises all formulas of the following forms:

$$(\varphi \wedge \psi) \to \varphi \tag{18.1}$$
$$(\varphi \wedge \psi) \to \psi \tag{18.2}$$
$$\varphi \to (\psi \to (\varphi \wedge \psi)) \tag{18.3}$$
$$\varphi \to (\varphi \vee \psi) \tag{18.4}$$
$$\varphi \to (\psi \vee \varphi) \tag{18.5}$$
$$(\varphi \to \chi) \to ((\psi \to \chi) \to ((\varphi \vee \psi) \to \chi)) \tag{18.6}$$
$$\varphi \to (\psi \to \varphi) \tag{18.7}$$
$$(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi)) \tag{18.8}$$
$$(\varphi \to \psi) \to ((\varphi \to \neg\psi) \to \neg\varphi) \tag{18.9}$$
$$\neg\varphi \to (\varphi \to \psi) \tag{18.10}$$
$$\top \tag{18.11}$$
$$\bot \to \varphi \tag{18.12}$$
$$(\varphi \to \bot) \to \neg\varphi \tag{18.13}$$
$$\neg\neg\varphi \to \varphi \tag{18.14}$$

**Definition 18.6** (Modus ponens)**.** If $\psi$ and $\psi \to \varphi$ already occur in a derivation, then $\varphi$ is a correct inference step.

We'll abbreviate the rule modus ponens as "MP."

## 18.3 Axioms and Rules for Quantifiers

**Definition 18.7** (Axioms for quantifiers)**.** The *axioms* governing quantifiers are all instances of the following:

$$\forall x\, \psi \to \psi(t), \tag{18.15}$$
$$\psi(t) \to \exists x\, \psi. \tag{18.16}$$

for any ground term $t$.

**Definition 18.8** (Rules for quantifiers)**.**

If $\psi \to \varphi(a)$ already occurs in the derivation and $a$ does not occur in $\Gamma$ or $\psi$, then $\psi \to \forall x\, \varphi(x)$ is a correct inference step.

If $\varphi(a) \to \psi$ already occurs in the derivation and $a$ does not occur in $\Gamma$ or $\psi$, then $\exists x\, \varphi(x) \to \psi$ is a correct inference step.

We'll abbreviate either of these by "QR."

## 18.4 Examples of Derivations

**Example 18.9.** Suppose we want to prove $(\neg\theta \vee \alpha) \to (\theta \to \alpha)$. Clearly, this is not an instance of any of our axioms, so we have to use the MP rule to derive it. Our only rule is MP, which given $\varphi$ and $\varphi \to \psi$ allows us to justify $\psi$. One strategy would be to use eq. (18.6) with $\varphi$ being $\neg\theta$, $\psi$ being $\alpha$, and $\chi$ being $\theta \to \alpha$, i.e., the instance

$$(\neg\theta \to (\theta \to \alpha)) \to ((\alpha \to (\theta \to \alpha)) \to ((\neg\theta \vee \alpha) \to (\theta \to \alpha))).$$

Why? Two applications of MP yield the last part, which is what we want. And we easily see that $\neg\theta \to (\theta \to \alpha)$ is an instance of eq. (18.10), and $\alpha \to (\theta \to \alpha)$ is an instance of eq. (18.7). So our derivation is:

| | | |
|---|---|---|
| 1. | $\neg\theta \to (\theta \to \alpha)$ | eq. (18.7) |
| 2. | $(\neg\theta \to (\theta \to \alpha)) \to$ $((\alpha \to (\theta \to \alpha)) \to ((\neg\theta \vee \alpha) \to (\theta \to \alpha)))$ | eq. (18.6) |
| 3. | $((\alpha \to (\theta \to \alpha)) \to ((\neg\theta \vee \alpha) \to (\theta \to \alpha))$ | 1, 2, MP |
| 4. | $\alpha \to (\theta \to \alpha)$ | eq. (18.7) |
| 5. | $(\neg\theta \vee \alpha) \to (\theta \to \alpha)$ | 3, 4, MP |

**Example 18.10.** Let's try to find a derivation of $\theta \to \theta$. It is not an instance of an axiom, so we have to use MP to derive it. eq. (18.7) is an axiom of the form $\varphi \to \psi$ to which we could apply MP. To be useful, of course, the $\psi$ which MP would justify as a correct step in this case would have to be $\theta \to \theta$, since this is what we want to derive. That means $\varphi$ would also have to be $\theta$, i.e., we might look at this instance of eq. (18.7):

$$\theta \to (\theta \to \theta)$$

In order to apply MP, we would also need to justify the corresponding second premise, namely $\varphi$. But in our case, that would be $\theta$, and we won't be able to derive $\theta$ by itself. So we need a different strategy.

The other axiom involving just $\to$ is eq. (18.8), i.e.,

$$(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$$

We could get to the last nested conditional by applying MP twice. Again, that would mean that we want an instance of eq. (18.8) where $\varphi \to \chi$ is $\theta \to \theta$, the formula we are aiming for. Then of course, $\varphi$ and $\chi$ are both $\theta$. How should we pick $\psi$ so that both $\varphi \to (\psi \to \chi)$ and $\varphi \to \psi$, i.e., in our case $\theta \to (\psi \to \theta)$ and $\theta \to \psi$, are also derivable? Well, the first of these is already an instance of eq. (18.7), whatever we decide $\psi$ to be. And $\theta \to \psi$ would be another instance of eq. (18.7) if $\psi$ were $(\theta \to \theta)$. So, our derivation is:

| | | |
|---|---|---|
| 1. | $\theta \to ((\theta \to \theta) \to \theta)$ | eq. (18.7) |
| 2. | $(\theta \to ((\theta \to \theta) \to \theta)) \to$ $((\theta \to (\theta \to \theta)) \to (\theta \to \theta))$ | eq. (18.8) |
| 3. | $(\theta \to (\theta \to \theta)) \to (\theta \to \theta)$ | 1, 2, MP |
| 4. | $\theta \to (\theta \to \theta)$ | eq. (18.7) |
| 5. | $\theta \to \theta$ | 3, 4, MP |

**Example 18.11.** Sometimes we want to show that there is a derivation of some formula from some other formulas $\Gamma$. For instance, let's show that we can derive $\varphi \to \chi$ from $\Gamma = \{\varphi \to \psi, \psi \to \chi\}$.

| | | |
|---|---|---|
| 1. | $\varphi \to \psi$ | HYP |
| 2. | $\psi \to \chi$ | HYP |
| 3. | $(\psi \to \chi) \to (\varphi \to (\psi \to \chi))$ | eq. (18.7) |
| 4. | $\varphi \to (\psi \to \chi)$ | 2, 3, MP |
| 5. | $(\varphi \to (\psi \to \chi)) \to$ $((\varphi \to \psi) \to (\varphi \to \chi))$ | eq. (18.8) |
| 6. | $((\varphi \to \psi) \to (\varphi \to \chi))$ | 4, 5, MP |
| 7. | $\varphi \to \chi$ | 1, 6, MP |

The lines labelled "HYP" (for "hypothesis") indicate that the formula on that line is an element of $\Gamma$.

**Proposition 18.12.** *If $\Gamma \vdash \varphi \to \psi$ and $\Gamma \vdash \psi \to \chi$, then $\Gamma \vdash \varphi \to \chi$*

*Proof.* Suppose $\Gamma \vdash \varphi \to \psi$ and $\Gamma \vdash \psi \to \chi$. Then there is a derivation of $\varphi \to \psi$ from $\Gamma$; and a derivation of $\psi \to \chi$ from $\Gamma$ as well. Combine these into a single derivation by concatenating them. Now add lines 3–7 of the derivation in the preceding example. This is a derivation of $\varphi \to \chi$—which is the last line of the new derivation—from $\Gamma$. Note that the justifications of lines 4 and 7 remain valid if the reference to line number 2 is replaced by reference to the last line of the derivation of $\varphi \to \psi$, and reference to line number 1 by reference to the last line of the derivation of $B \to \chi$. □

## 18.5 Derivations with Quantifiers

**Example 18.13.** Let us give a derivation of $(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to \forall x\, (\varphi(x) \land \psi(x))$.

First, note that

$$(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to \forall x\, \varphi(x)$$

is an instance of eq. (18.1), and

$$\forall x\, \varphi(x) \to \varphi(a)$$

of eq. (18.15). So, by Proposition 18.12, we know that

$$(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to \varphi(a)$$

is derivable. Likewise, since

$$(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to \forall y\, \psi(y) \qquad \text{and}$$
$$\forall y\, \psi(y) \to \psi(a)$$

are instances of eq. (18.2) and eq. (18.15), respectively,

$$(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to \psi(a)$$

is derivable by Proposition 18.12. Using an appropriate instance of eq. (18.3) and two applications of MP, we see that

$$(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to (\varphi(a) \land \psi(a))$$

is derivable. We can now apply QR to obtain

$$(\forall x\, \varphi(x) \land \forall y\, \psi(y)) \to \forall x\, (\varphi(x) \land \psi(x)).$$

## 18.6 Proof-Theoretic Notions

Just as we've defined a number of important semantic notions (validity, entailment, satisfiabilty), we now define corresponding *proof-theoretic notions*. These are not defined by appeal to satisfaction of sentences in structures, but by appeal to the derivability or non-derivability of certain formulas. It was an important discovery that these notions coincide. That they do is the content of the *soundness* and *completeness theorems*.

**Definition 18.14** (Derivability)**.** A formula $\varphi$ is *derivable* from $\Gamma$, written $\Gamma \vdash \varphi$, if there is a derivation from $\Gamma$ ending in $\varphi$.

**Definition 18.15** (Theorems)**.** A formula $\varphi$ is a *theorem* if there is a derivation of $\varphi$ from the empty set. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Definition 18.16** (Consistency)**.** A set $\Gamma$ of formulas is *consistent* if and only if $\Gamma \nvdash \bot$; it is *inconsistent* otherwise.

**Proposition 18.17** (Reflexivity)**.** *If $\varphi \in \Gamma$, then $\Gamma \vdash \varphi$.*

*Proof.* The formula $\varphi$ by itself is a derivation of $\varphi$ from $\Gamma$. □

**Proposition 18.18** (Monotony)**.** *If $\Gamma \subseteq \Delta$ and $\Gamma \vdash \varphi$, then $\Delta \vdash \varphi$.*

*Proof.* Any derivation of $\varphi$ from $\Gamma$ is also a derivation of $\varphi$ from $\Delta$. □

**Proposition 18.19** (Transitivity)**.** *If $\Gamma \vdash \varphi$ and $\{\varphi\} \cup \Delta \vdash \psi$, then $\Gamma \cup \Delta \vdash \psi$.*

*Proof.* Suppose $\{\varphi\} \cup \Delta \vdash \psi$. Then there is a derivation $\psi_1, \ldots, \psi_l = \psi$ from $\{\varphi\} \cup \Delta$. Some of the steps in that derivation will be correct because of a rule which refers to a prior line $\psi_i = \varphi$. By hypothesis, there is a derivation of $\varphi$ from $\Gamma$, i.e., a derivation $\varphi_1, \ldots, \varphi_k = \varphi$ where every $\varphi_i$ is an axiom, an element of $\Gamma$, or correct by a rule of inference. Now consider the sequence

$$\varphi_1, \ldots, \varphi_k = \varphi, \psi_1, \ldots, \psi_l = \psi.$$

This is a correct derivation of $\psi$ from $\Gamma \cup \Delta$ since every $B_i = \varphi$ is now justified by the same rule which justifies $\varphi_k = \varphi$. □

Note that this means that in particular if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. It follows also that if $\varphi_1, \ldots, \varphi_n \vdash \psi$ and $\Gamma \vdash \varphi_i$ for each $i$, then $\Gamma \vdash \psi$.

**Proposition 18.20.** *$\Gamma$ is inconsistent iff $\Gamma \vdash \varphi$ for every $\varphi$.*

*Proof.* Exercise. □

**Proposition 18.21** (Compactness)**.** 1. *If $\Gamma \vdash \varphi$ then there is a finite subset $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \vdash \varphi$.*

2. *If every finite subset of $\Gamma$ is consistent, then $\Gamma$ is consistent.*

*Proof.* 1. If $\Gamma \vdash \varphi$, then there is a finite sequence of formulas $\varphi_1, \ldots, \varphi_n$ so that $\varphi \equiv \varphi_n$ and each $\varphi_i$ is either a logical axiom, an element of $\Gamma$ or follows from previous formulas by modus ponens. Take $\Gamma_0$ to be those $\varphi_i$ which are in $\Gamma$. Then the derivation is likewise a derivation from $\Gamma_0$, and so $\Gamma_0 \vdash \varphi$.

2. This is the contrapositive of (1) for the special case $\varphi \equiv \bot$.

□

## 18.7 The Deduction Theorem

As we've seen, giving derivations in an axiomatic system is cumbersome, and derivations may be hard to find. Rather than actually write out long lists of formulas, it is generally easier to argue that such derivations exist, by making use of a few simple results. We've already established three such results: Proposition 18.17 says we can always assert that $\Gamma \vdash \varphi$ when we know that $\varphi \in \Gamma$. Proposition 18.18 says that if $\Gamma \vdash \varphi$ then also $\Gamma \cup \{\psi\} \vdash \varphi$. And Proposition 18.19 implies that if $\Gamma \vdash \varphi$ and $\varphi \vdash \psi$, then $\Gamma \vdash \psi$. Here's another simple result, a "meta"-version of modus ponens:

**Proposition 18.22.** *If $\Gamma \vdash \varphi$ and $\Gamma \vdash \varphi \to \psi$, then $\Gamma \vdash \psi$.*

*Proof.* We have that $\{\varphi, \varphi \to \psi\} \vdash \psi$:

1. $\varphi$ Hyp.
2. $\varphi \to \psi$ Hyp.
3. $\psi$ 1, 2, MP

By Proposition 18.19, $\Gamma \vdash \psi$. □

The most important result we'll use in this context is the deduction theorem:

**Theorem 18.23** (Deduction Theorem)**.** *$\Gamma \cup \{\varphi\} \vdash \psi$ if and only if $\Gamma \vdash \varphi \to \psi$.*

*Proof.* The "if" direction is immediate. If $\Gamma \vdash \varphi \to \psi$ then also $\Gamma \cup \{\varphi\} \vdash \varphi \to \psi$ by Proposition 18.18. Also, $\Gamma \cup \{\varphi\} \vdash \varphi$ by Proposition 18.17. So, by Proposition 18.22, $\Gamma \cup \{\varphi\} \vdash \psi$.

For the "only if" direction, we proceed by induction on the length of the derivation of $\psi$ from $\Gamma \cup \{\varphi\}$.

For the induction basis, we prove the claim for every derivation of length 1. A derivation of $\psi$ from $\Gamma \cup \{\varphi\}$ of length 1 consists of $\psi$ by itself; and if it is correct $\psi$ is either $\in \Gamma \cup \{\varphi\}$ or is an axiom. If $\psi \in \Gamma$ or is an axiom, then $\Gamma \vdash \psi$. We also have that $\Gamma \vdash \psi \to (\varphi \to \psi)$ by eq. (18.7), and Proposition 18.22 gives $\Gamma \vdash \varphi \to \psi$. If $\psi \in \{\varphi\}$ then $\Gamma \vdash \varphi \to \psi$ because then last sentence $\varphi \to \psi$ is the same as $\varphi \to \varphi$, and we have derived that in Example 18.10.

For the inductive step, suppose a derivation of $\psi$ from $\Gamma \cup \{\varphi\}$ ends with a step $\psi$ which is justified by modus ponens. (If it is not justified by modus ponens, $\psi \in \Gamma$, $\psi \equiv \varphi$, or $\psi$ is an axiom, and the same reasoning as in the induction basis applies.) Then some previous steps in the derivation are $\chi \to \psi$ and $\chi$, for some formula $\chi$, i.e., $\Gamma \cup \{\varphi\} \vdash \chi \to \psi$ and $\Gamma \cup \{\varphi\} \vdash \chi$, and the respective derivations are shorter, so the inductive hypothesis applies to them. We thus have both:

$$\Gamma \vdash \varphi \to (\chi \to \psi);$$
$$\Gamma \vdash \varphi \to \chi.$$

But also

$$\Gamma \vdash (\varphi \to (\chi \to \psi)) \to ((\varphi \to \chi) \to (\varphi \to \psi)),$$

by eq. (18.8), and two applications of Proposition 18.22 give $\Gamma \vdash \varphi \to \psi$, as required. □

Notice how eq. (18.7) and eq. (18.8) were chosen precisely so that the Deduction Theorem would hold.

The following are some useful facts about derivability, which we leave as exercises.

**Proposition 18.24.** *1. $\vdash (\varphi \to \psi) \to ((\psi \to \chi) \to (\varphi \to \chi))$;*

2. *If* $\Gamma \cup \{\neg\varphi\} \vdash \neg\psi$ *then* $\Gamma \cup \{\psi\} \vdash \varphi$ *(Contraposition);*
3. $\{\varphi, \neg\varphi\} \vdash \psi$ *(Ex Falso Quodlibet, Explosion);*
4. $\{\neg\neg\varphi\} \vdash \varphi$ *(Double Negation Elimination);*
5. *If* $\Gamma \vdash \neg\neg\varphi$ *then* $\Gamma \vdash \varphi$*;*

## 18.8 The Deduction Theorem with Quantifiers

**Theorem 18.25** (Deduction Theorem)**.** *If* $\Gamma \cup \{\varphi\} \vdash \psi$ *and then* $\Gamma \vdash \varphi \to \psi$.

*Proof.* We again proceed by induction on the length of the derivation of $\psi$ from $\Gamma \cup \{\varphi\}$.

The proof of the induction basis is identical to that in the proof of Theorem 18.23.

For the inductive step, suppose again that the derivation of $\psi$ from $\Gamma \cup \{\varphi\}$ ends with a step $\psi$ which is justified by an inference rule. If the inference rule is modus ponens, we proceed as in the proof of Theorem 18.23. If the inference rule is QR, we know that $\psi \equiv \chi \to \forall x\, \theta(x)$ and a formula of the form $\chi \to \theta(a)$ appears earlier in the derivation, where $a$ does not occur in $\chi$, $\varphi$, or $\Gamma$. We thus have that

$$\Gamma \cup \{\varphi\} \vdash \chi \to \theta(a)$$

and the induction hypothesis applies, i.e., we have that

$$\Gamma \vdash \varphi \to \theta(a)$$

By

$$\vdash (\varphi \to (\chi \to \theta(a))) \to ((\varphi \wedge \chi) \to \theta(a))$$

and modus ponens we get

$$\Gamma \vdash (\varphi \wedge \chi) \to \theta(a).$$

Since the eigenvariale condition still applies, we can add a step to this derivation justified by QR, and get:

$$\Gamma \vdash (\varphi \wedge \chi) \to \forall x\, \theta(x)$$

We also have

$$\vdash ((\varphi \wedge \chi) \to \forall x\, \theta(x)) \to (\varphi \to (\chi \to \forall x\, \theta(x))$$

so by modus ponens,

$$\Gamma \vdash \varphi \to (\chi \to \forall x\, \theta(x))$$

i.e., $\Gamma \vdash \psi$.

We leave the case where $\psi$ is justified by the rule QR, but is of the form $\exists x\, \theta(x) \to \chi$, as an exercise. □

## 18.9 Derivability and Consistency

We will now establish a number of properties of the derivability relation. They are independently interesting, but each will play a role in the proof of the completeness theorem.

**Proposition 18.26.** *If $\Gamma \vdash \varphi$ and $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* If $\Gamma \cup \{\varphi\}$ is inconsistent, then $\Gamma \cup \{\varphi\} \vdash \bot$. By Proposition 18.17, $\Gamma \vdash \psi$ for every $\psi \in \Gamma$. Since also $\Gamma \vdash \varphi$ by hypothesis, $\Gamma \vdash \psi$ for every $\psi \in \Gamma \cup \{\varphi\}$. By Proposition 18.19, $\Gamma \vdash \bot$, i.e., $\Gamma$ is inconsistent. □

**Proposition 18.27.** *$\Gamma \vdash \varphi$ iff $\Gamma \cup \{\neg\varphi\}$ is inconsistent.*

*Proof.* First suppose $\Gamma \vdash \varphi$. Then $\Gamma \cup \{\neg\varphi\} \vdash \varphi$ by Proposition 18.18. $\Gamma \cup \{\neg\varphi\} \vdash \neg\varphi$ by Proposition 18.17. We also have $\vdash \neg\varphi \to (\varphi \to \bot)$ by eq. (18.10). So by two applications of Proposition 18.22, we have $\Gamma \cup \{\neg\varphi\} \vdash \bot$.

Now assume $\Gamma \cup \{\neg\varphi\}$ is inconsistent, i.e., $\Gamma \cup \{\neg\varphi\} \vdash \bot$. By the deduction theorem, $\Gamma \vdash \neg\varphi \to \bot$. $\Gamma \vdash (\neg\varphi \to \bot) \to \neg\neg\varphi$ by eq. (18.13), so $\Gamma \vdash \neg\neg\varphi$ by Proposition 18.22. Since $\Gamma \vdash \neg\neg\varphi \to \varphi$ (eq. (18.14)), we have $\Gamma \vdash \varphi$ by Proposition 18.22 again. □

**Proposition 18.28.** *If $\Gamma \vdash \varphi$ and $\neg\varphi \in \Gamma$, then $\Gamma$ is inconsistent.*

*Proof.* $\Gamma \vdash \neg\varphi \to (\varphi \to \bot)$ by eq. (18.10). $\Gamma \vdash \bot$ by two applications of Proposition 18.22. □

**Proposition 18.29.** *If $\Gamma \cup \{\varphi\}$ and $\Gamma \cup \{\neg\varphi\}$ are both inconsistent, then $\Gamma$ is inconsistent.*

*Proof.* Exercise. □

## 18.10 Derivability and the Propositional Connectives

**Proposition 18.30.** *1. Both $\varphi \wedge \psi \vdash \varphi$ and $\varphi \wedge \psi \vdash \psi$*

*2. $\varphi, \psi \vdash \varphi \wedge \psi$.*

*Proof.* 1. From eq. (18.1) and eq. (18.1) by modus ponens.

2. From eq. (18.3) by two applications of modus ponens.

□

**Proposition 18.31.** *1. $\varphi \vee \psi, \neg\varphi, \neg\psi$ is inconsistent.*

*2. Both $\varphi \vdash \varphi \vee \psi$ and $\psi \vdash \varphi \vee \psi$.*

*Proof.* 1. From eq. (18.9) we get $\vdash \neg\varphi \to (\varphi \to \bot)$ and $\vdash \neg\varphi \to (\varphi \to \bot)$. So by the deduction theorem, we have $\{\neg\varphi\} \vdash \varphi \to \bot$ and $\{\neg\psi\} \vdash \psi \to \bot$. From eq. (18.6) we get $\{\neg\varphi, \neg\psi\} \vdash (\varphi \vee \psi) \to \bot$. By the deduction theorem, $\{\varphi \vee \psi, \neg\varphi, \neg\psi\} \vdash \bot$.

2. From eq. (18.4) and eq. (18.5) by modus ponsens.

□

**Proposition 18.32.** *1. $\varphi, \varphi \to \psi \vdash \psi$.*

*2. Both $\neg\varphi \vdash \varphi \to \psi$ and $\psi \vdash \varphi \to \psi$.*

*Proof.* 1. We can derive:

| | | |
|---|---|---|
| 1. | $\varphi$ | HYP |
| 2. | $\varphi \to \psi$ | HYP |
| 3. | $\psi$ | 1, 2, MP |

2. By eq. (18.10) and eq. (18.7) and the deduction theorem, respectively.

□

## 18.11 Derivability and the Quantifiers

**Theorem 18.33.** *If $c$ is a constant symbol not occurring in $\Gamma$ or $\varphi(x)$ and $\Gamma \vdash \varphi(c)$, then $\Gamma \vdash \forall x\, \varphi(x)$.*

*Proof.* By the deduction theorem, $\Gamma \vdash \top \to \varphi(c)$. Since $c$ does not occur in $\Gamma$ or $\top$, we get $\Gamma \vdash \top \to \varphi(c)$. By the deduction theorem again, $\Gamma \vdash \forall x\, \varphi(x)$. □

**Proposition 18.34.** *1. $\varphi(t) \vdash \exists x\, \varphi(x)$.*

*2. $\forall x\, \varphi(x) \vdash \varphi(t)$.*

*Proof.* 1. By eq. (18.16) and the deduction theorem.

2. By eq. (18.15) and the deduction theorem.

□

## 18.12 Soundness

A derivation system, such as axiomatic deduction, is *sound* if it cannot derive things that do not actually hold. Soundness is thus a kind of guaranteed safety property for derivation systems. Depending on which proof theoretic property is in question, we would like to know for instance, that

1. every derivable $\varphi$ is valid;
2. if $\varphi$ is derivable from some others $\Gamma$, it is also a consequence of them;
3. if a set of formulas $\Gamma$ is inconsistent, it is unsatisfiable.

These are important properties of a derivation system. If any of them do not hold, the derivation system is deficient—it would derive too much. Consequently, establishing the soundness of a derivation system is of the utmost importance.

**Proposition 18.35.** *If $\varphi$ is an axiom, then $\mathfrak{M}, s \vDash \varphi$ for each structure $\mathfrak{M}$ and assignment $s$.*

*Proof.* We have to verify that all the axioms are valid. For instance, here is the case for eq. (18.15): suppose $t$ is free for $x$ in $\varphi$, and assume $\mathfrak{M}, s \vDash \forall x\, \varphi$. Then by definition of satisfaction, for each $s' \sim_x s$, also $\mathfrak{M}, s' \vDash \varphi$, and in particular this holds when $s'(x) = \mathrm{Val}^{\mathfrak{M}}_s(t)$. By Proposition 12.46, $\mathfrak{M}, s \vDash \varphi[t/x]$. This shows that $\mathfrak{M}, s \vDash (\forall x\, \varphi \to \varphi[t/x])$. □

**Theorem 18.36** (Soundness)**.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* By induction on the length of the derivation of $\varphi$ from $\Gamma$. If there are no steps justified by inferences, then all formulas in the derivation are either instances of axioms or are in $\Gamma$. By the previous proposition, all the axioms are valid, and hence if $\varphi$ is an axiom then $\Gamma \vDash \varphi$. If $\varphi \in \Gamma$, then trivially $\Gamma \vDash \varphi$.

If the last step of the derivation of $\varphi$ is justified by modus ponens, then there are formulas $\psi$ and $\psi \to \varphi$ in the derivation, and the induction hypothesis applies to the part of the derivation ending in those formulas (since they contain at least one fewer steps justified by an inference). So, by induction hypothesis, $\Gamma \vDash \psi$ and $\Gamma \vDash \psi \to \varphi$. Then $\Gamma \vDash \varphi$ by Theorem 12.53.

Now suppose the last step is justified by QR. Then that step has the form $\chi \to \forall x\, B(x)$ and there is a preceding step $\chi \to \psi(c)$ with $c$ not in $\Gamma$, $\chi$, or $\forall x\, B(x)$. By induction hypothesis, $\Gamma \vDash \chi \to \forall x\, B(x)$. By Theorem 12.53, $\Gamma \cup \{\chi\} \vDash \psi(c)$.

Consider some structure $\mathfrak{M}$ such that $\mathfrak{M} \vDash \Gamma \cup \{\chi\}$. We need to show that $\mathfrak{M} \vDash \forall x\, \psi(x)$. Since $\forall x\, \psi(x)$ is a sentence, this means we have to show that for every variable assignment $s$, $\mathfrak{M}, s \vDash \psi(x)$ (Proposition 12.42). Since $\Gamma \cup \{\chi\}$ consists entirely of sentences, $\mathfrak{M}, s \vDash \theta$ for all $\theta \in \Gamma$ by Definition 12.35. Let $\mathfrak{M}'$ be like $\mathfrak{M}$ except that $c^{\mathfrak{M}'} = s(x)$. Since $c$ does not occur in $\Gamma$ or $\chi$, $\mathfrak{M}' \vDash \Gamma \cup \{\chi\}$ by Corollary 12.44. Since $\Gamma \cup \{\chi\} \vDash \psi(c)$, $\mathfrak{M}' \vDash B(c)$. Since $\psi(c)$ is a sentence, $\mathfrak{M}, s \vDash \psi(c)$ by Proposition 12.41. $\mathfrak{M}', s \vDash \psi(x)$ iff $\mathfrak{M}' \vDash \psi(c)$ by Proposition 12.46 (recall that $\psi(c)$ is just $\psi(x)[c/x]$). So, $\mathfrak{M}', s \vDash \psi(x)$. Since $c$ does not occur in $\psi(x)$, by Proposition 12.43, $\mathfrak{M}, s \vDash \psi(x)$. But $s$ was an arbitrary variable assignment, so $\mathfrak{M} \vDash \forall x\, \psi(x)$. Thus $\Gamma \cup \{\chi\} \vDash \forall x\, \psi(x)$. By Theorem 12.53, $\Gamma \vDash \chi \to \forall x\, \psi(x)$.

The case where $\varphi$ is justified by QR but is of the form $\exists x\, \psi(x) \to \chi$ is left as an exercise. □

**Corollary 18.37.** *If $\vdash \varphi$, then $\varphi$ is valid.*

**Corollary 18.38.** *If $\Gamma$ is satisfiable, then it is consistent.*

*Proof.* We prove the contrapositive. Suppose that $\Gamma$ is not consistent. Then $\Gamma \vdash \bot$, i.e., there is a derivation of $\bot$ from $\Gamma$. By Theorem 18.36, any structure $\mathfrak{M}$ that satisfies $\Gamma$ must satisfy $\bot$. Since $\mathfrak{M} \nvDash \bot$ for every structure $\mathfrak{M}$, no $\mathfrak{M}$ can satisfy $\Gamma$, i.e., $\Gamma$ is not satisfiable. □

## 18.13 Derivations with Identity predicate

In order to accommodate $=$ in derivations, we simply add new axiom schemas. The definition of derivation and $\vdash$ remains the same, we just also allow the new axioms.

**Definition 18.39** (Axioms for identity predicate)**.**

$$t = t, \tag{18.17}$$

$$t_1 = t_2 \to (\psi(t_1) \to \psi(t_2)), \tag{18.18}$$

for any ground terms $t$, $t_1$, $t_2$.

**Proposition 18.40.** *The axioms eq. (18.17) and eq. (18.18) are valid.*

*Proof.* Exercise. □

**Proposition 18.41.** *$\Gamma \vdash t = t$, for any term $t$ and set $\Gamma$.*

**Proposition 18.42.** *If $\Gamma \vdash \varphi(t_1)$ and $\Gamma \vdash t_1 = t_2$, then $\Gamma \vdash \varphi(t_2)$.*

*Proof.* The formula

$$(t_1 = t_2 \to (\varphi(t_1) \to \varphi(t_2)))$$

is an instance of eq. (18.18). The conclusion follows by two applications of MP. □

## Problems

**Problem 18.1.** Show that the following hold by exhibiting derivations from the axioms:

1. $(\varphi \wedge \psi) \to (\psi \wedge \varphi)$
2. $((\varphi \wedge \psi) \to \chi) \to (\varphi \to (\psi \to \chi))$
3. $\neg(\varphi \vee \psi) \to \neg\varphi$

**Problem 18.2.** Prove Proposition 18.20.

**Problem 18.3.** Prove Proposition 18.24

**Problem 18.4.** Complete the proof of Theorem 18.25.

**Problem 18.5.** Prove that $\Gamma \vdash \neg\varphi$ iff $\Gamma \cup \{\varphi\}$ is inconsistent.

**Problem 18.6.** Prove Proposition 18.29

**Problem 18.7.** Complete the proof of Theorem 18.36.

**Problem 18.8.** Prove Proposition 18.40.

# Chapter 19

# The Completeness Theorem

## 19.1 Introduction

The completeness theorem is one of the most fundamental results about logic. It comes in two formulations, the equivalence of which we'll prove. In its first formulation it says something fundamental about the relationship between semantic consequence and our proof system: if a sentence $\varphi$ follows from some sentences $\Gamma$, then there is also a derivation that establishes $\Gamma \vdash \varphi$. Thus, the proof system is as strong as it can possibly be without proving things that don't actually follow.

In its second formulation, it can be stated as a model existence result: every consistent set of sentences is satisfiable. Consistency is a proof-theoretic notion: it says that our proof system is unable to produce certain derivations. But who's to say that just because there are no derivations of a certain sort from $\Gamma$, it's guaranteed that there is a structure $\mathfrak{M}$? Before the completeness theorem was first proved—in fact before we had the proof systems we now do—the great German mathematician David Hilbert held the view that consistency of mathematical theories guarantees the existence of the objects they are about. He put it as follows in a letter to Gottlob Frege:

> If the arbitrarily given axioms do not contradict one another with all their consequences, then they are true and the things defined by the axioms exist. This is for me the criterion of truth and existence.

Frege vehemently disagreed. The second formulation of the completeness theorem shows that Hilbert was right in at least the sense that if the axioms are consistent, then *some* structure exists that makes them all true.

These aren't the only reasons the completeness theorem—or rather, its proof—is important. It has a number of important consequences, some of which we'll discuss separately. For instance, since any derivation that shows $\Gamma \vdash \varphi$ is finite and so can only use finitely many of the sentences in $\Gamma$, it follows by the completeness theorem that if $\varphi$ is a consequence of $\Gamma$, it is already

a consequence of a finite subset of $\Gamma$. This is called *compactness*. Equivalently, if every finite subset of $\Gamma$ is consistent, then $\Gamma$ itself must be consistent.

Although the compactness theorem follows from the completeness theorem via the detour through derivations, it is also possible to use the *the proof of* the completeness theorem to establish it directly. For what the proof does is take a set of sentences with a certain property—consistency—and constructs a structure out of this set that has certain properties (in this case, that it satisfies the set). Almost the very same construction can be used to directly establish compactness, by starting from "finitely satisfiable" sets of sentences instead of consistent ones. The construction also yields other consequences, e.g., that any satisfiable set of sentences has a finite or denumerable model. (This result is called the Löwenheim-Skolem theorem.) In general, the construction of structures from sets of sentences is used often in logic, and sometimes even in philosophy.

## 19.2 Outline of the Proof

The proof of the completeness theorem is a bit complex, and upon first reading it, it is easy to get lost. So let us outline the proof. The first step is a shift of perspective, that allows us to see a route to a proof. When completeness is thought of as "whenever $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$," it may be hard to even come up with an idea: for to show that $\Gamma \vdash \varphi$ we have to find a derivation, and it does not look like the hypothesis that $\Gamma \vDash \varphi$ helps us for this in any way. For some proof systems it is possible to directly construct a derivation, but we will take a slightly different tack. The shift in perspective required is this: completeness can also be formulated as: "if $\Gamma$ is consistent, it has a model." Perhaps we can use the information in $\Gamma$ together with the hypothesis that it is consistent to construct a model. After all, we know what kind of model we are looking for: one that is as $\Gamma$ describes it!

If $\Gamma$ contains only atomic sentences, it is easy to construct a model for it. Suppose the atomic sentences are all of the form $P(a_1, \ldots, a_n)$ where the $a_i$ are constant symbols. All we have to do is come up with a domain $|\mathfrak{M}|$ and an assignment for $P$ so that $\mathfrak{M} \vDash P(a_1, \ldots, a_n)$. But that's not very hard: put $|\mathfrak{M}| = \mathbb{N}$, $c_i^{\mathfrak{M}} = i$, and for every $P(a_1, \ldots, a_n) \in \Gamma$, put the tuple $\langle k_1, \ldots, k_n \rangle$ into $P^{\mathfrak{M}}$, where $k_i$ is the index of the constant symbol $a_i$ (i.e., $a_i \equiv c_{k_i}$).

Now suppose $\Gamma$ contains some formula $\neg\psi$, with $\psi$ atomic. We might worry that the construction of $\mathfrak{M}$ interferes with the possibility of making $\neg\psi$ true. But here's where the consistency of $\Gamma$ comes in: if $\neg\psi \in \Gamma$, then $\psi \notin \Gamma$, or else $\Gamma$ would be inconsistent. And if $\psi \notin \Gamma$, then according to our construction of $\mathfrak{M}$, $\mathfrak{M} \nvDash \psi$, so $\mathfrak{M} \vDash \neg\psi$. So far so good.

What if $\Gamma$ contains complex, non-atomic formulas? Say it contains $\varphi \wedge \psi$. To make that true, we should proceed as if both $\varphi$ and $\psi$ were in $\Gamma$. And if

$\varphi \vee \psi \in \Gamma$, then we will have to make at least one of them true, i.e., proceed as if one of them was in $\Gamma$.

This suggests the following idea: we add additional formulas to $\Gamma$ so as to (a) keep the resulting set consistent and (b) make sure that for every possible atomic sentence $\varphi$, either $\varphi$ is in the resulting set, or $\neg\varphi$ is, and (c) such that, whenever $\varphi \wedge \psi$ is in the set, so are both $\varphi$ and $\psi$, if $\varphi \vee \psi$ is in the set, at least one of $\varphi$ or $\psi$ is also, etc. We keep doing this (potentially forever). Call the set of all formulas so added $\Gamma^*$. Then our construction above would provide us with a structure $\mathfrak{M}$ for which we could prove, by induction, that all sentences in $\Gamma^*$ are true in it, and hence also all sentence in $\Gamma$ since $\Gamma \subseteq \Gamma^*$. It turns out that guaranteeing (a) and (b) is enough. A set of sentences for which (b) holds is called *complete*. So our task will be to extend the consistent set $\Gamma$ to a consistent and complete set $\Gamma^*$.

There is one wrinkle in this plan: if $\exists x\, \varphi(x) \in \Gamma$ we would hope to be able to pick some constant symbol $c$ and add $\varphi(c)$ in this process. But how do we know we can always do that? Perhaps we only have a few constant symbols in our language, and for each one of them we have $\neg\varphi(c) \in \Gamma$. We can't also add $\varphi(c)$, since this would make the set inconsistent, and we wouldn't know whether $\mathfrak{M}$ has to make $\varphi(c)$ or $\neg\varphi(c)$ true. Moreover, it might happen that $\Gamma$ contains only sentences in a language that has no constant symbols at all (e.g., the language of set theory).

The solution to this problem is to simply add infinitely many constants at the beginning, plus sentences that connect them with the quantifiers in the right way. (Of course, we have to verify that this cannot introduce an inconsistency.)

Our original construction works well if we only have constant symbols in the atomic sentences. But the language might also contain function symbols. In that case, it might be tricky to find the right functions on $\mathbb{N}$ to assign to these function symbols to make everything work. So here's another trick: instead of using $i$ to interpret $c_i$, just take the set of constant symbols itself as the domain. Then $\mathfrak{M}$ can assign every constant symbol to itself: $c_i^{\mathfrak{M}} = c_i$. But why not go all the way: let $|\mathfrak{M}|$ be all *terms* of the language! If we do this, there is an obvious assignment of functions (that take terms as arguments and have terms as values) to function symbols: we assign to the function symbol $f_i^n$ the function which, given $n$ terms $t_1, \ldots, t_n$ as input, produces the term $f_i^n(t_1, \ldots, t_n)$ as value.

The last piece of the puzzle is what to do with $=$. The predicate symbol $=$ has a fixed interpretation: $\mathfrak{M} \vDash t = t'$ iff $\mathrm{Val}^{\mathfrak{M}}(t) = \mathrm{Val}^{\mathfrak{M}}(t')$. Now if we set things up so that the value of a term $t$ is $t$ itself, then this structure will make *no* sentence of the form $t = t'$ true unless $t$ and $t'$ are one and the same term. And of course this is a problem, since basically every interesting theory in a language with function symbols will have as theorems sentences $t = t'$ where $t$ and $t'$ are not the same term (e.g., in theories of arithmetic: $(\mathsf{0} + \mathsf{0}) = \mathsf{0}$). To

solve this problem, we change the domain of $\mathfrak{M}$: instead of using terms as the objects in $|\mathfrak{M}|$, we use sets of terms, and each set is so that it contains all those terms which the sentences in $\Gamma$ require to be equal. So, e.g., if $\Gamma$ is a theory of arithmetic, one of these sets will contain: $\mathsf{0}$, $(\mathsf{0}+\mathsf{0})$, $(\mathsf{0}\times\mathsf{0})$, etc. This will be the set we assign to $\mathsf{0}$, and it will turn out that this set is also the value of all the terms in it, e.g., also of $(\mathsf{0}+\mathsf{0})$. Therefore, the sentence $(\mathsf{0}+\mathsf{0})=\mathsf{0}$ will be true in this revised structure.

So here's what we'll do. First we investigate the properties of complete consistent sets, in particular we prove that a complete consistent set contains $\varphi\wedge\psi$ iff it contains both $\varphi$ and $\psi$, $\varphi\vee\psi$ iff it contains at least one of them, etc. (Proposition 19.2). Then we define and investigate "saturated" sets of sentences. A saturated set is one which contains conditionals that link each quantified sentence to instances of it (Definition 19.5). We show that any consistent set $\Gamma$ can always be extended to a saturated set $\Gamma'$ (Lemma 19.6). If a set is consistent, saturated, and complete it also has the property that it contains $\exists x\,\varphi(x)$ iff it contains $\varphi(t)$ for some closed term $t$ and $\forall x\,\varphi(x)$ iff it contains $\varphi(t)$ for all closed terms $t$ (Proposition 19.7). We'll then take the saturated consistent set $\Gamma'$ and show that it can be extended to a saturated, consistent, and complete set $\Gamma^*$ (Lemma 19.8). This set $\Gamma^*$ is what we'll use to define our term model $\mathfrak{M}(\Gamma^*)$. The term model has the set of closed terms as its domain, and the interpretation of its predicate symbols is given by the atomic sentences in $\Gamma^*$ (Definition 19.9). We'll use the properties of saturated, complete consistent sets to show that indeed $\mathfrak{M}(\Gamma^*)\vDash\varphi$ iff $\varphi\in\Gamma^*$ (Lemma 19.11), and thus in particular, $\mathfrak{M}(\Gamma^*)\vDash\Gamma$. Finally, we'll consider how to define a term model if $\Gamma$ contains $=$ as well (Definition 19.15) and show that it satisfies $\Gamma^*$ (Lemma 19.17).

## 19.3 Complete Consistent Sets of Sentences

**Definition 19.1** (Complete set)**.** A set $\Gamma$ of sentences is *complete* iff for any sentence $\varphi$, either $\varphi\in\Gamma$ or $\neg\varphi\in\Gamma$.

Complete sets of sentences leave no questions unanswered. For any sentence $A$, $\Gamma$ "says" if $\varphi$ is true or false. The importance of complete sets extends beyond the proof of the completeness theorem. A theory which is complete and axiomatizable, for instance, is always decidable.

Complete consistent sets are important in the completeness proof since we can guarantee that every consistent set of sentences $\Gamma$ is contained in a complete consistent set $\Gamma^*$. A complete consistent set contains, for each sentence $\varphi$, either $\varphi$ or its negation $\neg\varphi$, but not both. This is true in particular for atomic sentences, so from a complete consistent set in a language suitably expanded by constant symbols, we can construct a structure where the interpretation of predicate symbols is defined according to which atomic sentences are in $\Gamma^*$. This structure can then be shown to make all sentences in $\Gamma^*$ (and hence also

all those in $\Gamma$) true. The proof of this latter fact requires that $\neg\varphi \in \Gamma^*$ iff $\varphi \notin \Gamma^*$, $(\varphi \vee \psi) \in \Gamma^*$ iff $\varphi \in \Gamma^*$ or $\psi \in \Gamma^*$, etc.

In what follows, we will often tacitly use the properties of reflexivity, monotonicity, and transitivity of $\vdash$ (see sections 15.8, 16.7, 17.7 and 18.6).

**Proposition 19.2.** *Suppose $\Gamma$ is complete and consistent. Then:*

1. *If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.*
2. *$\varphi \wedge \psi \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$.*
3. *$\varphi \vee \psi \in \Gamma$ iff either $\varphi \in \Gamma$ or $\psi \in \Gamma$.*
4. *$\varphi \to \psi \in \Gamma$ iff either $\varphi \notin \Gamma$ or $\psi \in \Gamma$.*

*Proof.* Let us suppose for all of the following that $\Gamma$ is complete and consistent.

1. If $\Gamma \vdash \varphi$, then $\varphi \in \Gamma$.

   Suppose that $\Gamma \vdash \varphi$. Suppose to the contrary that $\varphi \notin \Gamma$. Since $\Gamma$ is complete, $\neg\varphi \in \Gamma$. By Propositions 15.20, 16.20, 17.20 and 18.28, $\Gamma$ is inconsistent. This contradicts the assumption that $\Gamma$ is consistent. Hence, it cannot be the case that $\varphi \notin \Gamma$, so $\varphi \in \Gamma$.

2. Exercise.

3. First we show that if $\varphi \vee \psi \in \Gamma$, then either $\varphi \in \Gamma$ or $\psi \in \Gamma$. Suppose $\varphi \vee \psi \in \Gamma$ but $\varphi \notin \Gamma$ and $\psi \notin \Gamma$. Since $\Gamma$ is complete, $\neg\varphi \in \Gamma$ and $\neg\psi \in \Gamma$. By Propositions 15.23, 16.23, 17.23 and 18.31, item (1), $\Gamma$ is inconsistent, a contradiction. Hence, either $\varphi \in \Gamma$ or $\psi \in \Gamma$.

   For the reverse direction, suppose that $\varphi \in \Gamma$ or $\psi \in \Gamma$. By Propositions 15.23, 16.23, 17.23 and 18.31, item (2), $\Gamma \vdash \varphi \vee \psi$. By (1), $\varphi \vee \psi \in \Gamma$, as required.

4. Exercise.

□

## 19.4 Henkin Expansion

Part of the challenge in proving the completeness theorem is that the model we construct from a complete consistent set $\Gamma$ must make all the quantified formulas in $\Gamma$ true. In order to guarantee this, we use a trick due to Leon Henkin. In essence, the trick consists in expanding the language by infinitely many constant symbols and adding, for each formula with one free variable $\varphi(x)$ a formula of the form $\exists x\, \varphi \to \varphi(c)$, where $c$ is one of the new constant symbols. When we construct the structure satisfying $\Gamma$, this will guarantee that each true existential sentence has a witness among the new constants.

**Proposition 19.3.** *If $\Gamma$ is consistent in $\mathcal{L}$ and $\mathcal{L}'$ is obtained from $\mathcal{L}$ by adding a denumerable set of new constant symbols $d_0, d_1, \ldots$, then $\Gamma$ is consistent in $\mathcal{L}'$.*

**Definition 19.4** (Saturated set)**.** A set $\Gamma$ of formulas of a language $\mathcal{L}$ is *saturated* iff for each formula $\varphi(x) \in \mathrm{Frm}(\mathcal{L})$ with one free variable $x$ there is a constant symbol $c \in \mathcal{L}$ such that $\exists x\, \varphi(x) \to \varphi(c) \in \Gamma$.

The following definition will be used in the proof of the next theorem.

**Definition 19.5.** Let $\mathcal{L}'$ be as in Proposition 19.3. Fix an enumeration $\varphi_0(x_0)$, $\varphi_1(x_1)$, ...of all formulas $\varphi_i(x_i)$ of $\mathcal{L}'$ in which one variable ($x_i$) occurs free. We define the sentences $\theta_n$ by induction on $n$.

Let $c_0$ be the first constant symbol among the $d_i$ we added to $\mathcal{L}$ which does not occur in $\varphi_0(x_0)$. Assuming that $\theta_0, \ldots, \theta_{n-1}$ have already been defined, let $c_n$ be the first among the new constant symbols $d_i$ that occurs neither in $\theta_0, \ldots, \theta_{n-1}$ nor in $\varphi_n(x_n)$.

Now let $\theta_n$ be the formula $\exists x_n\, \varphi_n(x_n) \to \varphi_n(c_n)$.

**Lemma 19.6.** *Every consistent set $\Gamma$ can be extended to a saturated consistent set $\Gamma'$.*

*Proof.* Given a consistent set of sentences $\Gamma$ in a language $\mathcal{L}$, expand the language by adding a denumerable set of new constant symbols to form $\mathcal{L}'$. By Proposition 19.3, $\Gamma$ is still consistent in the richer language. Further, let $\theta_i$ be as in Definition 19.5. Let

$$\Gamma_0 = \Gamma$$
$$\Gamma_{n+1} = \Gamma_n \cup \{\theta_n\}$$

i.e., $\Gamma_{n+1} = \Gamma \cup \{\theta_0, \ldots, \theta_n\}$, and let $\Gamma' = \bigcup_n \Gamma_n$. $\Gamma'$ is clearly saturated.

If $\Gamma'$ were inconsistent, then for some $n$, $\Gamma_n$ would be inconsistent (Exercise: explain why). So to show that $\Gamma'$ is consistent it suffices to show, by induction on $n$, that each set $\Gamma_n$ is consistent.

The induction basis is simply the claim that $\Gamma_0 = \Gamma$ is consistent, which is the hypothesis of the theorem. For the induction step, suppose that $\Gamma_n$ is consistent but $\Gamma_{n+1} = \Gamma_n \cup \{\theta_n\}$ is inconsistent. Recall that $\theta_n$ is $\exists x_n\, \varphi_n(x_n) \to \varphi_n(c_n)$, where $\varphi_n(x_n)$ is a formula of $\mathcal{L}'$ with only the variable $x_n$ free. By the way we've chosen the $c_n$ (see Definition 19.5), $c_n$ does not occur in $A_n(x_n)$ nor in $\Gamma_n$.

If $\Gamma_n \cup \{\theta_n\}$ is inconsistent, then $\Gamma_n \vdash \neg\theta_n$, and hence both of the following hold:

$$\Gamma_n \vdash \exists x_n\, \varphi_n(x_n) \qquad \Gamma_n \vdash \neg\varphi_n(c_n)$$

Since $c_n$ does not occur in $\Gamma_n$ or in $\varphi_n(x_n)$, Theorems 15.25, 16.25, 17.25 and 18.33 applies. From $\Gamma_n \vdash \neg\varphi_n(c_n)$, we obtain $\Gamma_n \vdash \forall x_n\, \neg\varphi_n(x_n)$. Thus we have that both $\Gamma_n \vdash \exists x_n\, \varphi_n$ and $\Gamma_n \vdash \forall x_n\, \neg\varphi_n(x_n)$, so $\Gamma_n$ itself is inconsistent. (Note that $\forall x_n\, \neg\varphi_n(x_n) \vdash \neg\exists x_n\, \varphi_n(x_n)$.) Contradiction: $\Gamma_n$ was supposed to be consistent. Hence $\Gamma_n \cup \{\theta_n\}$ is consistent. □

We'll now show that *complete*, consistent sets which are saturated have the property that it contains a universally quantified sentence iff it contains all its instances and it contains an existentially quantified sentence iff it contains at least one instance. We'll use this to show that the structure we'll generate from a complete, consistent, saturated set makes all its quantified sentences true.

**Proposition 19.7.** *Suppose $\Gamma$ is complete, consistent, and saturated.*

1. *$\exists x\, \varphi(x) \in \Gamma$ iff $\varphi(t) \in \Gamma$ for at least one closed term $t$.*
2. *$\forall x\, \varphi(x) \in \Gamma$ iff $\varphi(t) \in \Gamma$ for all closed terms $t$.*

*Proof.* 1. First suppose that $\exists x\, \varphi(x) \in \Gamma$. Because $\Gamma$ is saturated, $(\exists x\, \varphi(x) \to \varphi(c)) \in \Gamma$ for some constant symbol $c$. By Propositions 15.24, 16.24, 17.24 and 18.32, item (1), and Proposition 19.2(1), $\varphi(c) \in \Gamma$.

   For the other direction, saturation is not necessary: Suppose $\varphi(t) \in \Gamma$. Then $\Gamma \vdash \exists x\, \varphi(x)$ by Propositions 15.26, 16.26, 17.26 and 18.34, item (1). By Proposition 19.2(1), $\exists x\, \varphi(x) \in \Gamma$.

2. Exercise.

□

## 19.5 Lindenbaum's Lemma

We now prove a lemma that shows that any consistent set of sentences is contained in some set of sentences which is not just consistent, but also complete. The proof works by adding one sentence at a time, guaranteeing at each step that the set remains consistent. We do this so that for every $\varphi$, either $\varphi$ or $\neg\varphi$ gets added at some stage. The union of all stages in that construction then contains either $\varphi$ or its negation $\neg\varphi$ and is thus complete. It is also consistent, since we made sure at each stage not to introduce an inconsistency.

**Lemma 19.8** (Lindenbaum's Lemma)**.** *Every consistent set $\Gamma$ in a language $\mathcal{L}$ can be extended to a complete and consistent set $\Gamma^*$.*

*Proof.* Let $\Gamma$ be consistent. Let $\varphi_0$, $\varphi_1$, ... be an enumeration of all the sentences of $\mathcal{L}$. Define $\Gamma_0 = \Gamma$, and

$$\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{\varphi_n\} & \text{if } \Gamma_n \cup \{\varphi_n\} \text{ is consistent;} \\ \Gamma_n \cup \{\neg\varphi_n\} & \text{otherwise.} \end{cases}$$

Let $\Gamma^* = \bigcup_{n \geq 0} \Gamma_n$.

Each $\Gamma_n$ is consistent: $\Gamma_0$ is consistent by definition. If $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$, this is because the latter is consistent. If it isn't, $\Gamma_{n+1} = \Gamma_n \cup \{\neg\varphi_n\}$. We have to verify that $\Gamma_n \cup \{\neg\varphi_n\}$ is consistent. Suppose it's not. Then *both* $\Gamma_n \cup \{\varphi_n\}$ and $\Gamma_n \cup \{\neg\varphi_n\}$ are inconsistent. This means that $\Gamma_n$ would be inconsistent by

Propositions 15.20, 16.20, 17.20 and 18.28, contrary to the induction hypothesis.

For every $n$ and every $i < n$, $\Gamma_i \subseteq \Gamma_n$. This follows by a simple induction on $n$. For $n = 0$, there are no $i < 0$, so the claim holds automatically. For the inductive step, suppose it is true for $n$. We have $\Gamma_{n+1} = \Gamma_n \cup \{\varphi_n\}$ or $= \Gamma_n \cup \{\neg\varphi_n\}$ by construction. So $\Gamma_n \subseteq \Gamma_{n+1}$. If $i < n$, then $\Gamma_i \subseteq \Gamma_n$ by inductive hypothesis, and so $\subseteq \Gamma_{n+1}$ by transitivity of $\subseteq$.

From this it follows that every finite subset of $\Gamma^*$ is a subset of $\Gamma_n$ for some $n$, since each $\psi \in \Gamma^*$ not already in $\Gamma_0$ is added at some stage $i$. If $n$ is the last one of these, then all $\psi$ in the finite subset are in $\Gamma_n$. So, every finite subset of $\Gamma^*$ is consistent. By Propositions 15.17, 16.17, 17.17 and 18.21, $\Gamma^*$ is consistent.

Every sentence of $\mathrm{Frm}(\mathcal{L})$ appears on the list used to define $\Gamma^*$. If $\varphi_n \notin \Gamma^*$, then that is because $\Gamma_n \cup \{\varphi_n\}$ was inconsistent. But then $\neg\varphi_n \in \Gamma^*$, so $\Gamma^*$ is complete. □

## 19.6 Construction of a Model

Right now we are not concerned about $=$, i.e., we only want to show that a consistent set $\Gamma$ of sentences not containing $=$ is satisfiable. We first extend $\Gamma$ to a consistent, complete, and saturated set $\Gamma^*$. In this case, the definition of a model $\mathfrak{M}(\Gamma^*)$ is simple: We take the set of closed terms of $\mathcal{L}'$ as the domain. We assign every constant symbol to itself, and make sure that more generally, for every closed term $t$, $\mathrm{Val}^{\mathfrak{M}(\Gamma^*)}(t) = t$. The predicate symbols are assigned extensions in such a way that an atomic sentence is true in $\mathfrak{M}(\Gamma^*)$ iff it is in $\Gamma^*$. This will obviously make all the atomic sentences in $\Gamma^*$ true in $\mathfrak{M}(\Gamma^*)$. The rest are true provided the $\Gamma^*$ we start with is consistent, complete, and saturated.

**Definition 19.9** (Term model)**.** Let $\Gamma^*$ be a complete and consistent, saturated set of sentences in a language $\mathcal{L}$. The *term model* $\mathfrak{M}(\Gamma^*)$ of $\Gamma^*$ is the structure defined as follows:

1. The domain $|\mathfrak{M}(\Gamma^*)|$ is the set of all closed terms of $\mathcal{L}$.
2. The interpretation of a constant symbol $c$ is $c$ itself: $c^{\mathfrak{M}(\Gamma^*)} = c$.
3. The function symbol $f$ is assigned the function which, given as arguments the closed terms $t_1, \ldots, t_n$, has as value the closed term $f(t_1, \ldots, t_n)$:
$$f^{\mathfrak{M}(\Gamma^*)}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$$
4. If $R$ is an $n$-place predicate symbol, then
$$\langle t_1, \ldots, t_n \rangle \in R^{\mathfrak{M}(\Gamma^*)} \text{ iff } R(t_1, \ldots, t_n) \in \Gamma^*.$$

A structure $\mathfrak{M}$ may make an existentially quantified sentence $\exists x\, \varphi(x)$ true without there being an instance $\varphi(t)$ that it makes true. A structure $\mathfrak{M}$ may make all instances $\varphi(t)$ of a universally quantified sentence $\forall x\, \varphi(x)$ true, without making $\forall x\, \varphi(x)$ true. This is because in general not every element of $|\mathfrak{M}|$ is the value of a closed term ($\mathfrak{M}$ may not be covered). This is the reason the satisfaction relation is defined via variable assignments. However, for our term model $\mathfrak{M}(\Gamma^*)$ this wouldn't be necessary—because it is covered. This is the content of the next result.

**Proposition 19.10.** *Let $\mathfrak{M}(\Gamma^*)$ be the term model of Definition 19.9.*

1. *$\mathfrak{M}(\Gamma^*) \vDash \exists x\, \varphi(x)$ iff $\mathfrak{M} \vDash \varphi(t)$ for at least one term $t$.*
2. *$\mathfrak{M}(\Gamma^*) \vDash \forall x\, \varphi(x)$ iff $\mathfrak{M} \vDash \varphi(t)$ for all terms $t$.*

*Proof.* 1. By Proposition 12.42, $\mathfrak{M}(\Gamma^*) \vDash \exists x\, \varphi(x)$ iff for at least one variable assignment $s$, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$. As $|\mathfrak{M}(\Gamma^*)|$ consists of the closed terms of $\mathcal{L}$, this is the case iff there is at least one closed term $t$ such that $s(x) = t$ and $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$. By Proposition 12.46, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(x)$ iff $\mathfrak{M}(\Gamma^*), s \vDash \varphi(t)$, where $s(x) = t$. By Proposition 12.41, $\mathfrak{M}(\Gamma^*), s \vDash \varphi(t)$ iff $\mathfrak{M}(\Gamma^*) \vDash \varphi(t)$, since $\varphi(t)$ is a sentence.

2. Exercise.

□

**Lemma 19.11** (Truth Lemma)**.** *Suppose $\varphi$ does not contain $=$. Then $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff $\varphi \in \Gamma^*$.*

*Proof.* We prove both directions simultaneously, and by induction on $\varphi$.

1. $\varphi \equiv \bot$: $\mathfrak{M}(\Gamma^*) \nvDash \bot$ by definition of satisfaction. On the other hand, $\bot \notin \Gamma^*$ since $\Gamma^*$ is consistent.
2. $\varphi \equiv R(t_1, \ldots, t_n)$: $\mathfrak{M}(\Gamma^*) \vDash R(t_1, \ldots, t_n)$ iff $\langle t_1, \ldots, t_n \rangle \in R^{\mathfrak{M}(\Gamma^*)}$ (by the definition of satisfaction) iff $R(t_1, \ldots, t_n) \in \Gamma^*$ (by the construction of $\mathfrak{M}(\Gamma^*)$).
3. $\varphi \equiv \neg\psi$: $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff $\mathfrak{M}(\Gamma^*) \nvDash \psi$ (by definition of satisfaction). By induction hypothesis, $\mathfrak{M}(\Gamma^*) \nvDash \psi$ iff $\psi \notin \Gamma^*$. Since $\Gamma^*$ is consistent and complete, $\psi \notin \Gamma^*$ iff $\neg\psi \in \Gamma^*$.
4. $\varphi \equiv \psi \wedge \chi$: exercise.
5. $\varphi \equiv \psi \vee \chi$: $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff at $\mathfrak{M}(\Gamma^*) \vDash \psi$ or $\mathfrak{M}(\Gamma^*) \vDash \chi$ (by definition of satisfaction) iff $\psi \in \Gamma^*$ or $\chi \in \Gamma^*$ (by induction hypothesis). This is the case iff $(\psi \vee \chi) \in \Gamma^*$ (by Proposition 19.2(3)).
6. $\varphi \equiv \psi \to \chi$: exercise.

7. $\varphi \equiv \forall x\, \psi(x)$: exercise.

8. $\varphi \equiv \exists x\, \psi(x)$: $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff $\mathfrak{M}(\Gamma^*) \vDash \psi(t)$ for at least one term $t$ (Proposition 19.10). By induction hypothesis, this is the case iff $\psi(t) \in \Gamma^*$ for at least one term $t$. By Proposition 19.7, this in turn is the case iff $\exists x\, \varphi(x) \in \Gamma^*$.

□

## 19.7 Identity

The construction of the term model given in the preceding section is enough to establish completeness for first-order logic for sets $\Gamma$ that do not contain $=$. The term model satisfies every $\varphi \in \Gamma^*$ which does not contain $=$ (and hence all $\varphi \in \Gamma$). It does not work, however, if $=$ is present. The reason is that $\Gamma^*$ then may contain a sentence $t = t'$, but in the term model the value of any term is that term itself. Hence, if $t$ and $t'$ are different terms, their values in the term model—i.e., $t$ and $t'$, respectively—are different, and so $t = t'$ is false. We can fix this, however, using a construction known as "factoring."

**Definition 19.12.** Let $\Gamma^*$ be a consistent and complete set of sentences in $\mathcal{L}$. We define the relation $\approx$ on the set of closed terms of $\mathcal{L}$ by

$$t \approx t' \quad \text{iff} \quad t = t' \in \Gamma^*$$

**Proposition 19.13.** *The relation $\approx$ has the following properties:*

1. *$\approx$ is reflexive.*

2. *$\approx$ is symmetric.*

3. *$\approx$ is transitive.*

4. *If $t \approx t'$, $f$ is a function symbol, and $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$ are terms, then*

$$f(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \approx f(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n).$$

5. *If $t \approx t'$, $R$ is a predicate symbol, and $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$ are terms, then*

$$R(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n) \in \Gamma^* \text{ iff } R(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n) \in \Gamma^*.$$

*Proof.* Since $\Gamma^*$ is consistent and complete, $t = t' \in \Gamma^*$ iff $\Gamma^* \vdash t = t'$. Thus it is enough to show the following:

1. $\Gamma^* \vdash t = t$ for all terms $t$.

2. If $\Gamma^* \vdash t = t'$ then $\Gamma^* \vdash t' = t$.
3. If $\Gamma^* \vdash t = t'$ and $\Gamma^* \vdash t' = t''$, then $\Gamma^* \vdash t = t''$.
4. If $\Gamma^* \vdash t = t'$, then

   $$\Gamma^* \vdash f(t_1, \ldots, t_{i-1}, t, t_{i+1}, , \ldots, t_n) = f(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n)$$

   for every $n$-place function symbol $f$ and terms $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$.
5. If $\Gamma^* \vdash t = t'$ and $\Gamma^* \vdash R(t_1, \ldots, t_{i-1}, t, t_{i+1}, \ldots, t_n)$, then $\Gamma^* \vdash R(t_1, \ldots, t_{i-1}, t', t_{i+1}, \ldots, t_n)$ for every $n$-place predicate symbol $R$ and terms $t_1, \ldots, t_{i-1}, t_{i+1}, \ldots, t_n$.

□

**Definition 19.14.** Suppose $\Gamma^*$ is a consistent and complete set in a language $\mathcal{L}$, $t$ is a term, and $\approx$ as in the previous definition. Then:

$$[t]_\approx = \{t' : t' \in \mathrm{Trm}(\mathcal{L}), t \approx t'\}$$

and $\mathrm{Trm}(\mathcal{L})/_\approx = \{[t]_\approx : t \in \mathrm{Trm}(\mathcal{L})\}$.

**Definition 19.15.** Let $\mathfrak{M} = \mathfrak{M}(\Gamma^*)$ be the term model for $\Gamma^*$. Then $\mathfrak{M}/_\approx$ is the following structure:

1. $|\mathfrak{M}/_\approx| = \mathrm{Trm}(\mathcal{L})/_\approx$.
2. $c^{\mathfrak{M}/_\approx} = [c]_\approx$
3. $f^{\mathfrak{M}/_\approx}([t_1]_\approx, \ldots, [t_n]_\approx) = [f(t_1, \ldots, t_n)]_\approx$
4. $\langle [t_1]_\approx, \ldots, [t_n]_\approx \rangle \in R^{\mathfrak{M}/_\approx}$ iff $\mathfrak{M} \vDash R(t_1, \ldots, t_n)$.

Note that we have defined $f^{\mathfrak{M}/_\approx}$ and $R^{\mathfrak{M}/_\approx}$ for elements of $\mathrm{Trm}(\mathcal{L})/_\approx$ by referring to them as $[t]_\approx$, i.e., via *representatives* $t \in [t]_\approx$. We have to make sure that these definitions do not depend on the choice of these representatives, i.e., that for some other choices $t'$ which determine the same equivalence classes ($[t]_\approx = [t']_\approx$), the definitions yield the same result. For instance, if $R$ is a one-place predicate symbol, the last clause of the definition says that $[t]_\approx \in R^{\mathfrak{M}/_\approx}$ iff $\mathfrak{M} \vDash R(t)$. If for some other term $t'$ with $t \approx t'$, $\mathfrak{M} \nvDash R(t)$, then the definition would require $[t']_\approx \notin R^{\mathfrak{M}/_\approx}$. If $t \approx t'$, then $[t]_\approx = [t']_\approx$, but we can't have both $[t]_\approx \in R^{\mathfrak{M}/_\approx}$ and $[t]_\approx \notin R^{\mathfrak{M}/_\approx}$. However, Proposition 19.13 guarantees that this cannot happen.

**Proposition 19.16.** *$\mathfrak{M}/_\approx$ is well defined, i.e., if $t_1, \ldots, t_n$, $t'_1, \ldots, t'_n$ are terms, and $t_i \approx t'_i$ then*

1. *$[f(t_1, \ldots, t_n)]_\approx = [f(t'_1, \ldots, t'_n)]_\approx$, i.e.,*

   $$f(t_1, \ldots, t_n) \approx f(t'_1, \ldots, t'_n)$$

   *and*

2. $\mathfrak{M} \vDash R(t_1, \ldots, t_n)$ *iff* $\mathfrak{M} \vDash R(t'_1, \ldots, t'_n)$, *i.e.,*

$$R(t_1, \ldots, t_n) \in \Gamma^* \text{ iff } R(t'_1, \ldots, t'_n) \in \Gamma^*.$$

*Proof.* Follows from Proposition 19.13 by induction on $n$. □

**Lemma 19.17.** $\mathfrak{M}/{\approx} \vDash \varphi$ *iff* $\varphi \in \Gamma^*$ *for all sentences* $\varphi$.

*Proof.* By induction on $\varphi$, just as in the proof of Lemma 19.11. The only case that needs additional attention is when $\varphi \equiv t = t'$.

$$\begin{aligned} \mathfrak{M}/{\approx} \vDash t = t' &\text{ iff } [t]_\approx = [t']_\approx \text{ (by definition of } \mathfrak{M}/{\approx}) \\ &\text{iff } t \approx t' \text{ (by definition of } [t]_\approx) \\ &\text{iff } t = t' \in \Gamma^* \text{ (by definition of } \approx). \end{aligned}$$

□

Note that while $\mathfrak{M}(\Gamma^*)$ is always enumerable and infinite, $\mathfrak{M}/{\approx}$ may be finite, since it may turn out that there are only finitely many classes $[t]_\approx$. This is to be expected, since $\Gamma$ may contain sentences which require any structure in which they are true to be finite. For instance, $\forall x \, \forall y \, x = y$ is a consistent sentence, but is satisfied only in structures with a domain that contains exactly one element.

## 19.8 The Completeness Theorem

Let's combine our results: we arrive at the completeness theorem.

**Theorem 19.18** (Completeness Theorem)**.** *Let* $\Gamma$ *be a set of sentences. If* $\Gamma$ *is consistent, it is satisfiable.*

*Proof.* Suppose $\Gamma$ is consistent. By Lemma 19.6, there is a saturated consistent set $\Gamma' \supseteq \Gamma$. By Lemma 19.8, there is a $\Gamma^* \supseteq \Gamma'$ which is consistent and complete. Since $\Gamma' \subseteq \Gamma^*$, for each sentence $\varphi$, $\Gamma^*$ contains a sentence of the form $\exists x \, \varphi \to \varphi(c)$ and so $\Gamma^*$ is saturated. If $\Gamma$ does not contain $=$, then by Lemma 19.11, $\mathfrak{M}(\Gamma^*) \vDash \varphi$ iff $\varphi \in \Gamma^*$. From this it follows in particular that for all $\varphi \in \Gamma$, $\mathfrak{M}(\Gamma^*) \vDash \varphi$, so $\Gamma$ is satisfiable. If $\Gamma$ does contain $=$, then by Lemma 19.17, $\mathfrak{M}/{\approx} \vDash \varphi$ iff $\varphi \in \Gamma^*$ for all sentences $\varphi$. In particular, $\mathfrak{M}/{\approx} \vDash \varphi$ for all $\varphi \in \Gamma$, so $\Gamma$ is satisfiable. □

**Corollary 19.19** (Completeness Theorem, Second Version)**.** *For all* $\Gamma$ *and* $\varphi$ *sentences: if* $\Gamma \vDash \varphi$ *then* $\Gamma \vdash \varphi$.

*Proof.* Note that the $\Gamma$'s in Corollary 19.19 and Theorem 19.18 are universally quantified. To make sure we do not confuse ourselves, let us restate Theorem 19.18 using a different variable: for any set of sentences $\Delta$, if $\Delta$ is consistent, it is satisfiable. By contraposition, if $\Delta$ is not satisfiable, then $\Delta$ is inconsistent. We will use this to prove the corollary.

Suppose that $\Gamma \models \varphi$. Then $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable by Proposition 12.51. Taking $\Gamma \cup \{\neg\varphi\}$ as our $\Delta$, the previous version of Theorem 19.18 gives us that $\Gamma \cup \{\neg\varphi\}$ is inconsistent. By Propositions 15.19, 16.19, 17.19 and 18.27, $\Gamma \vdash \varphi$. □

## 19.9 The Compactness Theorem

One important consequence of the completeness theorem is the compactness theorem. The compactness theorem states that if each *finite* subset of a set of sentences is satisfiable, the entire set is satisfiable—even if the set itself is infinite. This is far from obvious. There is nothing that seems to rule out, at first glance at least, the possibility of there being infinite sets of sentences which are contradictory, but the contradiction only arises, so to speak, from the infinite number. The compactness theorem says that such a scenario can be ruled out: there are no unsatisfiable infinite sets of sentences each finite subset of which is satisfiable. Like the completeness theorem, it has a version related to entailment: if an infinite set of sentences entails something, already a finite subset does.

**Definition 19.20.** A set $\Gamma$ of formulas is *finitely satisfiable* if and only if every finite $\Gamma_0 \subseteq \Gamma$ is satisfiable.

**Theorem 19.21** (Compactness Theorem)**.** *The following hold for any sentences $\Gamma$ and $\varphi$:*

1. *$\Gamma \models \varphi$ iff there is a finite $\Gamma_0 \subseteq \Gamma$ such that $\Gamma_0 \models \varphi$.*
2. *$\Gamma$ is satisfiable if and only if it is finitely satisfiable.*

*Proof.* We prove (2). If $\Gamma$ is satisfiable, then there is a structure $\mathfrak{M}$ such that $\mathfrak{M} \models \varphi$ for all $\varphi \in \Gamma$. Of course, this $\mathfrak{M}$ also satisfies every finite subset of $\Gamma$, so $\Gamma$ is finitely satisfiable.

Now suppose that $\Gamma$ is finitely satisfiable. Then every finite subset $\Gamma_0 \subseteq \Gamma$ is satisfiable. By soundness (Corollaries 16.29, 15.31, 17.31 and 18.38), every finite subset is consistent. Then $\Gamma$ itself must be consistent by Propositions 15.17, 16.17, 17.17 and 18.21. By completeness (Theorem 19.18), since $\Gamma$ is consistent, it is satisfiable. □

**Example 19.22.** In every model $\mathfrak{M}$ of a theory $\Gamma$, each term $t$ of course picks out an element of $|\mathfrak{M}|$. Can we guarantee that it is also true that every element

of $|\mathfrak{M}|$ is picked out by some term or other? In other words, are there theories $\Gamma$ all models of which are covered? The compactness theorem shows that this is not the case if $\Gamma$ has infinite models. Here's how to see this: Let $\mathfrak{M}$ be an infinite model of $\Gamma$, and let $c$ be a constant symbol not in the language of $\Gamma$. Let $\Delta$ be the set of all sentences $c \neq t$ for $t$ a term in the language $\mathcal{L}$ of $\Gamma$, i.e.,

$$\Delta = \{c \neq t : t \in \mathrm{Trm}(\mathcal{L})\}.$$

A finite subset of $\Gamma \cup \Delta$ can be written as $\Gamma' \cup \Delta'$, with $\Gamma' \subseteq \Gamma$ and $\Delta' \subseteq \Delta$. Since $\Delta'$ is finite, it can contain only finitely many terms. Let $a \in |\mathfrak{M}|$ be an element of $|\mathfrak{M}|$ not picked out by any of them, and let $\mathfrak{M}'$ be the structure that is just like $\mathfrak{M}$, but also $c^{\mathfrak{M}'} = a$. Since $a \neq \mathrm{Val}^{\mathfrak{M}}(t)$ for all $t$ occuring in $\Delta'$, $\mathfrak{M}' \vDash \Delta'$. Since $\mathfrak{M} \vDash \Gamma$, $\Gamma' \subseteq \Gamma$, and $c$ does not occur in $\Gamma$, also $\mathfrak{M}' \vDash \Gamma'$. Together, $\mathfrak{M}' \vDash \Gamma' \cup \Delta'$ for every finite subset $\Gamma' \cup \Delta'$ of $\Gamma \cup \Delta$. So every finite subset of $\Gamma \cup \Delta$ is satisfiable. By compactness, $\Gamma \cup \Delta$ itself is satisfiable. So there are models $\mathfrak{M} \vDash \Gamma \cup \Delta$. Every such $\mathfrak{M}$ is a model of $\Gamma$, but is not covered, since $\mathrm{Val}^{\mathfrak{M}}(c) \neq \mathrm{Val}^{\mathfrak{M}}(t)$ for all terms $t$ of $\mathcal{L}$.

**Example 19.23.** Consider a language $\mathcal{L}$ containing the predicate symbol $<$, constant symbols $0$, $1$, and function symbols $+$, $\times$, $-$, $\div$. Let $\Gamma$ be the set of all sentences in this language true in $\mathfrak{Q}$ with domain $\mathbb{Q}$ and the obvious interpretations. $\Gamma$ is the set of all sentences of $\mathcal{L}$ true about the rational numbers. Of course, in $\mathbb{Q}$ (and even in $\mathbb{R}$), there are no numbers which are greater than 0 but less than $1/k$ for all $k \in \mathbb{Z}^+$. Such a number, if it existed, would be an *infinitesimal:* non-zero, but infinitely small. The compactness theorem shows that there are models of $\Gamma$ in which infinitesimals exist: Let $\Delta$ be $\{0 < c\} \cup \{c < (1 \div \overline{k}) : k \in \mathbb{Z}^+\}$ (where $\overline{k} = (1 + (1 + \cdots + (1 + 1)\ldots))$ with $k$ 1's). For any finite subset $\Delta_0$ of $\Delta$ there is a $K$ such that all the sentences $c < \overline{k}$ in $\Delta_0$ have $k < K$. If we expand $\mathfrak{Q}$ to $\mathfrak{Q}'$ with $c^{\mathfrak{Q}'} = 1/K$ we have that $\mathfrak{Q}' \vDash \Gamma \cup \Delta_0$, and so $\Gamma \cup \Delta$ is finitely satisfiable (Exercise: prove this in detail). By compactness, $\Gamma \cup \Delta$ is satisfiable. Any model $\mathfrak{S}$ of $\Gamma \cup \Delta$ contains an infinitesimal, namely $c^{\mathfrak{S}}$.

**Example 19.24.** We know that first-order logic with identity predicate can express that the size of the domain must have some minimal size: The sentence $\varphi_{\geq n}$ (which says "there are at least $n$ distinct objects") is true only in structures where $|\mathfrak{M}|$ has at least $n$ objects. So if we take

$$\Delta = \{\varphi_{\geq n} : n \geq 1\}$$

then any model of $\Delta$ must be infinite. Thus, we can guarantee that a theory only has infinite models by adding $\Delta$ to it: the models of $\Gamma \cup \Delta$ are all and only the infinite models of $\Gamma$.

So first-order logic can express infinitude. The compactness theorem shows that it cannot express finitude, however. For suppose some set of sentences $\Lambda$

were satisfied in all and only finite structures. Then $\Delta \cup \Lambda$ is finitely satisfiable. Why? Suppose $\Delta' \cup \Lambda' \subseteq \Delta \cup \Lambda$ is finite with $\Delta' \subseteq \Delta$ and $\Lambda' \subseteq \Lambda$. Let $n$ be the largest number such that $\varphi_{\geq n} \in \Delta'$. $\Lambda$, being satisfied in all finite structures, has a model $\mathfrak{M}$ with finitely many but $\geq n$ elements. But then $\mathfrak{M} \vDash \Delta' \cup \Lambda'$. By compactness, $\Delta \cup \Lambda$ has an infinite model, contradicting the assumption that $\Lambda$ is satisfied only in finite structures.

## 19.10 A Direct Proof of the Compactness Theorem

We can prove the Compactness Theorem directly, without appealing to the Completeness Theorem, using the same ideas as in the proof of the completeness theorem. In the proof of the Completeness Theorem we started with a consistent set $\Gamma$ of sentences, expanded it to a consistent, saturated, and complete set $\Gamma^*$ of sentences, and then showed that in the term model $\mathfrak{M}(\Gamma^*)$ constructed from $\Gamma^*$, all sentences of $\Gamma$ are true, so $\Gamma$ is satisfiable.

We can use the same method to show that a finitely satisfiable set of sentences is satisfiable. We just have to prove the corresponding versions of the results leading to the truth lemma where we replace "consistent" with "finitely satisfiable."

**Proposition 19.25.** *Suppose $\Gamma$ is complete and finitely satisfiable. Then:*

1. *$(\varphi \wedge \psi) \in \Gamma$ iff both $\varphi \in \Gamma$ and $\psi \in \Gamma$.*
2. *$(\varphi \vee \psi) \in \Gamma$ iff either $\varphi \in \Gamma$ or $\psi \in \Gamma$.*
3. *$(\varphi \to \psi) \in \Gamma$ iff either $\varphi \notin \Gamma$ or $\psi \in \Gamma$.*

**Lemma 19.26.** *Every finitely satisfiable set $\Gamma$ can be extended to a saturated finitely satisfiable set $\Gamma'$.*

**Proposition 19.27.** *Suppose $\Gamma$ is complete, finitely satisfiable, and saturated.*

1. *$\exists x\, \varphi(x) \in \Gamma$ iff $\varphi(t) \in \Gamma$ for at least one closed term $t$.*
2. *$\forall x\, \varphi(x) \in \Gamma$ iff $\varphi(t) \in \Gamma$ for all closed terms $t$.*

**Lemma 19.28.** *Every finitely satisfiable set $\Gamma$ can be extended to a complete and finitely satisfiable set $\Gamma^*$.*

**Theorem 19.29** (Compactness)**.** *$\Gamma$ is satisfiable if and only if it is finitely satisfiable.*

*Proof.* If $\Gamma$ is satisfiable, then there is a structure $\mathfrak{M}$ such that $\mathfrak{M} \vDash \varphi$ for all $\varphi \in \Gamma$. Of course, this $\mathfrak{M}$ also satisfies every finite subset of $\Gamma$, so $\Gamma$ is finitely satisfiable.

Now suppose that $\Gamma$ is finitely satisfiable. By Lemma 19.26, there is a finitely satisfiable, saturated set $\Gamma' \supseteq \Gamma$. By Lemma 19.28, $\Gamma'$ can be extended

to a complete and finitely satisfiable set $\Gamma^*$, and $\Gamma^*$ is still saturated. Construct the term model $\mathfrak{M}(\Gamma^*)$ as in Definition 19.9. Note that Proposition 19.10 did not rely on the fact that $\Gamma^*$ is consistent (or complete or saturated, for that matter), but just on the fact that $\mathfrak{M}(\Gamma^*)$ is covered. The proof of the Truth Lemma (Lemma 19.11) goes through if we replace references to Proposition 19.2 and Proposition 19.7 by references to Proposition 19.25 and Proposition 19.27 □

## 19.11 The Löwenheim-Skolem Theorem

The Löwenheim-Skolem Theorem says that if a theory has an infinite model, then it also has a model that is at most denumerable. An immediate consequene of this fact is that first-order logic cannot express that the size of a structure is non-enumerable: any sentence or set of sentences satisfied in all non-enumerable structures is also satisfied in some enumerable structure.

**Theorem 19.30.** *If $\Gamma$ is consistent then it has an enumerable model, i.e., it is satisfiable in a structure whose domain is either finite or denumerable.*

*Proof.* If $\Gamma$ is consistent, the structure $\mathfrak{M}$ delivered by the proof of the completeness theorem has a domain $|\mathfrak{M}|$ that is no larger than the set of the terms of the language $\mathcal{L}$. So $\mathfrak{M}$ is at most denumerable. □

**Theorem 19.31.** *If $\Gamma$ is consistent set of sentences in the language of first-order logic without identity, then it has a denumerable model, i.e., it is satisfiable in a structure whose domain is infinite and enumerable.*

*Proof.* If $\Gamma$ is consistent and contains no sentences in which identity appears, then the structure $\mathfrak{M}$ delivered by the proof of the completness theorem has a domain $|\mathfrak{M}|$ identical to the set of terms of the language $\mathcal{L}'$. So $\mathfrak{M}$ is denumerable, since $\mathrm{Trm}(\mathcal{L}')$ is. □

**Example 19.32** (Skolem's Paradox)**.** Zermelo-Fraenkel set theory **ZFC** is a very powerful framework in which practically all mathematical statements can be expressed, including facts about the sizes of sets. So for instance, **ZFC** can prove that the set $\mathbb{R}$ of real numbers is non-enumerable, it can prove Cantor's Theorem that the power set of any set is larger than the set itself, etc. If **ZFC** is consistent, its models are all infinite, and moreover, they all contain elements about which the theory says that they are non-enumerable, such as the element that makes true the theorem of **ZFC** that the power set of the natural numbers exists. By the Löwenheim-Skolem Theorem, **ZFC** also has enumerable models—models that contain "non-enumerable" sets but which themselves are enumerable.

## Problems

**Problem 19.1.** Complete the proof of Proposition 19.2.

**Problem 19.2.** Complete the proof of Proposition 19.10.

**Problem 19.3.** Complete the proof of Lemma 19.11.

**Problem 19.4.** Complete the proof of Proposition 19.13.

**Problem 19.5.** Use Corollary 19.19 to prove Theorem 19.18, thus showing that the two formulations of the completeness theorem are equivalent.

**Problem 19.6.** In order for a derivation system to be complete, its rules must be strong enough to prove every unsatisfiable set inconsistent. Which of the rules of derivation were necessary to prove completeness? Are any of these rules not used anywhere in the proof? In order to answer these questions, make a list or diagram that shows which of the rules of derivation were used in which results that lead up to the proof of Theorem 19.18. Be sure to note any tacit uses of rules in these proofs.

**Problem 19.7.** Prove (1) of Theorem 19.21.

**Problem 19.8.** In the standard model of arithmetic $\mathfrak{N}$, there is no element $k \in |\mathfrak{N}|$ which satisfies every formula $\overline{n} < x$ (where $\overline{n}$ is $0'^{\cdots'}$ with $n$ $'$s). Use the compactness theorem to show that the set of sentences in the language of arithmetic which are true in the standard model of arithmetic $\mathfrak{N}$ are also true in a structure $\mathfrak{N}'$ that contains an element which *does* satisfy every formula $\overline{n} < x$.

**Problem 19.9.** Prove Proposition 19.25. Avoid the use of $\vdash$.

**Problem 19.10.** Prove Lemma 19.26. (Hint: The crucial step is to show that if $\Gamma_n$ is finitely satisfiable, so is $\Gamma_n \cup \{\theta_n\}$, without any appeal to derivations or consistency.)

**Problem 19.11.** Prove Proposition 19.27.

**Problem 19.12.** Prove Lemma 19.28. (Hint: the crucial step is to show that if $\Gamma_n$ is finitely satisfiable, then either $\Gamma_n \cup \{\varphi_n\}$ or $\Gamma_n \cup \{\neg\varphi_n\}$ is finitely satisfiable.)

**Problem 19.13.** Write out the complete proof of the Truth Lemma (Lemma 19.11) in the version required for the proof of Theorem 19.29.

# Chapter 20

# Beyond First-order Logic

This chapter, adapted from Jeremy Avigad's logic notes, gives the briefest of glimpses into which other logical systems there are. It is intended as a chapter suggesting further topics for study in a course that does not cover them. Each one of the topics mentioned here will—hopefully—eventually receive its own part-level treatment in the Open Logic Project.

## 20.1 Overview

First-order logic is not the only system of logic of interest: there are many extensions and variations of first-order logic. A logic typically consists of the formal specification of a language, usually, but not always, a deductive system, and usually, but not always, an intended semantics. But the technical use of the term raises an obvious question: what do logics that are not first-order logic have to do with the word "logic," used in the intuitive or philosophical sense? All of the systems described below are designed to model reasoning of some form or another; can we say what makes them logical?

No easy answers are forthcoming. The word "logic" is used in different ways and in different contexts, and the notion, like that of "truth," has been analyzed from numerous philosophical stances. For example, one might take the goal of logical reasoning to be the determination of which statements are necessarily true, true a priori, true independent of the interpretation of the nonlogical terms, true by virtue of their form, or true by linguistic convention; and each of these conceptions requires a good deal of clarification. Even if one restricts one's attention to the kind of logic used in mathematics, there is little agreement as to its scope. For example, in the *Principia Mathematica*, Russell and Whitehead tried to develop mathematics on the basis of logic, in the *logicist* tradition begun by Frege. Their system of logic was a form of higher-type

logic similar to the one described below. In the end they were forced to introduce axioms which, by most standards, do not seem purely logical (notably, the axiom of infinity, and the axiom of reducibility), but one might nonetheless hold that some forms of higher-order reasoning should be accepted as logical. In contrast, Quine, whose ontology does not admit "propositions" as legitimate objects of discourse, argues that second-order and higher-order logic are really manifestations of set theory in sheep's clothing; in other words, systems involving quantification over predicates are not purely logical.

For now, it is best to leave such philosophical issues for a rainy day, and simply think of the systems below as formal idealizations of various kinds of reasoning, logical or otherwise.

## 20.2 Many-Sorted Logic

In first-order logic, variables and quantifiers range over a single domain. But it is often useful to have multiple (disjoint) domains: for example, you might want to have a domain of numbers, a domain of geometric objects, a domain of functions from numbers to numbers, a domain of abelian groups, and so on.

Many-sorted logic provides this kind of framework. One starts with a list of "sorts"—the "sort" of an object indicates the "domain" it is supposed to inhabit. One then has variables and quantifiers for each sort, and (usually) an identity predicate for each sort. Functions and relations are also "typed" by the sorts of objects they can take as arguments. Otherwise, one keeps the usual rules of first-order logic, with versions of the quantifier-rules repeated for each sort.

For example, to study international relations we might choose a language with two sorts of objects, French citizens and German citizens. We might have a unary relation, "drinks wine," for objects of the first sort; another unary relation, "eats wurst," for objects of the second sort; and a binary relation, "forms a multinational married couple," which takes two arguments, where the first argument is of the first sort and the second argument is of the second sort. If we use variables $a, b, c$ to range over French citizens and $x, y, z$ to range over German citizens, then

$$\forall a \forall x[(MarriedTo(a, x) \rightarrow (DrinksWine(a) \vee \neg EatsWurst(x))]]$$

asserts that if any French person is married to a German, either the French person drinks wine or the German doesn't eat wurst.

Many-sorted logic can be embedded in first-order logic in a natural way, by lumping all the objects of the many-sorted domains together into one first-order domain, using unary predicate symbols to keep track of the sorts, and relativizing quantifiers. For example, the first-order language corresponding to the example above would have unary predicate symbolss "$German$" and

"$French$," in addition to the other relations described, with the sort requirements erased. A sorted quantifier $\forall x\, \varphi$, where $x$ is a variable of the German sort, translates to

$$\forall x\, (German(x) \to \varphi).$$

We need to add axioms that insure that the sorts are separate—e.g., $\forall x\, \neg(German(x) \wedge French(x))$—as well as axioms that guarantee that "drinks wine" only holds of objects satisfying the predicate $French(x)$, etc. With these conventions and axioms, it is not difficult to show that many-sorted sentences translate to first-order sentences, and many-sorted derivations translate to first-order derivations. Also, many-sorted structures "translate" to corresponding first-order structures and vice-versa, so we also have a completeness theorem for many-sorted logic.

## 20.3 Second-Order logic

The language of second-order logic allows one to quantify not just over a domain of individuals, but over relations on that domain as well. Given a first-order language $\mathcal{L}$, for each $k$ one adds variables $R$ which range over $k$-ary relations, and allows quantification over those variables. If $R$ is a variable for a $k$-ary relation, and $t_1, \ldots, t_k$ are ordinary (first-order) terms, $R(t_1, \ldots, t_k)$ is an atomic formula. Otherwise, the set of formulas is defined just as in the case of first-order logic, with additional clauses for second-order quantification. Note that we only have the identity predicate for first-order terms: if $R$ and $S$ are relation variables of the same arity $k$, we can define $R = S$ to be an abbreviation for

$$\forall x_1 \ldots \forall x_k\, (R(x_1, \ldots, x_k) \leftrightarrow S(x_1, \ldots, x_k)).$$

The rules for second-order logic simply extend the quantifier rules to the new second order variables. Here, however, one has to be a little bit careful to explain how these variables interact with the predicate symbols of $\mathcal{L}$, and with formulas of $\mathcal{L}$ more generally. At the bare minimum, relation variables count as terms, so one has inferences of the form

$$\varphi(R) \vdash \exists R\, \varphi(R)$$

But if $\mathcal{L}$ is the language of arithmetic with a constant relation symbol $<$, one would also expect the following inference to be valid:

$$x < y \vdash \exists R\, R(x, y)$$

or for a given formula $\varphi$,

$$\varphi(x_1, \ldots, x_k) \vdash \exists R\, R(x_1, \ldots, x_k)$$

More generally, we might want to allow inferences of the form

$$\varphi[\lambda \vec{x}.\, \psi(\vec{x})/R] \vdash \exists R\, \varphi$$

where $\varphi[\lambda \vec{x}.\, \psi(\vec{x})/R]$ denotes the result of replacing every atomic formula of the form $Rt_1, \ldots, t_k$ in $\varphi$ by $\psi(t_1, \ldots, t_k)$. This last rule is equivalent to having a *comprehension schema*, i.e., an axiom of the form

$$\exists R\, \forall x_1, \ldots, x_k\, (\varphi(x_1, \ldots, x_k) \leftrightarrow R(x_1, \ldots, x_k)),$$

one for each formula $\varphi$ in the second-order language, in which $R$ is not a free variable. (Exercise: show that if $R$ is allowed to occur in $\varphi$, this schema is inconsistent!)

When logicians refer to the "axioms of second-order logic" they usually mean the minimal extension of first-order logic by second-order quantifier rules together with the comprehension schema. But it is often interesting to study weaker subsystems of these axioms and rules. For example, note that in its full generality the axiom schema of comprehension is *impredicative*: it allows one to assert the existence of a relation $R(x_1, \ldots, x_k)$ that is "defined" by a formula with second-order quantifiers; and these quantifiers range over the set of all such relations—a set which includes $R$ itself! Around the turn of the twentieth century, a common reaction to Russell's paradox was to lay the blame on such definitions, and to avoid them in developing the foundations of mathematics. If one prohibits the use of second-order quantifiers in the formula $\varphi$, one has a *predicative* form of comprehension, which is somewhat weaker.

From the semantic point of view, one can think of a second-order structure as consisting of a first-order structure for the language, coupled with a set of relations on the domain over which the second-order quantifiers range (more precisely, for each $k$ there is a set of relations of arity $k$). Of course, if comprehension is included in the proof system, then we have the added requirement that there are enough relations in the "second-order part" to satisfy the comprehension axioms—otherwise the proof system is not sound! One easy way to insure that there are enough relations around is to take the second-order part to consist of *all* the relations on the first-order part. Such a structure is called *full*, and, in a sense, is really the "intended structure" for the language. If we restrict our attention to full structures we have what is known as the *full* second-order semantics. In that case, specifying a structure boils down to specifying the first-order part, since the contents of the second-order part follow from that implicitly.

To summarize, there is some ambiguity when talking about second-order logic. In terms of the proof system, one might have in mind either

1. A "minimal" second-order proof system, together with some comprehension axioms.

2. The "standard" second-order proof system, with full comprehension.

In terms of the semantics, one might be interested in either

1. The "weak" semantics, where a structure consists of a first-order part, together with a second-order part big enough to satisfy the comprehension axioms.

2. The "standard" second-order semantics, in which one considers full structures only.

When logicians do not specify the proof system or the semantics they have in mind, they are usually refering to the second item on each list. The advantage to using this semantics is that, as we will see, it gives us categorical descriptions of many natural mathematical structures; at the same time, the proof system is quite strong, and sound for this semantics. The drawback is that the proof system is *not* complete for the semantics; in fact, *no* effectively given proof system is complete for the full second-order semantics. On the other hand, we will see that the proof system *is* complete for the weakened semantics; this implies that if a sentence is not provable, then there is *some* structure, not necessarily the full one, in which it is false.

The language of second-order logic is quite rich. One can identify unary relations with subsets of the domain, and so in particular you can quantify over these sets; for example, one can express induction for the natural numbers with a single axiom

$$\forall R\,((R(\mathsf{0}) \wedge \forall x\,(R(x) \to R(x'))) \to \forall x\,R(x)).$$

If one takes the language of arithmetic to have symbols $\mathsf{0}, \prime, +, \times$ and $<$, one can add the following axioms to describe their behavior:

1. $\forall x\,\neg x' = \mathsf{0}$
2. $\forall x\,\forall y\,(s(x) = s(y) \to x = y)$
3. $\forall x\,(x + \mathsf{0}) = x$
4. $\forall x\,\forall y\,(x + y') = (x + y)'$
5. $\forall x\,(x \times \mathsf{0}) = \mathsf{0}$
6. $\forall x\,\forall y\,(x \times y') = ((x \times y) + x)$
7. $\forall x\,\forall y\,(x < y \leftrightarrow \exists z\,y = (x + z'))$

It is not difficult to show that these axioms, together with the axiom of induction above, provide a categorical description of the structure $\mathfrak{N}$, the standard model of arithmetic, provided we are using the full second-order semantics. Given any structure $\mathfrak{M}$ in which these axioms are true, define a function $f$

from $\mathbb{N}$ to the domain of $\mathfrak{M}$ using ordinary recursion on $\mathbb{N}$, so that $f(0) = \mathrm{o}^{\mathfrak{M}}$ and $f(x+1) = \prime^{\mathfrak{M}}(f(x))$. Using ordinary induction on $\mathbb{N}$ and the fact that axioms (1) and (2) hold in $\mathfrak{M}$, we see that $f$ is injective. To see that $f$ is surjective, let $P$ be the set of elements of $|\mathfrak{M}|$ that are in the range of $f$. Since $\mathfrak{M}$ is full, $P$ is in the second-order domain. By the construction of $f$, we know that $\mathrm{o}^{\mathfrak{M}}$ is in $P$, and that $P$ is closed under $\prime^{\mathfrak{M}}$. The fact that the induction axiom holds in $\mathfrak{M}$ (in particular, for $P$) guarantees that $P$ is equal to the entire first-order domain of $\mathfrak{M}$. This shows that $f$ is a bijection. Showing that $f$ is a homomorphism is no more difficult, using ordinary induction on $\mathbb{N}$ repeatedly.

In set-theoretic terms, a function is just a special kind of relation; for example, a unary function $f$ can be identified with a binary relation $R$ satisfying $\forall x\, \exists y\, R(x, y)$. As a result, one can quantify over functions too. Using the full semantics, one can then define the class of infinite structures to be the class of structures $\mathfrak{M}$ for which there is an injective function from the domain of $\mathfrak{M}$ to a proper subset of itself:

$$\exists f\, (\forall x\, \forall y\, (f(x) = f(y) \to x = y) \wedge \exists y\, \forall x\, f(x) \neq y).$$

The negation of this sentence then defines the class of finite structures.

In addition, one can define the class of well-orderings, by adding the following to the definition of a linear ordering:

$$\forall P\, (\exists x\, P(x) \to \exists x\, (P(x) \wedge \forall y\, (y < x \to \neg P(y)))).$$

This asserts that every non-empty set has a least element, modulo the identification of "set" with "one-place relation". For another example, one can express the notion of connectedness for graphs, by saying that there is no non-trivial separation of the vertices into disconnected parts:

$$\neg \exists A\, (\exists x\, A(x) \wedge \exists y\, \neg A(y) \wedge \forall w\, \forall z\, ((A(w) \wedge \neg A(z)) \to \neg R(w, z))).$$

For yet another example, you might try as an exercise to define the class of finite structures whose domain has even size. More strikingly, one can provide a categorical description of the real numbers as a complete ordered field containing the rationals.

In short, second-order logic is much more expressive than first-order logic. That's the good news; now for the bad. We have already mentioned that there is no effective proof system that is complete for the full second-order semantics. For better or for worse, many of the properties of first-order logic are absent, including compactness and the Löwenheim-Skolem theorems.

On the other hand, if one is willing to give up the full second-order semantics in terms of the weaker one, then the minimal second-order proof system is complete for this semantics. In other words, if we read $\vdash$ as "proves in the minimal system" and $\models$ as "logically implies in the weaker semantics", we can show that whenever $\Gamma \models \varphi$ then $\Gamma \vdash \varphi$. If one wants to include specific

comprehension axioms in the proof system, one has to restrict the semantics to second-order structures that satisfy these axioms: for example, if $\Delta$ consists of a set of comprehension axioms (possibly all of them), we have that if $\Gamma \cup \Delta \models \varphi$, then $\Gamma \cup \Delta \vdash \varphi$. In particular, if $\varphi$ is not provable using the comprehension axioms we are considering, then there is a model of $\neg\varphi$ in which these comprehension axioms nonetheless hold.

The easiest way to see that the completeness theorem holds for the weaker semantics is to think of second-order logic as a many-sorted logic, as follows. One sort is interpreted as the ordinary "first-order" domain, and then for each $k$ we have a domain of "relations of arity $k$." We take the language to have built-in relation symbols "$true_k(R, x_1, \ldots, x_k)$" which is meant to assert that $R$ holds of $x_1, \ldots, x_k$, where $R$ is a variable of the sort "$k$-ary relation" and $x_1$, $\ldots$, $x_k$ are objects of the first-order sort.

With this identification, the weak second-order semantics is essentially the usual semantics for many-sorted logic; and we have already observed that many-sorted logic can be embedded in first-order logic. Modulo the translations back and forth, then, the weaker conception of second-order logic is really a form of first-order logic in disguise, where the domain contains both "objects" and "relations" governed by the appropriate axioms.

## 20.4 Higher-Order logic

Passing from first-order logic to second-order logic enabled us to talk about sets of objects in the first-order domain, within the formal language. Why stop there? For example, third-order logic should enable us to deal with sets of sets of objects, or perhaps even sets which contain both objects and sets of objects. And fourth-order logic will let us talk about sets of objects of that kind. As you may have guessed, one can iterate this idea arbitrarily.

In practice, higher-order logic is often formulated in terms of functions instead of relations. (Modulo the natural identifications, this difference is inessential.) Given some basic "sorts" $A$, $B$, $C$, ...(which we will now call "types"), we can create new ones by stipulating

> If $\sigma$ and $\tau$ are finite types then so is $\sigma \to \tau$.

Think of types as syntactic "labels," which classify the objects we want in our domain; $\sigma \to \tau$ describes those objects that are functions which take objects of type $\sigma$ to objects of type $\tau$. For example, we might want to have a type $\Omega$ of truth values, "true" and "false," and a type $\mathbb{N}$ of natural numbers. In that case, you can think of objects of type $\mathbb{N} \to \Omega$ as unary relations, or subsets of $\mathbb{N}$; objects of type $\mathbb{N} \to \mathbb{N}$ are functions from natural numers to natural numbers; and objects of type $(\mathbb{N} \to \mathbb{N}) \to \mathbb{N}$ are "functionals," that is, higher-type functions that take functions to numbers.

As in the case of second-order logic, one can think of higher-order logic as a kind of many-sorted logic, where there is a sort for each type of object we want to consider. But it is usually clearer just to define the syntax of higher-type logic from the ground up. For example, we can define a set of finite types inductively, as follows:

1. $\mathbb{N}$ is a finite type.
2. If $\sigma$ and $\tau$ are finite types, then so is $\sigma \to \tau$.
3. If $\sigma$ and $\tau$ are finite types, so is $\sigma \times \tau$.

Intuitively, $\mathbb{N}$ denotes the type of the natural numbers, $\sigma \to \tau$ denotes the type of functions from $\sigma$ to $\tau$, and $\sigma \times \tau$ denotes the type of pairs of objects, one from $\sigma$ and one from $\tau$. We can then define a set of terms inductively, as follows:

1. For each type $\sigma$, there is a stock of variables $x, y, z, \ldots$ of type $\sigma$
2. $0$ is a term of type $\mathbb{N}$
3. $S$ (successor) is a term of type $\mathbb{N} \to \mathbb{N}$
4. If $s$ is a term of type $\sigma$, and $t$ is a term of type $\mathbb{N} \to (\sigma \to \sigma)$, then $R_{st}$ is a term of type $\mathbb{N} \to \sigma$
5. If $s$ is a term of type $\tau \to \sigma$ and $t$ is a term of type $\tau$, then $s(t)$ is a term of type $\sigma$
6. If $s$ is a term of type $\sigma$ and $x$ is a variable of type $\tau$, then $\lambda x.\, s$ is a term of type $\tau \to \sigma$.
7. If $s$ is a term of type $\sigma$ and $t$ is a term of type $\tau$, then $\langle s, t \rangle$ is a term of type $\sigma \times \tau$.
8. If $s$ is a term of type $\sigma \times \tau$ then $p_1(s)$ is a term of type $\sigma$ and $p_2(s)$ is a term of type $\tau$.

Intuitively, $R_{st}$ denotes the function defined recursively by

$$R_{st}(0) = s$$
$$R_{st}(x+1) = t(x, R_{st}(x)),$$

$\langle s, t \rangle$ denotes the pair whose first component is $s$ and whose second component is $t$, and $p_1(s)$ and $p_2(s)$ denote the first and second elements ("projections") of $s$. Finally, $\lambda x.\, s$ denotes the function $f$ defined by

$$f(x) = s$$

for any $x$ of type $\sigma$; so item (6) gives us a form of comprehension, enabling us to define functions using terms. Formulas are built up from identity predicate statements $s = t$ between terms of the same type, the usual propositional connectives, and higher-type quantification. One can then take the axioms of the system to be the basic equations governing the terms defined above, together with the usual rules of logic with quantifiers and identity predicate.

If one augments the finite type system with a type $\Omega$ of truth values, one has to include axioms which govern its use as well. In fact, if one is clever, one can get rid of complex formulas entirely, replacing them with terms of type $\Omega$! The proof system can then be modified accordingly. The result is essentially the *simple theory of types* set forth by Alonzo Church in the 1930s.

As in the case of second-order logic, there are different versions of higher-type semantics that one might want to use. In the full version, variables of type $\sigma \to \tau$ range over the set of *all* functions from the objects of type $\sigma$ to objects of type $\tau$. As you might expect, this semantics is too strong to admit a complete, effective proof system. But one can consider a weaker semantics, in which a structure consists of sets of elements $T_\tau$ for each type $\tau$, together with appropriate operations for application, projection, etc. If the details are carried out correctly, one can obtain completeness theorems for the kinds of proof systems described above.

Higher-type logic is attractive because it provides a framework in which we can embed a good deal of mathematics in a natural way: starting with $\mathbb{N}$, one can define real numbers, continuous functions, and so on. It is also particularly attractive in the context of intuitionistic logic, since the types have clear "constructive" intepretations. In fact, one can develop constructive versions of higher-type semantics (based on intuitionistic, rather than classical logic) that clarify these constructive interpretations quite nicely, and are, in many ways, more interesting than the classical counterparts.

## 20.5 Intuitionistic Logic

In constrast to second-order and higher-order logic, intuitionistic first-order logic represents a restriction of the classical version, intended to model a more "constructive" kind of reasoning. The following examples may serve to illustrate some of the underlying motivations.

Suppose someone came up to you one day and announced that they had determined a natural number $x$, with the property that if $x$ is prime, the Riemann hypothesis is true, and if $x$ is composite, the Riemann hypothesis is false. Great news! Whether the Riemann hypothesis is true or not is one of the big open questions of mathematics, and here they seem to have reduced the problem to one of calculation, that is, to the determination of whether a specific number is prime or not.

What is the magic value of $x$? They describe it as follows: $x$ is the natural number that is equal to 7 if the Riemann hypothesis is true, and 9 otherwise.

Angrily, you demand your money back. From a classical point of view, the description above does in fact determine a unique value of $x$; but what you really want is a value of $x$ that is given *explicitly*.

To take another, perhaps less contrived example, consider the following question. We know that it is possible to raise an irrational number to a rational power, and get a rational result. For example, $\sqrt{2}^2 = 2$. What is less clear is whether or not it is possible to raise an irrational number to an *irrational* power, and get a rational result. The following theorem answers this in the affirmative:

**Theorem 20.1.** *There are irrational numbers $a$ and $b$ such that $a^b$ is rational.*

*Proof.* Consider $\sqrt{2}^{\sqrt{2}}$. If this is rational, we are done: we can let $a = b = \sqrt{2}$. Otherwise, it is irrational. Then we have

$$(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^2 = 2,$$

which is certainly rational. So, in this case, let $a$ be $\sqrt{2}^{\sqrt{2}}$, and let $b$ be $\sqrt{2}$. □

Does this constitute a valid proof? Most mathematicians feel that it does. But again, there is something a little bit unsatisfying here: we have proved the existence of a pair of real numbers with a certain property, without being able to say *which* pair of numbers it is. It is possible to prove the same result, but in such a way that the pair $a$, $b$ *is* given in the proof: take $a = \sqrt{3}$ and $b = \log_3 4$. Then

$$a^b = \sqrt{3}^{\log_3 4} = 3^{1/2\cdot\log_3 4} = (3^{\log_3 4})^{1/2} = 4^{1/2} = 2,$$

since $3^{\log_3 x} = x$.

Intuitionistic logic is designed to model a kind of reasoning where moves like the one in the first proof are disallowed. Proving the existence of an $x$ satisfying $\varphi(x)$ means that you have to give a specific $x$, and a proof that it satisfies $\varphi$, like in the second proof. Proving that $\varphi$ or $\psi$ holds requires that you can prove one or the other.

Formally speaking, intuitionistic first-order logic is what you get if you omit restrict a proof system for first-order logic in a certain way. Similarly, there are intuitionistic versions of second-order or higher-order logic. From the mathematical point of view, these are just formal deductive systems, but, as already noted, they are intended to model a kind of mathematical reasoning. One can take this to be the kind of reasoning that is justified on a certain philosophical view of mathematics (such as Brouwer's intuitionism); one can take it to be a kind of mathematical reasoning which is more "concrete" and satisfying (along the lines of Bishop's constructivism); and one can argue

about whether or not the formal description captures the informal motivation. But whatever philosophical positions we may hold, we can study intuitionistic logic as a formally presented logic; and for whatever reasons, many mathematical logicians find it interesting to do so.

There is an informal constructive interpretation of the intuitionist connectives, usually known as the Brouwer-Heyting-Kolmogorov interpretation. It runs as follows: a proof of $\varphi \wedge \psi$ consists of a proof of $\varphi$ paired with a proof of $\psi$; a proof of $\varphi \vee \psi$ consists of either a proof of $\varphi$, or a proof of $\psi$, where we have explicit information as to which is the case; a proof of $\varphi \to \psi$ consists of a procedure, which transforms a proof of $\varphi$ to a proof of $\psi$; a proof of $\forall x\, \varphi(x)$ consists of a procedure which returns a proof of $\varphi(x)$ for any value of $x$; and a proof of $\exists x\, \varphi(x)$ consists of a value of $x$, together with a proof that this value satisfies $\varphi$. One can describe the interpretation in computational terms known as the "Curry-Howard isomorphism" or the "formulas-as-types paradigm": think of a formula as specifying a certain kind of data type, and proofs as computational objects of these data types that enable us to see that the corresponding formula is true.

Intuitionistic logic is often thought of as being classical logic "minus" the law of the excluded middle. This following theorem makes this more precise.

**Theorem 20.2.** *Intuitionistically, the following axiom schemata are equivalent:*

1. $(\varphi \to \bot) \to \neg\varphi$.

2. $\varphi \vee \neg\varphi$

3. $\neg\neg\varphi \to \varphi$

Obtaining instances of one schema from either of the others is a good exercise in intuitionistic logic.

The first deductive systems for intuitionistic propositional logic, put forth as formalizations of Brouwer's intuitionism, are due, independently, to Kolmogorov, Glivenko, and Heyting. The first formalization of intuitionistic first-order logic (and parts of intuitionist mathematics) is due to Heyting. Though a number of classically valid schemata are not intuitionistically valid, many are.

The *double-negation translation* describes an important relationship between classical and intuitionist logic. It is defined inductively follows (think of $\varphi^N$

as the "intuitionist" translation of the classical formula $\varphi$):

$$\begin{aligned}
\varphi^N &\equiv \neg\neg\varphi \quad \text{for atomic formulas } \varphi \\
(\varphi \wedge \psi)^N &\equiv (\varphi^N \wedge \psi^N) \\
(\varphi \vee \psi)^N &\equiv \neg\neg(\varphi^N \vee \psi^N) \\
(\varphi \to \psi)^N &\equiv (\varphi^N \to \psi^N) \\
(\forall x\, \varphi)^N &\equiv \forall x\, \varphi^N \\
(\exists x\, \varphi)^N &\equiv \neg\neg\exists x\, \varphi^N
\end{aligned}$$

Kolmogorov and Glivenko had versions of this translation for propositional logic; for predicate logic, it is due to Gödel and Gentzen, independently. We have

**Theorem 20.3.** *1. $\varphi \leftrightarrow \varphi^N$ is provable classically*

*2. If $\varphi$ is provable classically, then $\varphi^N$ is provable intuitionistically.*

We can now envision the following dialogue. Classical mathematician: "I've proved $\varphi$!" Intuitionist mathematician: "Your proof isn't valid. What you've really proved is $\varphi^N$." Classical mathematician: "Fine by me!" As far as the classical mathematician is concerned, the intuitionist is just splitting hairs, since the two are equivalent. But the intuitionist insists there is a difference.

Note that the above translation concerns pure logic only; it does not address the question as to what the appropriate *nonlogical* axioms are for classical and intuitionistic mathematics, or what the relationship is between them. But the following slight extension of the theorem above provides some useful information:

**Theorem 20.4.** *If $\Gamma$ proves $\varphi$ classically, $\Gamma^N$ proves $\varphi^N$ intuitionistically.*

In other words, if $\varphi$ is provable from some hypotheses classically, then $\varphi^N$ is provable from their double-negation translations.

To show that a sentence or propositional formula is intuitionistically valid, all you have to do is provide a proof. But how can you show that it is not valid? For that purpose, we need a semantics that is sound, and preferrably complete. A semantics due to Kripke nicely fits the bill.

We can play the same game we did for classical logic: define the semantics, and prove soundness and completeness. It is worthwhile, however, to note the following distinction. In the case of classical logic, the semantics was the "obvious" one, in a sense implicit in the meaning of the connectives. Though one can provide some intuitive motivation for Kripke semantics, the latter does not offer the same feeling of inevitability. In addition, the notion of a classical structure is a natural mathematical one, so we can either take the notion of a structure to be a tool for studying classical first-order logic, or take

classical first-order logic to be a tool for studying mathematical structures. In contrast, Kripke structures can only be viewed as a logical construct; they don't seem to have independent mathematical interest.

A Kripke structure $\mathfrak{M} = \langle W, R, V \rangle$ for a propositional language consists of a set $W$, partial order $R$ on $W$ with a least element, and an "monotone" assignment of propositional variables to the elements of $W$. The intuition is that the elements of $W$ represent "worlds," or "states of knowledge"; an element $v \geq u$ represents a "possible future state" of $u$; and the propositional variables assigned to $u$ are the propositions that are known to be true in state $u$. The forcing relation $\mathfrak{M}, w \Vdash \varphi$ then extends this relationship to arbitrary formulas in the language; read $\mathfrak{M}, w \Vdash \varphi$ as "$\varphi$ is true in state $w$." The relationship is defined inductively, as follows:

1. $\mathfrak{M}, w \Vdash p_i$ iff $p_i$ is one of the propositional variables assigned to $w$.
2. $\mathfrak{M}, w \nVdash \bot$.
3. $\mathfrak{M}, w \Vdash (\varphi \wedge \psi)$ iff $\mathfrak{M}, w \Vdash \varphi$ and $\mathfrak{M}, w \Vdash \psi$.
4. $\mathfrak{M}, w \Vdash (\varphi \vee \psi)$ iff $\mathfrak{M}, w \Vdash \varphi$ or $\mathfrak{M}, w \Vdash \psi$.
5. $\mathfrak{M}, w \Vdash (\varphi \to \psi)$ iff, whenever $w' \geq w$ and $\mathfrak{M}, w' \Vdash \varphi$, then $\mathfrak{M}, w' \Vdash \psi$.

It is a good exercise to try to show that $\neg(p \wedge q) \to (\neg p \vee \neg q)$ is not intuitionistically valid, by cooking up a Kripke structure that provides a counterexample.

## 20.6 Modal Logics

Consider the following example of a conditional sentence:

> If Jeremy is alone in that room, then he is drunk and naked and dancing on the chairs.

This is an example of a conditional assertion that may be materially true but nonetheless misleading, since it seems to suggest that there is a stronger link between the antecedent and conclusion other than simply that either the antecedent is false or the consequent true. That is, the wording suggests that the claim is not only true in this particular world (where it may be trivially true, because Jeremy is not alone in the room), but that, moreover, the conclusion *would have* been true *had* the antecedent been true. In other words, one can take the assertion to mean that the claim is true not just in this world, but in any "possible" world; or that it is *necessarily* true, as opposed to just true in this particular world.

Modal logic was designed to make sense of this kind of necessity. One obtains modal propositional logic from ordinary propositional logic by adding a

box operator; which is to say, if $\varphi$ is a formula, so is $\Box\varphi$. Intuitively, $\Box\varphi$ asserts that $\varphi$ is *necessarily* true, or true in any possible world. $\Diamond\varphi$ is usually taken to be an abbreviation for $\neg\Box\neg\varphi$, and can be read as asserting that $\varphi$ is *possibly* true. Of course, modality can be added to predicate logic as well.

Kripke structures can be used to provide a semantics for modal logic; in fact, Kripke first designed this semantics with modal logic in mind. Rather than restricting to partial orders, more generally one has a set of "possible worlds," $P$, and a binary "accessibility" relation $R(x, y)$ between worlds. Intuitively, $R(p, q)$ asserts that the world $q$ is compatible with $p$; i.e., if we are "in" world $p$, we have to entertain the possibility that the world could have been like $q$.

Modal logic is sometimes called an "intensional" logic, as opposed to an "extensional" one. The intended semantics for an extensional logic, like classical logic, will only refer to a single world, the "actual" one; while the semantics for an "intensional" logic relies on a more elaborate ontology. In addition to structureing necessity, one can use modality to structure other linguistic constructions, reinterpreting $\Box$ and $\Diamond$ according to the application. For example:

1. In provability logic, $\Box\varphi$ is read "$\varphi$ is provable" and $\Diamond\varphi$ is read "$\varphi$ is consistent."
2. In epistemic logic, one might read $\Box\varphi$ as "I know $\varphi$" or "I believe $\varphi$."
3. In temporal logic, one can read $\Box\varphi$ as "$\varphi$ is always true" and $\Diamond\varphi$ as "$\varphi$ is sometimes true."

One would like to augment logic with rules and axioms dealing with modality. For example, the system **S4** consists of the ordinary axioms and rules of propositional logic, together with the following axioms:

$$\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$$
$$\Box\varphi \to \varphi$$
$$\Box\varphi \to \Box\Box\varphi$$

as well as a rule, "from $\varphi$ conclude $\Box\varphi$." **S5** adds the following axiom:

$$\Diamond\varphi \to \Box\Diamond\varphi$$

Variations of these axioms may be suitable for different applications; for example, S5 is usually taken to characterize the notion of logical necessity. And the nice thing is that one can usually find a semantics for which the proof system is sound and complete by restricting the accessibility relation in the Kripke structures in natural ways. For example, **S4** corresponds to the class of Kripke structures in which the accessibility relation is reflexive and transitive. **S5** corresponds to the class of Kripke structures in which the accessibility

relation is *universal*, which is to say that every world is accessible from every other; so $\Box\varphi$ holds if and only if $\varphi$ holds in every world.

## 20.7 Other Logics

As you may have gathered by now, it is not hard to design a new logic. You too can create your own a syntax, make up a deductive system, and fashion a semantics to go with it. You might have to be a bit clever if you want the proof system to be complete for the semantics, and it might take some effort to convince the world at large that your logic is truly interesting. But, in return, you can enjoy hours of good, clean fun, exploring your logic's mathematical and computational properties.

Recent decades have witnessed a veritable explosion of formal logics. Fuzzy logic is designed to model reasoning about vague properties. Probabilistic logic is designed to model reasoning about uncertainty. Default logics and nonmonotonic logics are designed to model defeasible forms of reasoning, which is to say, "reasonable" inferences that can later be overturned in the face of new information. There are epistemic logics, designed to model reasoning about knowledge; causal logics, designed to model reasoning about causal relationships; and even "deontic" logics, which are designed to model reasoning about moral and ethical obligations. Depending on whether the primary motivation for introducing these systems is philosophical, mathematical, or computational, you may find such creatures studies under the rubric of mathematical logic, philosophical logic, artificial intelligence, cognitive science, or elsewhere.

The list goes on and on, and the possibilities seem endless. We may never attain Leibniz' dream of reducing all of human reason to calculation—but that can't stop us from trying.

# Part IV

# Model Theory

Material on model theory is incomplete and experimental. It is currently simply an adaptation of Aldo Antonelli's notes on model theory, less those topics covered in the part on first-order logic (theories, completeness, compactness). It requires much more introduction, motivation, and explanation, as well as exercises, to be useful for a textbook. Andy Arana is at planning to work on this part specifically (issue #65).

# Chapter 21

# Basics of Model Theory

## 21.1 Reducts and Expansions

Often it is useful or necessary to compare languages which have symbols in common, as well as structures for these languages. The most comon case is when all the symbols in a language $\mathcal{L}$ are also part of a language $\mathcal{L}'$, i.e., $\mathcal{L} \subseteq \mathcal{L}'$. An $\mathcal{L}$-structure $\mathfrak{M}$ can then always be expanded to an $\mathcal{L}'$-structure by adding interpretations of the additional symbols while leaving the interpretations of the common symbols the same. On the other hand, from an $\mathcal{L}'$-structure $\mathfrak{M}'$ we can obtain an $\mathcal{L}$-structure simpy by "forgetting" the interpretations of the symbols that do not occur in $\mathcal{L}$.

**Definition 21.1.** Suppose $\mathcal{L} \subseteq \mathcal{L}'$, $\mathfrak{M}$ is an $\mathcal{L}$-structure and $\mathfrak{M}'$ is an $\mathcal{L}'$-structure. $\mathfrak{M}$ is the *reduct* of $\mathfrak{M}'$ to $\mathcal{L}$, and $\mathfrak{M}'$ is an *expansion* of $\mathfrak{M}$ to $\mathcal{L}'$ iff

1. $|\mathfrak{M}| = |\mathfrak{M}'|$
2. For every constant symbol $c \in \mathcal{L}$, $c^{\mathfrak{M}} = c^{\mathfrak{M}'}$.
3. For every function symbol $f \in \mathcal{L}$, $f^{\mathfrak{M}} = f^{\mathfrak{M}'}$.
4. For every predicate symbol $P \in \mathcal{L}$, $P^{\mathfrak{M}} = P^{\mathfrak{M}'}$.

**Proposition 21.2.** *If an $\mathcal{L}$-structure $\mathfrak{M}$ is a reduct of an $\mathcal{L}'$-structure $\mathfrak{M}'$, then for all $\mathcal{L}$-sentences $\varphi$,*

$$\mathfrak{M} \vDash \varphi \text{ iff } \mathfrak{M}' \vDash \varphi.$$

*Proof.* Exercise. □

**Definition 21.3.** When we have an $\mathcal{L}$-structure $\mathfrak{M}$, and $\mathcal{L}' = \mathcal{L} \cup \{P\}$ is the expansion of $\mathcal{L}$ obtained by adding a single $n$-place predicate symbol $P$, and $R \subseteq |\mathfrak{M}|^n$ is an $n$-place relation, then we write $(\mathfrak{M}, R)$ for the expansion $\mathfrak{M}'$ of $\mathfrak{M}$ with $P^{\mathfrak{M}'} = R$.

## 21.2 Substructures

The domain of a structure $\mathfrak{M}$ may be a subset of another $\mathfrak{M}'$. But we should obviously only consider $\mathfrak{M}$ a "part" of $\mathfrak{M}'$ if not only $|\mathfrak{M}| \subseteq |\mathfrak{M}'|$, but $\mathfrak{M}$ and $\mathfrak{M}'$ "agree" in how they interpret the symbols of the language at least on the shared part $|\mathfrak{M}|$.

**Definition 21.4.** Given structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$, we say that $\mathfrak{M}$ is a *substructure* of $\mathfrak{M}'$, and $\mathfrak{M}'$ an *extension* of $\mathfrak{M}$, written $\mathfrak{M} \subseteq \mathfrak{M}'$, iff

1. $|\mathfrak{M}| \subseteq |\mathfrak{M}'|$,
2. For each constant $c \in \mathcal{L}$, $c^{\mathfrak{M}} = c^{\mathfrak{M}'}$;
3. For each $n$-place predicate symbol $f \in \mathcal{L}$ $f^{\mathfrak{M}}(a_1, \ldots, a_n) = f^{\mathfrak{M}'}(a_1, \ldots, a_n)$ for all $a_1, \ldots, a_n \in |\mathfrak{M}|$.
4. For each $n$-place predicate symbol $R \in \mathcal{L}$, $\langle a_1, \ldots, a_n \rangle \in R^{\mathfrak{M}}$ iff $\langle a_1, \ldots, a_n \rangle \in R^{\mathfrak{M}'}$ for all $a_1, \ldots, a_n \in |\mathfrak{M}|$.

*Remark* 1. If the language contains no constant or function symbols, then any $N \subseteq |\mathfrak{M}|$ determines a substructure $\mathfrak{N}$ of $\mathfrak{M}$ with domain $|\mathfrak{N}| = N$ by putting $R^{\mathfrak{N}} = R^{\mathfrak{M}} \cap N^n$.

## 21.3 Overspill

**Theorem 21.5.** *If a set $\Gamma$ of sentences has arbitrarily large finite models, then it has an infinite model.*

*Proof.* Expand the language of $\Gamma$ by adding countably many new constants $c_0$, $c_1$, ... and consider the set $\Gamma \cup \{c_i \neq c_j : i \neq j\}$. To say that $\Gamma$ has arbitrarily large finite models means that for every $m > 0$ there is $n \geq m$ such that $\Gamma$ has a model of cardinality $n$. This implies that $\Gamma \cup \{c_i \neq c_j : i \neq j\}$ is finitely satisfiable. By compactness, $\Gamma \cup \{c_i \neq c_j : i \neq j\}$ has a model $\mathfrak{M}$ whose domain must be infinite, since it satisfies all inequalities $c_i \neq c_j$. □

**Proposition 21.6.** *There is no sentence $\varphi$ of any first-order language that is true in a structure $\mathfrak{M}$ if and only if the domain $|\mathfrak{M}|$ of the structure is infinite.*

*Proof.* If there were such a $\varphi$, its negation $\neg\varphi$ would be true in all and only the finite structures, and it would therefore have arbitrarily large finite models but it would lack an infinite model, contradicting Theorem 21.5. □

## 21.4 Isomorphic Structures

First-order structures can be alike in one of two ways. One way in which the can be alike is that they make the same sentences true. We call such structures *elementarily equivalent*. But structures can be very different and still make the same sentences true—for instance, one can be enumerable and the other not. This is because there are lots of features of a structure that cannot be expressed in first-order languages, either because the language is not rich enough, or because of fundamental limitations of first-order logic such as the Löwenheim-Skolem theorem. So another, stricter, aspect in which structures can be alike is if they are fundamentally the same, in the sense that they only differ in the objects that make them up, but not in their structural features. A way of making this precise is by the notion of an *isomorphism*.

**Definition 21.7.** Given two structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$, we say that $\mathfrak{M}$ is *elementarily equivalent to* $\mathfrak{M}'$, written $\mathfrak{M} \equiv \mathfrak{M}'$, if and only if for every sentence $\varphi$ of $\mathcal{L}$, $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}' \vDash \varphi$.

**Definition 21.8.** Given two structures $\mathfrak{M}$ and $\mathfrak{M}'$ for the same language $\mathcal{L}$, we say that $\mathfrak{M}$ is *isomorphic to* $\mathfrak{M}'$, written $\mathfrak{M} \simeq \mathfrak{M}'$, if and only if there is a function $h\colon |\mathfrak{M}| \to |\mathfrak{M}'|$ such that:

1. $h$ is injective: if $h(x) = h(y)$ then $x = y$;
2. $h$ is surjective: for every $y \in |\mathfrak{M}'|$ there is $x \in |\mathfrak{M}|$ such that $h(x) = y$;
3. for every constant symbol $c$: $h(c^{\mathfrak{M}}) = c^{\mathfrak{M}'}$;
4. for every $n$-place predicate symbol $P$:

$$\langle a_1, \ldots, a_n \rangle \in P^{\mathfrak{M}} \quad \text{iff} \quad \langle h(a_1), \ldots, h(a_n) \rangle \in P^{\mathfrak{M}'};$$

5. for every $n$-place function symbol $f$:

$$h(f^{\mathfrak{M}}(a_1, \ldots, a_n)) = f^{\mathfrak{M}'}(h(a_1), \ldots, h(a_n)).$$

**Theorem 21.9.** *If $\mathfrak{M} \simeq \mathfrak{M}'$ then $\mathfrak{M} \equiv \mathfrak{M}'$.*

*Proof.* Let $h$ be an isomorphism of $\mathfrak{M}$ onto $\mathfrak{M}'$. For any assignment $s$, $h \circ s$ is the composition of $h$ and $s$, i.e., the assignment in $\mathfrak{M}'$ such that $(h \circ s)(x) = h(s(x))$. By induction on $t$ and $\varphi$ one can prove the stronger claims:

a. $h(\mathrm{Val}^{\mathfrak{M}}_s(t)) = \mathrm{Val}^{\mathfrak{M}'}_{h \circ s}(t)$.

b. $\mathfrak{M}, s \vDash \varphi$ iff $\mathfrak{M}', h \circ s \vDash \varphi$.

The first is proved by induction on the complexity of $t$.

1. If $t \equiv c$, then $\mathrm{Val}_s^{\mathfrak{M}}(c) = c^{\mathfrak{M}}$ and $\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(c) = c^{\mathfrak{M}'}$. Thus, $h(\mathrm{Val}_s^{\mathfrak{M}}(t)) = h(c^{\mathfrak{M}}) = c^{\mathfrak{M}'}$ (by (3) of Definition 21.8) $= \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t)$.

2. If $t \equiv x$, then $\mathrm{Val}_s^{\mathfrak{M}}(x) = s(x)$ and $\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(x) = h(s(x))$. Thus, $h(\mathrm{Val}_s^{\mathfrak{M}}(x)) = h(s(x)) = \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(x)$.

3. If $t \equiv f(t_1, \ldots, t_n)$, then

$$\mathrm{Val}_s^{\mathfrak{M}}(t) = f^{\mathfrak{M}}(\mathrm{Val}_s^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}_s^{\mathfrak{M}}(t_n)) \quad \text{and}$$
$$\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t) = f^{\mathfrak{M}}(\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_1), \ldots, \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_n)).$$

   The induction hypothesis is that for each $i$, $h(\mathrm{Val}_s^{\mathfrak{M}}(t_i)) = \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_i)$. So,

$$\begin{aligned} h(\mathrm{Val}_s^{\mathfrak{M}}(t)) &= h(f^{\mathfrak{M}}(\mathrm{Val}_s^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}_s^{\mathfrak{M}}(t_n)) \\ &= h(f^{\mathfrak{M}}(\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_1), \ldots, \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_n)) && (21.1) \\ &= f^{\mathfrak{M}'}(\mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_1), \ldots, \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t_n)) && (21.2) \\ &= \mathrm{Val}_{h\circ s}^{\mathfrak{M}'}(t) \end{aligned}$$

   Here, eq. (21.1) follows by induction hypothesis and eq. (21.2) by (5) of Definition 21.8.

Part (2) is left as an exercise.

If $\varphi$ is a sentence, the assignments $s$ and $h \circ s$ are irrelevant, and we have $\mathfrak{M} \vDash \varphi$ iff $\mathfrak{M}' \vDash \varphi$. □

**Definition 21.10.** An *automorphism* of a structure $\mathfrak{M}$ is an isomorphism of $\mathfrak{M}$ onto itself.

## 21.5 The Theory of a Structure

Every structure $\mathfrak{M}$ makes some sentences true, and some false. The set of all the sentences it makes true is called its *theory*. That set is in fact a theory, since anything it entails must be true in all its models, including $\mathfrak{M}$.

**Definition 21.11.** Given a structure $\mathfrak{M}$, the *theory* of $\mathfrak{M}$ is the set $\mathrm{Th}(\mathfrak{M})$ of sentences that are true in $\mathfrak{M}$, i.e., $\mathrm{Th}(\mathfrak{M}) = \{\varphi : \mathfrak{M} \vDash \varphi\}$.

We also use the term "theory" informally to refer to sets of sentences having an intended interpretation, whether deductively closed or not.

**Proposition 21.12.** *For any $\mathfrak{M}$, $\mathrm{Th}(\mathfrak{M})$ is complete.*

*Proof.* For any sentence $\varphi$ either $\mathfrak{M} \vDash \varphi$ or $\mathfrak{M} \vDash \neg\varphi$, so either $\varphi \in \mathrm{Th}(\mathfrak{M})$ or $\neg\varphi \in \mathrm{Th}(\mathfrak{M})$. □

**Proposition 21.13.** *If* $\mathfrak{N} \models \varphi$ *for every* $\varphi \in \mathrm{Th}(\mathfrak{M})$*, then* $\mathfrak{M} \equiv \mathfrak{N}$.

*Proof.* Since $\mathfrak{N} \vDash \varphi$ for all $\varphi \in \mathrm{Th}(\mathfrak{M})$, $\mathrm{Th}(\mathfrak{M}) \subseteq \mathrm{Th}(\mathfrak{N})$. If $\mathfrak{N} \vDash \varphi$, then $\mathfrak{N} \nvDash \neg\varphi$, so $\neg\varphi \notin \mathrm{Th}(\mathfrak{M})$. Since $\mathrm{Th}(\mathfrak{M})$ is complete, $\varphi \in \mathrm{Th}(\mathfrak{M})$. So, $\mathrm{Th}(\mathfrak{N}) \subseteq \mathrm{Th}(\mathfrak{M})$, and we have $\mathfrak{M} \equiv \mathfrak{N}$. □

*Remark* 2. Consider $\mathfrak{R} = \langle \mathbb{R}, < \rangle$, the structure whose domain is the set $\mathbb{R}$ of the real numbers, in the language comprising only a 2-place predicate symbol interpreted as the $<$ relation over the reals. Clearly $\mathfrak{R}$ is non-enumerable; however, since $\mathrm{Th}(\mathfrak{R})$ is obviously consistent, by the Löwenheim-Skolem theorem it has an enumerable model, say $\mathfrak{S}$, and by Proposition 21.13, $\mathfrak{R} \equiv \mathfrak{S}$. Moreover, since $\mathfrak{R}$ and $\mathfrak{S}$ are not isomorphic, this shows that the converse of Theorem 21.9 fails in general.

## 21.6 Partial Isomorphisms

**Definition 21.14.** Given two structures $\mathfrak{M}$ and $\mathfrak{N}$, a *partial isomorphism* from $\mathfrak{M}$ to $\mathfrak{N}$ is a finite partial function $p$ taking arguments in $|\mathfrak{M}|$ and returning values in $|\mathfrak{N}|$, which satisfies the isomorphism conditions from Definition 21.8 on its domain:

1. $p$ is injective;
2. for every constant symbol $c$: if $p(c^{\mathfrak{M}})$ is defined, then $p(c^{\mathfrak{M}}) = c^{\mathfrak{N}}$;
3. for every $n$-place predicate symbol $P$: if $a_1, \ldots, a_n$ are in the domain of $p$, then $\langle a_1, \ldots, a_n \rangle \in P^{\mathfrak{M}}$ if and only if $\langle p(a_1), \ldots, p(a_n) \rangle \in P^{\mathfrak{N}}$;
4. for every $n$-place function symbol $f$: if $a_1, \ldots, a_n$ are in the domain of $p$, then $p(f^{\mathfrak{M}}(a_1, \ldots, a_n)) = f^{\mathfrak{N}}(p(a_1),\ dots, p(a_n))$.

That $p$ is finite means that $\mathrm{dom}(p)$ is finite.

Notice that the empty function $\emptyset$ is always a partial isomorphism between any two structures.

**Definition 21.15.** Two structures $\mathfrak{M}$ and $\mathfrak{N}$, are *partially isomorphic*, written $\mathfrak{M} \simeq_p \mathfrak{N}$, if and only if there is a non-empty set $I$ of partial isomorphisms between $\mathfrak{M}$ and $\mathfrak{N}$ satisfying the *back-and-forth* property:

1. (*Forth*) For every $p \in I$ and $a \in |\mathfrak{M}|$ there is $q \in I$ such that $p \subseteq q$ and $a$ is in the domain of $q$;
2. (*Back*) For every $p \in I$ and $b \in |\mathfrak{N}|$ there is $q \in I$ such that $p \subseteq q$ and $b$ is in the range of $q$.

**Theorem 21.16.** *If* $\mathfrak{M} \simeq_p \mathfrak{N}$ *and* $\mathfrak{M}$ *and* $\mathfrak{N}$ *are enumerable, then* $\mathfrak{M} \simeq \mathfrak{N}$.

*Proof.* Since $\mathfrak{M}$ and $\mathfrak{N}$ are enumerable, let $|\mathfrak{M}| = \{a_0, a_1, \ldots\}$ and $|\mathfrak{N}| = \{b_0, b_1, \ldots\}$. Starting with an arbitrary $p_0 \in I$, we define an increasing sequence of partial isomorphisms $p_0 \subseteq p_1 \subseteq p_2 \subseteq \cdots$ as follows:

1. if $n+1$ is odd, say $n = 2r$, then using the Forth property find a $p_{n+1} \in I$ such that $p_n \subseteq p_{n+1}$ and $a_r$ is in the domain of $p_{n+1}$;
2. if $n+1$ is even, say $n+1 = 2r$, then using the Back property find a $p_{n+1} \in I$ such that $p_n \subseteq p_{n+1}$ and $b_r$ is in the range of $p_{n+1}$.

If we now put:

$$p = \bigcup_{n \geq 0} p_n,$$

we have that $p$ is a an isomorphism between $\mathfrak{M}$ and $\mathfrak{N}$. □

**Theorem 21.17.** *Suppose $\mathfrak{M}$ and $\mathfrak{N}$ are structures for a purely relational language (a language containing only predicate symbols, and no function symbols or constants). Then if $\mathfrak{M} \simeq_p \mathfrak{N}$, also $\mathfrak{M} \equiv \mathfrak{N}$.*

*Proof.* By induction on formulas, one shows that if $a_1, \ldots, a_n$ and $b_1, \ldots, b_n$ are such that there is a partial isomorphism $p$ mapping each $a_i$ to $b_i$ and $s_1(x_i) = a_i$ and $s_2(x_i) = b_i$ (for $i = 1, \ldots, n$), then $\mathfrak{M}, s_1 \vDash \varphi$ if and only if $\mathfrak{N}, s_2 \vDash \varphi$. The case for $n = 0$ gives $\mathfrak{M} \equiv \mathfrak{N}$. □

*Remark* 3. If function symbols are present, the previous result is still true, but one needs to consider the isomorphism induced by $p$ between the substructure of $\mathfrak{M}$ generated by $a_1, \ldots, a_n$ and the substructure of $\mathfrak{N}$ generated by $b_1, \ldots, b_n$.

The previous result can be "broken down" into stages by establishing a connection between the number of nested quantifiers in a formula and how many times the relevant partial isomorphisms can be extended.

**Definition 21.18.** For any formula $\varphi$, the *quantifier rank* of $\varphi$, denoted by $\mathrm{qr}(\varphi) \in \mathbb{N}$, is recursively defined as the highest number of nested quantifiers in $\varphi$. Two structures $\mathfrak{M}$ and $\mathfrak{N}$ are *n-equivalent*, written $\mathfrak{M} \equiv_n \mathfrak{N}$, if they agree on all sentences of quantifier rank less than or equal to $n$.

**Proposition 21.19.** *Let $\mathcal{L}$ be a finite purely relational language, i.e., a language containing finitely many predicate symbols and constant symbols, and no function symbols. Then for each $n \in \mathbb{N}$ there are only finitely many first-order sentences in the language $\mathcal{L}$ that have quantifier rank no greater than $n$, up to logical equivalence.*

*Proof.* By induction on $n$. □

**Definition 21.20.** Given a structure $\mathfrak{M}$, let $|\mathfrak{M}|^{<\omega}$ be the set of all finite sequences over $|\mathfrak{M}|$. We use $\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$ to range over finite sequences of elements. If $\mathbf{a} \in |\mathfrak{M}|^{<\omega}$ and $a \in |\mathfrak{M}|$, then $\mathbf{a}a$ represents the *concatenation* of $\mathbf{a}$ with $a$.

**Definition 21.21.** Given structures $\mathfrak{M}$ and $\mathfrak{N}$, we define relations $I_n \subseteq |\mathfrak{M}|^{<\omega} \times |\mathfrak{N}|^{<\omega}$ between sequences of equal length, by recursion on $n$ as follows:

1. $I_0(\mathbf{a}, \mathbf{b})$ if and only if $\mathbf{a}$ and $\mathbf{b}$ satisfy the same atomic formulas in $\mathfrak{M}$ and $\mathfrak{N}$; i.e., if $s_1(x_i) = a_i$ and $s_2(x_i) = b_i$ and $\varphi$ is atomic with all variables among $x_1, \ldots, x_n$, then $\mathfrak{M}, s_1 \vDash \varphi$ if and only if $\mathfrak{N}, s_2 \vDash \varphi$.

2. $I_{n+1}(\mathbf{a}, \mathbf{b})$ if and only if for every $a \in A$ there is a $b \in B$ such that $I_n(\mathbf{a}a, \mathbf{b}b)$, and vice-versa.

**Definition 21.22.** Write $\mathfrak{M} \approx_n \mathfrak{N}$ if $I_n(\Lambda, \Lambda)$ holds of $\mathfrak{M}$ and $\mathfrak{N}$ (where $\Lambda$ is the empty sequence).

**Theorem 21.23.** *Let $\mathcal{L}$ be a purely relational language. Then $I_n(\mathbf{a}, \mathbf{b})$ implies that for every $\varphi$ such that $\operatorname{qr}(\varphi) \leq n$, we have $\mathfrak{M}, \mathbf{a} \vDash \varphi$ if and only if $\mathfrak{N}, \mathbf{b} \vDash \varphi$ (where again $\mathbf{a}$ satisfies $\varphi$ if any $s$ such that $s(x_i) = a_i$ satisfies $\varphi$). Moreover, if $\mathcal{L}$ is finite, the converse also holds.*

*Proof.* The proof that $I_n(\mathbf{a}, \mathbf{b})$ implies that $\mathbf{a}$ and $\mathbf{b}$ satisfy the same formulas of quantifier rank no greater than $n$ is by an easy induction on $\varphi$. For the converse we proceed by induction on $n$, using Proposition 21.19, which ensures that for each $n$ there are at most finitely many non-equivalent formulas of that quantifier rank.

For $n = 0$ the hypothesis that $\mathbf{a}$ and $\mathbf{b}$ satisfy the same quantifier-free formulas gives that they satisfy the same atomic ones, so that $I_0(\mathbf{a}, \mathbf{b})$.

For the $n + 1$ case, suppose that $\mathbf{a}$ and $\mathbf{b}$ satisfy the same formulas of quantifier rank no greater than $n + 1$; in order to show that $I_{n+1}(\mathbf{a}, \mathbf{b})$ suffices to show that for each $a \in |\mathfrak{M}|$ there is a $b \in |\mathfrak{N}|$ such that $I_n(\mathbf{a}a, \mathbf{b}b)$, and by the inductive hypothesis again suffices to show that for each $a \in |\mathfrak{M}|$ there is a $b \in |\mathfrak{N}|$ such that $\mathbf{a}a$ and $\mathbf{b}b$ satisfy the same formulas of quantifier rank no greater than $n$.

Given $a \in |\mathfrak{M}|$, let $\tau_n^a$ be set of formulas $\psi(x, \mathbf{y})$ of rank no greater than $n$ satisfied by $\mathbf{a}a$ in $\mathfrak{M}$; $\tau_n^a$ is finite, so we can assume it is a single first-order formula. It follows that $\mathbf{a}$ satisfies $\exists x\, \tau_n^a(x, \mathbf{y})$, which has quantifier rank no greater than $n + 1$. By hypothesis $\mathbf{b}$ satisfies the same formula in $\mathfrak{N}$, so that there is a $b \in |\mathfrak{N}|$ such that $\mathbf{b}b$ satisfies $\tau_n^a$; in particular, $\mathbf{b}b$ satisfies the same formulas of quantifier rank no greater than $n$ as $\mathbf{a}a$. Similarly one shows that for every $b \in |\mathfrak{N}|$ there is $a \in |\mathfrak{M}|$ such that $\mathbf{a}a$ and $\mathbf{b}b$ satisfy the same formulas of quantifier rank no greater than $n$, which completes the proof. □

**Corollary 21.24.** *If $\mathfrak{M}$ and $\mathfrak{N}$ are purely relational structures in a finite language, then $\mathfrak{M} \approx_n \mathfrak{N}$ if and only if $\mathfrak{M} \equiv_n \mathfrak{N}$. In particular $\mathfrak{M} \equiv \mathfrak{N}$ if and only if for each $n$, $\mathfrak{M} \approx_n \mathfrak{N}$.*

## 21.7 Dense Linear Orders

**Definition 21.25.** A *dense linear ordering without endpoints* is a structure $\mathfrak{M}$ for the language containg a single 2-place predicate symbol $<$ satisfying the following sentences:

1. $\forall x\, x < x$;
2. $\forall x \forall y \forall z\, (x < y \to (y < z \to x < z))$;
3. $\forall x \forall y\, (x < y \vee x = y \vee y < x)$;
4. $\forall x \exists y\, x < y$;
5. $\forall x \exists y\, y < x$;
6. $\forall x \forall y\, (x < y \to \exists z\, (x < z \wedge z < y))$.

**Theorem 21.26.** *Any two enumerable dense linear orderings without endpoints are isomorphic.*

*Proof.* Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be enumerable dense linear orderings without endpoints, with $<_1 = <^{\mathfrak{M}_1}$ and $<_2 = <^{\mathfrak{M}_2}$, and let $\mathcal{I}$ be the set of all partial isomorphisms between them. $\mathcal{I}$ is not empty since at least $\emptyset \in \mathcal{I}$. We show that $\mathcal{I}$ satisfies the Back-and-Forth property. Then $\mathfrak{M}_1 \simeq_p \mathfrak{M}_2$, and the theorem follows by Theorem 21.16.

To show $\mathcal{I}$ satisifes the Forth property, let $p \in \mathcal{I}$ and let $p(a_i) = b_i$ for $i = 1, \ldots, n$, and without loss of generality suppose $a_1 <_1 a_2 <_1 \cdots <_1 a_n$. Given $a \in |\mathfrak{M}_1|$, find $b \in |\mathfrak{M}_2|$ as follows:

1. if $a <_2 a_1$ let $b \in |\mathfrak{M}_2|$ be such that $b <_2 b_1$;
2. if $a_n <_1 a$ let $b \in |\mathfrak{M}_2|$ be such that $b_n <_2 b$;
3. if $a_i <_1 a <_1 a_{i+1}$ for some $i$, then let $b \in |\mathfrak{M}_2|$ be such that $b_i <_2 b <_2 b_{i+1}$.

It is always possible to find a $b$ with the desired property since $\mathfrak{M}_2$ is a dense linear ordering without endpoints. Define $q = p \cup \{\langle a, b\rangle\}$ so that $q \in \mathcal{I}$ is the desired extension of $p$. This establishes the Forth property. The Back property is similar. So $\mathfrak{M}_1 \simeq_p \mathfrak{M}_2$; by Theorem 21.16, $\mathfrak{M}_1 \simeq \mathfrak{M}_2$. □

*Remark* 4. Let $\mathfrak{S}$ be any enumerable dense linear ordering without endpoints. Then (by Theorem 21.26) $\mathfrak{S} \simeq \mathfrak{Q}$, where $\mathfrak{Q} = (\mathbb{Q}, <)$ is the enumerable dense linear ordering having the set $\mathbb{Q}$ of the rational numbers as its domain. Now consider again the structure $\mathfrak{R} = (\mathbb{R}, <)$ from Remark 2. We saw that there is an enumerable structure $\mathfrak{S}$ such that $\mathfrak{R} \equiv \mathfrak{S}$. But $\mathfrak{S}$ is an enumerable dense linear ordering without endpoints, and so it is isomorphic (and hence elementarily equivalent) to the structure $\mathfrak{Q}$. By transitivity of elementary equivalence,

$\mathfrak{R} \equiv \mathfrak{Q}$. (We could have shown this directly by establishing $\mathfrak{R} \simeq_p \mathfrak{Q}$ by the same back-and-forth argument.)

## Problems

**Problem 21.1.** Prove Proposition 21.2.

**Problem 21.2.** Carry out the proof of (b) of Theorem 21.9 in detail. Make sure to note where each of the five properties characterizing isomorphisms of Definition 21.8 is used.

**Problem 21.3.** Show that for any structure $\mathfrak{M}$, if $X$ is a definable subset of $\mathfrak{M}$, and $h$ is an automorphism of $\mathfrak{M}$, then $X = \{h(x) : x \in X\}$ (i.e., $X$ is fixed under $h$).

**Problem 21.4.** Show in detail that $p$ as defined in Theorem 21.16 is in fact an isomorphism.

**Problem 21.5.** Complete the proof of Theorem 21.26 by verifying that $\mathcal{I}$ satisfies the Back property.

# Chapter 22

# Models of Arithmetic

## 22.1 Introduction

The *standard model* of aritmetic is the structure $\mathfrak{N}$ with $|\mathfrak{N}| = \mathbb{N}$ in which $\mathsf{o}$, $\mathsf{'}$, $+$, $\times$, and $<$ are interpreted as you would expect. That is, $\mathsf{o}$ is 0, $\mathsf{'}$ is the successor function, $+$ is interpeted as addition and $\times$ as multiplication of the numbers in $\mathbb{N}$. Specifically,

$$\begin{aligned}
\mathsf{o}^{\mathfrak{N}} &= 0 \\
\mathsf{'}^{\mathfrak{N}}(n) &= n + 1 \\
+^{\mathfrak{N}}(n, m) &= n + m \\
\times^{\mathfrak{N}}(n, m) &= nm
\end{aligned}$$

Of course, there are structures for $\mathcal{L}_A$ that have domains other than $\mathbb{N}$. For instance, we can take $\mathfrak{M}$ with domain $|\mathfrak{M}| = \{a\}^*$ (the finite sequences of the single symbol $a$, i.e., $\emptyset$, $a$, $aa$, $aaa$, ...), and interpretations

$$\begin{aligned}
\mathsf{o}^{\mathfrak{M}} &= \emptyset \\
\mathsf{'}^{\mathfrak{M}}(s) &= s \frown a \\
+^{\mathfrak{M}}(n, m) &= a^{n+m} \\
\times^{\mathfrak{M}}(n, m) &= a^{nm}
\end{aligned}$$

These two structures are "essentially the same" in the sense that the only difference is the elements of the domains but not how the elements of the domains are related among each other by the interpretation functions. We say that the two structures are *isomorphic*.

It is an easy consequence of the compactness theorem that any theory true in $\mathfrak{N}$ also has models that are not isomorphic to $\mathfrak{N}$. Such structures are called *non-standard*. The interesting thing about them is that while the elements of a standard model (i.e., $\mathfrak{N}$, but also all structures isomorphic to it) are exhausted

by the values of the standard numerals $\overline{n}$, i.e.,

$$|\mathfrak{N}| = \{\mathrm{Val}^{\mathfrak{N}}(\overline{n}) : n \in \mathbb{N}\}$$

that isn't the case in non-standard models: if $\mathfrak{M}$ is non-standard, then there is at least one $x \in |\mathfrak{M}|$ such that $x \neq \mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for all $n$.

These non-standard elements are pretty neat: they are "infinite natural numbers." But their existence also explains, in a sense, the incompleteness phenomena. Cconsider an example, e.g., the consistency statement for Peano arithmetic, $\mathsf{Con}_{\mathbf{PA}}$, i.e., $\neg\exists x\, \mathsf{Prf}_{\mathbf{PA}}(x, \ulcorner\bot\urcorner)$. Since **PA** neither proves $\mathsf{Con}_{\mathbf{PA}}$ nor $\neg\mathsf{Con}_{\mathbf{PA}}$, either can be consistently added to **PA**. Since **PA** is consistent, $\mathfrak{N} \models \mathsf{Con}_{\mathbf{PA}}$, and consequently $\mathfrak{N} \not\models \neg\mathsf{Con}_{\mathbf{PA}}$. So $\mathfrak{N}$ is *not* a model of $\mathbf{PA} \cup \{\neg\mathsf{Con}_{\mathbf{PA}}\}$, and all its models must be nonstandard. Models of $\mathbf{PA} \cup \{\neg\mathsf{Con}_{\mathbf{PA}}\}$ must contain some element that serves as the witness that makes $\exists x\, \mathsf{Prf}_{\mathbf{PA}}(\ulcorner\bot\urcorner)$ true, i.e., a Gödel number of a derivation of a contradiction from **PA**. Such an element can't be standard—since $\mathbf{PA} \vdash \neg\mathsf{Prf}_{\mathbf{PA}}(\overline{n}, \ulcorner\bot\urcorner)$ for every $n$.

## 22.2 Standard Models of Arithmetic

The language of arithmetic $\mathcal{L}_A$ is obviously intended to be about numbers, specifically, about natural numbers. So, "the" standard model $\mathfrak{N}$ is special: it is the model we want to talk about. But in logic, we are often just interested in structural properties, and any two structures taht are isomorphic share those. So we can be a bit more liberal, and consider any structure that is isomorphic to $\mathfrak{N}$ "standard."

**Definition 22.1.** A structure for $\mathcal{L}_A$ is *standard* if it is isomorphic to $\mathfrak{N}$.

**Proposition 22.2.** *If a structure $\mathfrak{M}$ standard, its domain is the set of values of the standard numerals, i.e.,*

$$|\mathfrak{M}| = \{\mathrm{Val}^{\mathfrak{M}}(\overline{n}) : n \in \mathbb{N}\}$$

*Proof.* Clearly, every $\mathrm{Val}^{\mathfrak{M}}(\overline{n}) \in |\mathfrak{M}|$. We just have to show that every $x \in |\mathfrak{M}|$ is equal to $\mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for some $n$. Since $\mathfrak{M}$ is standard, it is isomorphic to $\mathfrak{N}$. Suppose $g\colon \mathbb{N} \to |\mathfrak{M}|$ is an isomorphism. Then $g(n) = g(\mathrm{Val}^{\mathfrak{N}}(\overline{n})) = \mathrm{Val}^{\mathfrak{M}}(\overline{n})$. But for every $x \in |\mathfrak{M}|$, there is an $n \in \mathbb{N}$ such that $g(n) = x$, since $g$ is surjective. □

If a structure $\mathfrak{M}$ for $\mathcal{L}_A$ is standard, the elements of its domain can all be named by the standard numerals $\overline{0}$, $\overline{1}$, $\overline{2}$, ..., i.e., the terms $\mathsf{o}$, $\mathsf{o}'$, $\mathsf{o}''$, etc. Of course, this does not mean that the elements of $|\mathfrak{M}|$ *are* the numbers, just that we can pick them out the same way we can pick out the numbers in $|\mathfrak{N}|$.

**Proposition 22.3.** *If $\mathfrak{M} \models \mathbf{Q}$, and $|\mathfrak{M}| = \{\mathrm{Val}^{\mathfrak{M}}(\overline{n}) : n \in \mathbb{N}\}$, then $\mathfrak{M}$ is standard.*

*Proof.* We have to show that $\mathfrak{M}$ is isomorphic to $\mathfrak{N}$. Consider the function $g\colon \mathbb{N} \to |\mathfrak{M}|$ defined by $g(n) = \mathrm{Val}^{\mathfrak{M}}(\overline{n})$. By the hypothesis, $g$ is surjective. It is also injective: $\mathbf{Q} \vdash \overline{n} \neq \overline{m}$ whenever $n \neq m$. Thus, since $\mathfrak{M} \models \mathbf{Q}$, $\mathfrak{M} \models \overline{n} \neq \overline{m}$, whenever $n \neq m$. Thus, if $n \neq m$, then $\mathrm{Val}^{\mathfrak{M}}(\overline{n}) \neq \mathrm{Val}^{\mathfrak{M}}(\overline{m})$, i.e., $g(n) \neq g(m)$.

We also have to verify that $g$ is an isomorphism.

1. We have $g(\mathsf{o}^{\mathfrak{N}}) = g(0)$ since, $\mathsf{o}^{\mathfrak{N}} = 0$. By definition of $g$, $g(0) = \mathrm{Val}^{\mathfrak{M}}(\overline{0})$. But $\overline{0}$ is just $\mathsf{o}$, and the value of a term which happens to be a constant symbol is given by what the structure assigns to that constant symbol, i.e., $\mathrm{Val}^{\mathfrak{M}}(\mathsf{o}) = \mathsf{o}^{\mathfrak{M}}$. So we have $g(\mathsf{o}^{\mathfrak{N}}) = \mathsf{o}^{\mathfrak{M}}$ as required.

2. $g(\mathsf{'}^{\mathfrak{N}}(n)) = g(n+1)$, since $\mathsf{'}$ in $\mathfrak{N}$ is the successor function on $\mathbb{N}$. Then, $g(n+1) = \mathrm{Val}^{\mathfrak{M}}(\overline{n+1})$ by definition of $g$. But $\overline{n+1}$ is the same term as $\overline{n}'$, so $\mathrm{Val}^{\mathfrak{M}}(\overline{n+1}) = \mathrm{Val}^{\mathfrak{M}}(\overline{n}')$. By the definition of the value function, this is $= \mathsf{'}^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\overline{n}))$. Since $\mathrm{Val}^{\mathfrak{M}}(\overline{n}) = g(n)$ we get $g(\mathsf{'}^{\mathfrak{N}}(n)) = \mathsf{'}^{\mathfrak{M}}(g(n))$.

3. $g(+^{\mathfrak{N}}(n,m)) = g(n+m)$, since $+$ in $\mathfrak{N}$ is the addition function on $\mathbb{N}$. Then, $g(n+m) = \mathrm{Val}^{\mathfrak{M}}(\overline{n+m})$ by definition of $g$. But $\mathbf{Q} \vdash \overline{n+m} = (\overline{n} + \overline{m})$, so $\mathrm{Val}^{\mathfrak{M}}(\overline{n+m}) = \mathrm{Val}^{\mathfrak{M}}(\overline{n} + \overline{m})$. By the definition of the value function, this is $= +^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\overline{n}), \mathrm{Val}^{\mathfrak{M}}(\overline{m}))$. Since $\mathrm{Val}^{\mathfrak{M}}(\overline{n}) = g(n)$ and $\mathrm{Val}^{\mathfrak{M}}(\overline{m}) = g(m)$, we get $g(+^{\mathfrak{N}}(n,m)) = +^{\mathfrak{M}}(g(n), g(m))$.

4. $g(\times^{\mathfrak{N}}(n,m)) = \times^{\mathfrak{M}}(g(n), g(m))$: Exercise.

5. $\langle n, m\rangle \in <^{\mathfrak{N}}$ iff $n < m$. If $n < m$, then $\mathbf{Q} \vdash \overline{n} < \overline{m}$, and also $\mathfrak{M} \models \overline{n} < \overline{m}$. Thus $\langle \mathrm{Val}^{\mathfrak{M}}(\overline{n}), \mathrm{Val}^{\mathfrak{M}}(\overline{m})\rangle \in <^{\mathfrak{M}}$, i.e., $\langle g(n), g(m)\rangle \in <^{\mathfrak{M}}$. If $n \not< m$, then $\mathbf{Q} \vdash \neg \overline{n} < \overline{m}$, and consequently $\mathfrak{M} \not\models \overline{n} < \overline{m}$. Thus, as before, $\langle g(n), g(m)\rangle \notin <^{\mathfrak{M}}$. Together, we get: $\langle n, m\rangle \in <^{\mathfrak{N}}$ iff $\langle g(n), g(m)\rangle \in <^{\mathfrak{M}}$.

□

The function $g$ is the most obvious way of defining a mapping from $\mathbb{N}$ to the domain of any other structure $\mathfrak{M}$ for $\mathcal{L}_A$, since every such $\mathfrak{M}$ contains elements named by $\overline{0}$, $\overline{1}$, $\overline{2}$, etc. So it isn't surprising that if $\mathfrak{M}$ makes at least some basic statements about the $\overline{n}$'s true in the same way that $\mathfrak{N}$ does, and $g$ is also bijective, then $g$ will turn into an isomorphism. In fact, if $|\mathfrak{M}|$ contains no elements other than what the $\overline{n}$'s name, it's the only one.

**Proposition 22.4.** *If $\mathfrak{M}$ is standard, then $g$ from the proof of Proposition 22.3 is the only isomorphism from $\mathfrak{N}$ to $\mathfrak{M}$.*

*Proof.* Suppose $h\colon \mathbb{N} \to |\mathfrak{M}|$ is an isomorphism between $\mathfrak{N}$ and $\mathfrak{M}$. We show that $g = h$ by induction on $n$. If $n = 0$, then $g(0) = \mathsf{o}^{\mathfrak{M}}$ by definition of $g$. But since $h$ is an isomorphism, $h(0) = h(\mathsf{o}^{\mathfrak{N}}) = \mathsf{o}^{\mathfrak{M}}$, so $g(0) = h(0)$.

Now consider the case for $n+1$. We have

$$\begin{aligned}
g(n+1) &= \mathrm{Val}^{\mathfrak{M}}(\overline{n+1}) \text{ by definition of } g \\
&= \mathrm{Val}^{\mathfrak{M}}(\overline{n}') \\
&= \prime^{\mathfrak{M}}(\mathrm{Val}^{\mathfrak{M}}(\overline{n})) \\
&= \prime^{\mathfrak{M}}(g(n)) \text{ by definition of } g \\
&= \prime^{\mathfrak{M}}(h(n)) \text{ by induction hypothesis} \\
&= h(\prime^{\mathfrak{N}}(n)) \text{ since } h \text{ is an isomorphism} \\
&= h(n+1)
\end{aligned}$$

□

For any denumerable set $X$, there's a bijection between $\mathbb{N}$ and $X$, so every such set $X$ is potentially the domain of a standard model. In fact, once you pick an object $z \in X$ and a suitable function $s\colon X \to X$ as $\mathrm{o}^{\mathfrak{X}}$ and $\prime^{\mathfrak{X}}$, the interpretation of $+$, $\times$, and $<$ is already fixed. Only functions $s = \prime^{\mathfrak{X}}$ that are both injective and surjective are suitable in a standard model. It has to be injective since the successor function in $\mathfrak{N}$ is, and that $\prime$ is injective is expressed by a sentence true in $\mathfrak{N}$ which $\mathfrak{X}$ thus also has to make true. It has to be surjective because otherwise there would be some $x \in X$ not in the domain of $s$, i.e., the sentence $\forall x\, \exists y\, y' = x$ would be false—but it is true in $\mathfrak{N}$.

## 22.3 Non-Standard Models

We call a structure for $\mathcal{L}_A$ standard if it is isomorphic to $\mathfrak{N}$. If a structure isn't isomorphic to $\mathfrak{N}$, it is called non-standard.

**Definition 22.5.** A structure $\mathfrak{M}$ for $\mathcal{L}_A$ is *non-standard* if it is not isomorphic to $\mathfrak{N}$. The elements $x \in |\mathfrak{M}|$ which are equal to $\mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for some $n \in \mathbb{N}$ are called *standard numbers* (of $\mathfrak{M}$), and those not, *non-standard numbers*.

By Proposition 22.2, any standard structure for $\mathcal{L}_A$ contains only standard elements. Consequently, a non-standard structure must contain at least one non-standard element. In fact, the existence of a non-standard element guarantees that the structure is non-standard.

**Proposition 22.6.** *If a structure $\mathfrak{M}$ for $\mathcal{L}_A$ contains a non-standard number, $\mathfrak{M}$ is non-standard.*

*Proof.* Suppose not, i.e., suppose $\mathfrak{M}$ standard but contains a non-standard number $x$. Let $g\colon \mathbb{N} \to |\mathfrak{M}|$ be an isomorphism. It is easy to see (by induction on $n$) that $g(\mathrm{Val}^{\mathfrak{N}}(\overline{n})) = \mathrm{Val}^{\mathfrak{M}}(\overline{n})$. In other words, $g$ maps standard numbers of $\mathfrak{N}$ to standard numbers of $\mathfrak{M}$. If $\mathfrak{M}$ contains a non-standard number, $g$ cannot be surjective, contrary to hypothesis. □

It is easy enough to specify non-standard structures for $\mathcal{L}_A$. For instance, take the structure with domain $\mathbb{Z}$ and interpret all non-logical symbols as usual. Since negative numbers are not values of $\overline{n}$ for any $n$, this structure is non-standard. Of course, it will not be a *model* of arithmetic in the sense that it makes the same sentences true as $\mathfrak{N}$. For instance, $\forall x\, x' \neq 0$ is false. However, we can prove that non-standard models of arithmetic exist easily enough, using the compactness theorem.

**Proposition 22.7.** *Let* $\mathbf{TA} = \{\varphi : \mathfrak{N} \vDash \varphi\}$ *be the theory of* $\mathbf{N}$. $\mathbf{TA}$ *has an enumerable non-standard model.*

*Proof.* Expand $\mathcal{L}_A$ by a new constant symbol $c$ and consider the set of sentences

$$\Gamma = \mathbf{TA} \cup \{c \neq \overline{0}, c \neq \overline{1}, c \neq \overline{2}, \dots\}$$

Any model $\mathfrak{M}^c$ of $\Gamma$ would contain an element $x = c^{\mathfrak{M}}$ which is non-standard, since $x \neq \mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for all $n \in \mathbb{N}$. Also, obviously, $\mathfrak{M}^c \vDash \mathbf{TA}$, since $\mathbf{TA} \subseteq \Gamma$. If we turn $\mathfrak{M}^c$ into a structure $\mathfrak{M}$ for $\mathcal{L}_A$ simply by forgetting about $c$, its domain still contains the non-standard $x$, and also $\mathfrak{M} \vDash \mathbf{TA}$. The latter is guaranteed since $c$ does not occur in $\mathbf{TA}$. So, it suffices to show that $\Gamma$ has a model.

We use the compactness theorem to show that $\Gamma$ has a model. If every finite subset of $\Gamma$ is satisfiable, so is $\Gamma$. Consider any finite subset $\Gamma_0 \subseteq \Gamma$. $\Gamma_0$ includes some sentences of $\mathbf{TA}$ and some of the form $c \neq \overline{n}$, but only finitely many. Suppose $k$ is the largest number so that $c \neq \overline{k} \in \Gamma_0$. Define $\mathfrak{N}_k$ by expanding $\mathfrak{N}$ to include the interpretation $c^{\mathfrak{N}_k} = k + 1$. $\mathfrak{N}_k \vDash \Gamma_0$: if $\varphi \in \mathbf{TA}$, $\mathfrak{N}_k \vDash \varphi$ since $\mathfrak{N}_k$ is just like $\mathfrak{N}$ in all respects except $c$, and $c$ does not occur in $\varphi$. And $\mathfrak{N}_k \vDash c \neq \overline{n}$, since $n \leq k$, and $\mathrm{Val}^{\mathfrak{N}_k}(c) = k + 1$. Thus, every finite subset of $\Gamma$ is satisfiable. □

## 22.4 Models of Q

We know that there are non-standard structures that make the same sentences true as $\mathfrak{N}$ does, i.e., is a model of $\mathbf{TA}$. Since $\mathfrak{N} \vDash \mathbf{Q}$, any model of $\mathbf{TA}$ is also a model of $\mathbf{Q}$. $\mathbf{Q}$ is much weaker than $\mathbf{TA}$, e.g., $\mathbf{Q} \nvdash \forall x \forall y\, (x + y) = (y + x)$. Weaker theories are easier to satisfy: they have more models. E.g., $\mathbf{Q}$ has models which make $\forall x \forall y\, (x + y) = (y + x)$ false, but those cannot also be models of $\mathbf{TA}$, or $\mathbf{PA}$ for that matter. Models of $\mathbf{Q}$ are also relatively simple: we can specify them explicitly.

**Example 22.8.** Consider the structure $\mathfrak{K}$ with domain $|\mathfrak{K}| = \mathbb{N} \cup \{a\}$ and interpretations

$$\begin{aligned}
\mathsf{o}^{\mathfrak{K}} &= 0 \\
\prime^{\mathfrak{K}}(x) &= \begin{cases} x+1 & \text{if } x \in \mathbb{N} \\ a & \text{if } x = a \end{cases} \\
+^{\mathfrak{K}}(x,y) &= \begin{cases} x+y & \text{if } x, y \in \mathbb{N} \\ a & \text{otherwise} \end{cases} \\
\times^{\mathfrak{K}}(x,y) &= \begin{cases} xy & \text{if } x, y \in \mathbb{N} \\ a & \text{otherwise} \end{cases} \\
<^{\mathfrak{K}} &= \{\langle x, y\rangle : x, y \in \mathbb{N} \text{ and } x < y\} \cup \{\langle x, a\rangle : x \in |\mathfrak{K}|\}
\end{aligned}$$

To show that $\mathfrak{K} \vDash \mathbf{Q}$ we have to verify that all axioms of **Q** are true in $\mathfrak{K}$. For convenience, let's write $x^*$ for $\prime^{\mathfrak{K}}(x)$ (the "successor" of $x$ in $\mathfrak{K}$), $x \oplus y$ for $+^{\mathfrak{K}}(x,y)$ (the "sum" of $x$ and $y$ in $\mathfrak{K}$, $x \otimes y$ for $\times^{\mathfrak{K}}(x,y)$ (the "product" of $x$ and $y$ in $\mathfrak{K}$), and $x \lessdot y$ for $\langle x, y\rangle \in <^{\mathfrak{K}}$. With these abbreviations, we can give the operations in $\mathfrak{K}$ more perspicuously as

| $x$ | $x^*$ |
|---|---|
| $n$ | $n+1$ |
| $a$ | $a$ |

| $x \oplus y$ | $m$ | $a$ |
|---|---|---|
| $n$ | $n+m$ | $a$ |
| $a$ | $a$ | $a$ |

| $x \otimes y$ | $m$ | $a$ |
|---|---|---|
| $n$ | $nm$ | $a$ |
| $a$ | $a$ | $a$ |

We have $n \lessdot m$ iff $n < m$ for $n, m \in \mathbb{N}$ and $x \lessdot a$ for all $x \in |\mathfrak{K}|$.

$\mathfrak{K} \vDash \forall x \forall y\, (x' = y' \to x = y)$ since $*$ is injective. $\mathfrak{K} \vDash \forall x\, \mathsf{o} \neq x'$ since 0 is not a $*$-successor in $\mathfrak{K}$. $\mathfrak{N} \vDash \forall x\, (x \neq \mathsf{o} \to \exists y\, x = y')$ since for every $n > 0$, $n = (n-1)^*$, and $a = a^*$.

$\mathfrak{K} \vDash \forall x\, (x + \mathsf{o}) = x$ since $n \oplus 0 = n + 0 = n$, and $a \oplus 0 = a$ by definition of $\oplus$. $\mathfrak{K} \vDash \forall x \forall y\, (x + y') = (x + y)'$ is a bit trickier. If $n$, $m$ are both standard, we have:

$$(n \oplus m^*) = (n + (m+1)) = (n+m) + 1 = (n \oplus m)^*$$

since $\oplus$ and $^*$ agree with $+$ and $\prime$ on standard numbers. Now suppose $x \in |\mathfrak{K}|$. Then

$$(x \oplus a^*) = (x \oplus a) = a = a^* = (x \oplus a)^*$$

The remaining case is if $y \in |\mathfrak{K}|$ but $x = a$. Here we also have to distinguish cases according to whether $y = n$ is standard or $y = b$:

$$\begin{aligned}
(a \oplus n^*) &= (a \oplus (n+1)) = a = a^* = (x \oplus n)^* \\
(a \oplus a^*) &= (a \oplus a) = a = a^* = (x \oplus a)^*
\end{aligned}$$

This is of course a bit more detailed than needed. For instance, since $a \oplus z = a$ whatever $z$ is, we can immediately conclude $a \oplus a^* = a$. The remaining axioms can be verified the same way.

$\mathfrak{K}$ is thus a model of **Q**. Its "addition" $\oplus$ is also commutative. But there are other sentences true in $\mathfrak{N}$ but false in $\mathfrak{K}$, and vice versa. For instance, $a \nleqslant a$, so $\mathfrak{K} \vDash \exists x\, x < x$ and $\mathfrak{K} \nvDash \forall x \neg x < x$. This shows that $\mathbf{Q} \nvdash \forall x \neg x < x$.

**Example 22.9.** Consider the structure $\mathfrak{L}$ with domain $|\mathfrak{L}| = \mathbb{N} \cup \{a, b\}$ and interpretations $'^{\mathfrak{L}} = *$, $+^{\mathfrak{L}} = \oplus$ given by

| $x$ | $x^*$ |
|---|---|
| $n$ | $n+1$ |
| $a$ | $a$ |
| $b$ | $b$ |

| $x \oplus y$ | $m$ | $a$ | $b$ |
|---|---|---|---|
| $n$ | $n+m$ | $b$ | $a$ |
| $a$ | $a$ | $b$ | $a$ |
| $b$ | $b$ | $b$ | $a$ |

Since $*$ is injective, 0 is not in its range, and every $x \in |\mathfrak{L}|$ other than 0 is, axioms $Q_1$–$Q_3$ are true in $\mathfrak{L}$. For any $x$, $x \oplus 0 = x$, so $Q_4$ is true as well. For $Q_5$, consider $x \oplus y^*$ and $(x \oplus y)^*$. They are equal if $x$ and $y$ are both standard, since then $*$ and $\oplus$ agree with $'$ and $+$. If $x$ is non-standard, and $y$ is standard, we have $x \oplus y^* = x = x^* = (x \oplus y)^*$. If $x$ and $y$ are both non-standard, we have four cases:

$$a \oplus a^* = b = b^* = (a \oplus a)^*$$
$$b \oplus b^* = a = a^* = (b \oplus b)^*$$
$$b \oplus a^* = b = b^* = (b \oplus y)^*$$
$$a \oplus b^* = a = a^* = (a \oplus b)^*$$

If $x$ is standard, but $y$ is non-standard, we have

$$n \oplus a^* = n \oplus a = b = b^* = (n \oplus a)^*$$
$$n \oplus b^* = n \oplus b = a = a^* = (n \oplus b)^*$$

So, $\mathfrak{L} \vDash Q_5$. However, $a \oplus 0 \neq 0 \oplus a$, so $\mathfrak{L} \nvDash \forall x \forall y\, (x + y) = (y + x)$.

We've explicitly constructed models of **Q** in which the non-standard elements live "beyond" the standard elements. In fact, that much is required by the axioms. A non-standard element $x$ cannot be $\leqslant 0$. Otherwise, for some $z$, $x \oplus z^* = 0$ by $Q8$. But then $0 = x \oplus z^* = (x \oplus z)^*$ by $Q_5$, contradicting $Q_2$. Also, for every $n$, $\mathbf{Q} \vdash \forall x\, (x < \overline{n}' \to (x = \overline{0} \vee x = \overline{1} \vee \cdots \vee x = \overline{n}))$, so we can't have $a \leqslant n$ for any $n > 0$.

## 22.5 Computable Models of Arithmetic

The standard model $\mathfrak{N}$ has two nice features. Its domain is the natural numbers $\mathbb{N}$, i.e., its elements are just the kinds of things we want to talk about

using the language of arithmetic, and the standard numeral $\overline{n}$ actually picks out $n$. The other nice feature is that the interpretations of the non-logical symbols of $\mathcal{L}_A$ are all *computable*. The successor, addition, and multiplication functions which serve as $\prime^{\mathfrak{N}}$, $+^{\mathfrak{N}}$, and $\times^{\mathfrak{N}}$ are computable functions of numbers. (Computable by Turing machines, or definable by primitive recursion, say.) And the less-than relation on $\mathfrak{N}$, i.e., $<^{\mathfrak{N}}$, is decidable.

Non-standard models of arithmetical theories such as **Q** and **PA** must contain non-standard elements. Thus their domains typically include elements in addition to $\mathbb{N}$. However, any countable structure can be built on any denumerable set, including $\mathbb{N}$. So there are also non-standard models with domain $\mathbb{N}$. In such models $\mathfrak{M}$, of course, at least some numbers cannot play the roles they usually play, since some $k$ must be different from $\mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for all $n \in \mathbb{N}$.

**Definition 22.10.** A structure $\mathfrak{M}$ for $\mathcal{L}_A$ is *computable* iff $|\mathfrak{M}| = \mathbb{N}$ and $\prime^{\mathfrak{M}}$, $+^{\mathfrak{M}}$, $\times^{\mathfrak{M}}$ are computable functions and $<^{\mathfrak{M}}$ is a decidable relation.

**Example 22.11.** Recall the structure $\mathfrak{K}$ from Example 22.8 Its domain was $|\mathfrak{K}| = \mathbb{N} \cup \{a\}$ and interpretations

$$
\begin{aligned}
\mathsf{o}^{\mathfrak{K}} &= 0 \\
\prime^{\mathfrak{K}}(x) &= \begin{cases} x+1 & \text{if } x \in \mathbb{N} \\ a & \text{if } x = a \end{cases} \\
+^{\mathfrak{K}}(x,y) &= \begin{cases} x+y & \text{if } x, y \in \mathbb{N} \\ a & \text{otherwise} \end{cases} \\
\times^{\mathfrak{K}}(x,y) &= \begin{cases} xy & \text{if } x, y \in \mathbb{N} \\ a & \text{otherwise} \end{cases} \\
<^{\mathfrak{K}} &= \{\langle x, y\rangle : x, y \in \mathbb{N} \text{ and } x < y\} \cup \{\langle x, a\rangle : n \in |\mathfrak{K}|\}
\end{aligned}
$$

But $|\mathfrak{K}|$ is denumerable and so is equinumerous with $\mathbb{N}$. For instance, $g\colon \mathbb{N} \to |\mathfrak{K}|$ with $g(0) = a$ and $g(n) = n + 1$ for $n > 0$ is a bijection. We can turn it into an isomorphism between a new model $\mathfrak{K}'$ of **Q** and $\mathfrak{K}$. In $\mathfrak{K}'$, we have to assign different functions and relations to the symbols of $\mathcal{L}_A$, since different elements of $\mathbb{N}$ play the roles of standard and non-standard numbers.

Specifically, 0 now plays the role of $a$, not of the smallest standard number. The smallest standard number is now 1. So we assign $\mathsf{o}^{\mathfrak{K}'} = 1$. The successor function is also different now: given a standard number, i.e., an $n > 0$, it still returns $n + 1$. But 0 now plays the role of $a$, which is its own successor. So

$\prime^{\mathfrak{K}'}(0) = 0$. For addition and multiplication we likewise have

$$+^{\mathfrak{K}'}(x,y) = \begin{cases} x+y & \text{if } x,y>0 \\ 0 & \text{otherwise} \end{cases}$$

$$\times^{\mathfrak{K}'}(x,y) = \begin{cases} xy & \text{if } x,y>0 \\ 0 & \text{otherwise} \end{cases}$$

And we have $\langle x,y\rangle \in <^{\mathfrak{K}'}$ iff $x<y$ and $x>0$ and $y>0$, or if $y=0$.

All of these functions are computable functions of natural numbers and $<^{\mathfrak{K}'}$ is a decidable relation on $\mathbb{N}$—but they are not the same functions as successor, addition, and multiplication on $\mathbb{N}$, and $<^{\mathfrak{K}'}$ is not the same relation as $<$ on $\mathbb{N}$.

This example shows that **Q** has computable non-standard models with domain $\mathbb{N}$. However, the following result shows that this is not true for models of **PA** (and thus also for models of **TA**).

**Theorem 22.12** (Tennenbaum's Theorem)**.** $\mathfrak{N}$ *is the only computable model of* **PA**.

## Problems

**Problem 22.1.** Show that the converse of Proposition 22.2 is false, i.e., give an example of a structure $\mathfrak{M}$ with $|\mathfrak{M}| = \{\mathrm{Val}^{\mathfrak{M}}(\overline{n}) : n \in \mathbb{N}\}$ that is not isomorphic to $\mathfrak{N}$.

**Problem 22.2.** Recall that **Q** contains the axioms

$$\forall x\, \forall y\, (x' = y' \to x = y) \tag{$Q_1$}$$

$$\forall x\, 0 \neq x' \tag{$Q_2$}$$

$$\forall x\, (x \neq 0 \to \exists y\, x = y') \tag{$Q_3$}$$

Give structures $\mathfrak{M}_1, \mathfrak{M}_2, \mathfrak{M}_3$ such that

1. $\mathfrak{M}_1 \vDash Q_1$, $\mathfrak{M}_1 \vDash Q_2$, $\mathfrak{M}_1 \nvDash Q_3$;
2. $\mathfrak{M}_2 \vDash Q_1$, $\mathfrak{M}_2 \nvDash Q_2$, $\mathfrak{M}_2 \vDash Q_3$; and
3. $\mathfrak{M}_3 \nvDash Q_1$, $\mathfrak{M}_3 \vDash Q_2$, $\mathfrak{M}_3 \vDash Q_3$;

Obviously, you just have to specify $0^{\mathfrak{M}_i}$ and $\prime^{\mathfrak{M}_i}$ for each.

**Problem 22.3.** Prove that $\mathfrak{K}$ from Example 22.8 satisifies the remaining axioms of **Q**,

$$\forall x\, (x \times 0) = 0 \tag{$Q_6$}$$

$$\forall x\, \forall y\, (x \times y') = ((x \times y) + x) \tag{$Q_7$}$$

$$\forall x\, \forall y\, (x < y \leftrightarrow \exists z\, (x + z' = y)) \tag{$Q_8$}$$

Find a sentence only involving $\prime$ true in $\mathfrak{N}$ but false in $\mathfrak{K}$.

**Problem 22.4.** Expand $\mathfrak{L}$ of Example 22.9 to include $\otimes$ and $\oslash$ that interpret $\times$ and $<$. Show that your structure satisifies the remaining axioms of **Q**,

$$\forall x\,(x \times \mathsf{o}) = \mathsf{o} \tag{$Q_6$}$$
$$\forall x\,\forall y\,(x \times y') = ((x \times y) + x) \tag{$Q_7$}$$
$$\forall x\,\forall y\,(x < y \leftrightarrow \exists z\,(x + z' = y)) \tag{$Q_8$}$$

**Problem 22.5.** In $\mathfrak{L}$ of Example 22.9, $a^* = a$ and $b^* = b$. Is there a model of **Q** in which $a^* = b$ and $b^* = a$?

**Problem 22.6.** Give a structure $\mathfrak{L}'$ with $|\mathfrak{L}'| = \mathbb{N}$ isomorphic to $\mathfrak{L}$ of Example 22.9.

# Chapter 23

# The Interpolation Theorem

## 23.1 Introduction

The interpolation theorem is the following result: Suppose $\vDash \varphi \to \psi$. Then there is a sentence $\chi$ such that $\vDash \varphi \to \chi$ and $\vDash \chi \to \psi$. Moreover, every constant symbol, function symbol, and predicate symbol (other than $=$) in $\chi$ occurs both in $\varphi$ and $\psi$. The sentence $\chi$ is called an *interpolant* of $\varphi$ and $\psi$.

The interpolation theorem is interesting in its own right, but its main importance lies in the fact that it can be used to prove results about definability in a theory, and the conditions under which combining two consistent theories results in a consistent theory. The first result is known as the Beth definability theorem; the second, Robinson's joint consistency theorem.

## 23.2 Separation of Sentences

A bit of groundwork is needed before we can proceed with the proof of the interpolation theorem. An interpolant for $\varphi$ and $\psi$ is a sentence $\chi$ such that $\varphi \vDash \chi$ and $\chi \vDash \psi$. By contraposition, the latter is true iff $\neg\psi \vDash \neg\chi$. A sentence $\chi$ with this property is said to *separate* $\varphi$ and $\neg\psi$. So finding an interpolant for $\varphi$ and $\psi$ amounts to finding a sentence that separates $\varphi$ and $\neg\psi$. As so often, it will be useful to consider a generalization: a sentence that separates two *sets* of sentences.

**Definition 23.1.** A sentence $\chi$ *separates* sets of sentences $\Gamma$ and $\Delta$ if and only if $\Gamma \vDash \chi$ and $\Delta \vDash \neg\chi$. If no such sentence exists, then $\Gamma$ and $\Delta$ are *inseparable*.

The inclusion relations between the classes of models of $\Gamma$, $\Delta$ and $\chi$ are represented below:

**Lemma 23.2.** *Suppose $\mathcal{L}_0$ is the language containing every constant symbol, function symbol and predicate symbol (other than $\doteq$) that occurs in* both *$\Gamma$ and $\Delta$, and let*

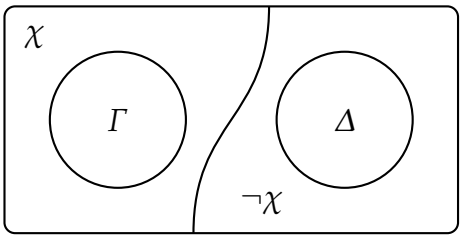



Figure 23.1: $\chi$ separates $\Gamma$ and $\Delta$

*$\mathcal{L}_0'$ be obtained by the addition of infinitely many new constant symbols $c_n$ for $n \geq 0$. Then if $\Gamma$ and $\Delta$ are inseparable in $\mathcal{L}_0$, they are also inseparable in $\mathcal{L}_0'$.*

*Proof.* We proceed indirectly: suppose by way of contradiction that $\Gamma$ and $\Delta$ are separated in $\mathcal{L}_0'$. Then $\Gamma \vDash \chi[c/x]$ and $\Delta \vDash \neg\chi[c/x]$ for some $\chi \in \mathcal{L}_0$ (where $c$ is a new constant symbol—the case where $\chi$ contains more than one such new constant symbol is similar). By compactness, there are *finite* subsets $\Gamma_0$ of $\Gamma$ and $\Delta_0$ of $\Delta$ such that $\Gamma_0 \vDash \chi[c/x]$ and $\Delta_0 \vDash \neg\chi[c/x]$. Let $\gamma$ be the conjunction of all formulas in $\Gamma_0$ and $\delta$ the conjunction of all formulas in $\Delta_0$. Then

$$\gamma \vDash \chi[c/x], \qquad \delta \vDash \neg\chi[c/x].$$

From the former, by Generalization, we have $\gamma \vDash \forall x\, \chi$, and from the latter by contraposition, $\chi[c/x] \vDash \neg\delta$, whence also $\forall x\, \chi \vDash \neg\delta$. Contraposition again gives $\delta \vDash \neg\forall x\, \chi$. By monotony,

$$\Gamma \vDash \forall x\, \chi, \qquad \Delta \vDash \neg\forall x\, \chi,$$

so that $\forall x\, \chi$ separates $\Gamma$ and $\Delta$ in $\mathcal{L}_0$. □

**Lemma 23.3.** *Suppose that $\Gamma \cup \{\exists x\, \sigma\}$ and $\Delta$ are inseparable, and $c$ is a new constant symbol not in $\Gamma$, $\Delta$, or $\sigma$. Then $\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and $\Delta$ are also inseparable.*

*Proof.* Suppose for contradiction that $\chi$ separates $\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and $\Delta$, while at the same time $\Gamma \cup \{\exists x \sigma\}$ and $\Delta$ are inseparable. We distinguish two cases:

1. $c$ does not occur in $\chi$: in this case $\Gamma \cup \{\exists x\, \sigma, \neg\chi\}$ is satisfiable (otherwise $\chi$ separates $\Gamma \cup \{\exists x\, \sigma\}$ and $\Delta$). It remains so if $\sigma[c/x]$ is added, so $\chi$ does not separate $\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and $\Delta$ after all.

2. $c$ does occur in $\chi$ so that $\chi$ has the form $\chi[c/x]$. Then we have that

$$\Gamma \cup \{\exists x\, \sigma, \sigma[c/x]\} \vDash \chi[c/x],$$

whence $\Gamma, \exists x\, \sigma \vDash \forall x\, (\sigma \to \chi)$ by the Deduction Theorem and Generalization, and finally $\Gamma \cup \{\exists x\, \sigma\} \vDash \exists x\, \chi$. On the other hand, $\Delta \vDash \neg\chi[c/x]$ and hence by Generalization $\Delta \vDash \neg\exists x\, \chi$. So $\Gamma \cup \{\exists x\, \sigma\}$ and $\Delta$ are separable, a contradiction. □

## 23.3 Craig's Interpolation Theorem

**Theorem 23.4** (Craig's Interpolation Theorem)**.** *If $\vDash \varphi \to \psi$, then there is a sentence $\chi$ such that $\vDash \varphi \to \chi$ and $\vDash \chi \to \psi$, and every constant symbol, function symbol, and predicate symbol (other than $=$) in $\chi$ occurs both in $\varphi$ and $\psi$. The sentence $\chi$ is called an* interpolant *of $\varphi$ and $\psi$.*

*Proof.* Suppose $\mathcal{L}_1$ is the language of $\varphi$ and $\mathcal{L}_2$ is the language of $\psi$. Let $\mathcal{L}_0 = \mathcal{L}_1 \cap \mathcal{L}_2$. For each $i \in \{0, 1, 2\}$, let $\mathcal{L}'_i$ be obtained from $\mathcal{L}_i$ by adding the infinitely many new constant symbols $c_0, c_1, c_2, \ldots$.

If $\varphi$ is unsatisfiable, $\exists x\, x \neq x$ is an interpolant. If $\neg\psi$ is unsatisfiable (and hence $\psi$ is valid), $\exists x\, x = x$ is an interpolant. So we may assume also that both $\varphi$ and $\neg\psi$ are satisfiable.

In order to prove the contrapositive of the Interpolation Theorem, assume that there is no interpolant for $\varphi$ and $\psi$. In other words, assume that $\{\varphi\}$ and $\{\neg\psi\}$ are inseparable in $\mathcal{L}_0$.

Our goal is to extend the pair $(\{\varphi\}, \{\neg\psi\})$ to a maximally inseparable pair $(\Gamma^*, \Delta^*)$. Let $\varphi_0$, $\varphi_1$, $\varphi_2$, ...enumerate the sentences of $\mathcal{L}_1$, and $\psi_0$, $\psi_1$, $\psi_2$, ...enumerate the sentences of $\mathcal{L}_2$. We define two increasing sequences of sets of sentences $(\Gamma_n, \Delta_n)$, for $n \geq 0$, as follows. Put $\Gamma_0 = \{\varphi\}$ and $\Delta_0 = \{\neg\psi\}$. Assuming $(\Gamma_n, \Delta_n)$ are already defined, define $\Gamma_{n+1}$ and $\Delta_{n+1}$ by:

1. If $\Gamma_n \cup \{\varphi_n\}$ and $\Delta_n$ are inseparable in $\mathcal{L}'_0$, put $\varphi_n$ in $\Gamma_{n+1}$. Moreover, if $\varphi_n$ is an existential formula $\exists x\, \sigma$ then pick a new constant symbol $c$ not occurring in $\Gamma_n$, $\Delta_n$, $\varphi_n$ or $\psi_n$, and put $\sigma[c/x]$ in $\Gamma_{n+1}$.

2. If $\Gamma_{n+1}$ and $\Delta_n \cup \{\psi_n\}$ are inseparable in $\mathcal{L}'_0$, put $\psi_n$ in $\Delta_{n+1}$. Moreover, if $\psi_n$ is an existential formula $\exists x\, \sigma$, then pick a new constant symbol $c$ not occurring in $\Gamma_{n+1}$, $\Delta_n$, $\varphi_n$ or $\psi_n$, and put $\sigma[c/x]$ in $\Delta_{n+1}$.

Finally, define:

$$\Gamma^* = \bigcup_{n \geq 0} \Gamma_n, \qquad \Delta^* = \bigcup_{n \geq 0} \Delta_n.$$

By simultaneous induction on $n$ we can now prove:

1. $\Gamma_n$ and $\Delta_n$ are inseparable in $\mathcal{L}'_0$;

2. $\Gamma_{n+1}$ and $\Delta_n$ are inseparable in $\mathcal{L}'_0$.

The basis for (1) is given by Lemma 23.2. For part (2), we need to distinguish three cases:

1. If $\Gamma_0 \cup \{\varphi_0\}$ and $\Delta_0$ are separable, then $\Gamma_1 = \Gamma_0$ and (2) is just (1);
2. If $\Gamma_1 = \Gamma_0 \cup \{\varphi_0\}$, then $\Gamma_1$ and $\Delta_0$ are inseparable by construction.
3. It remains to consider the case where $\varphi_0$ is existential, so that $\Gamma_1 = \Gamma_0 \cup \{\exists x\, \sigma, \sigma[c/x]\}$. By construction, $\Gamma_0 \cup \{\exists x\, \sigma\}$ and $\Delta_0$ are inseparable, so that by Lemma 23.3 also $\Gamma_0 \cup \{\exists x\, \sigma, \sigma[c/x]\}$ and $\Delta_0$ are inseparable.

This completes the basis of the induction for (1) and (2) above. Now for the inductive step. For (1), if $\Delta_{n+1} = \Delta_n \cup \{\psi_n\}$ then $\Gamma_{n+1}$ and $\Delta_{n+1}$ are inseparable by construction (even when $\psi_n$ is existential, by Lemma 23.3); if $\Delta_{n+1} = \Delta_n$ (because $\Gamma_{n+1}$ and $\Delta_n \cup \{\psi_n\}$ are separable), then we use the induction hypothesis on (2). For the inductive step for (2), if $\Gamma_{n+2} = \Gamma_{n+1} \cup \{\varphi_{n+1}\}$ then $\Gamma_{n+2}$ and $\Delta_{n+1}$ are inseparable by construction (even when $\varphi_{n+1}$ is existential, by Lemma 23.3); and if $\Gamma_{n+2} = \Gamma_{n+1}$ then we use the inductive case for (1) just proved. This concludes the induction on (1) and (2).

It follows that $\Gamma^*$ and $\Delta^*$ are inseparable; if not, by compactness, there is $n \geq 0$ that separates $\Gamma_n$ and $\Delta_n$, against (1). In particular, $\Gamma^*$ and $\Delta^*$ are consistent: for if the former or the latter is inconsistent, then they are separated by $\exists x\, x \neq x$ or $\forall x\, x = x$, respectively.

We now show that $\Gamma^*$ is maximally consistent in $\mathcal{L}'_1$ and likewise $\Delta^*$ in $\mathcal{L}'_2$. For the former, suppose that $\varphi_n \notin \Gamma^*$ and $\neg\varphi_n \notin \Gamma^*$, for some $n \geq 0$. If $\varphi_n \notin \Gamma^*$ then $\Gamma_n \cup \{\varphi_n\}$ is separable from $\Delta_n$, and so there is $\chi \in \mathcal{L}'_0$ such that both:

$$\Gamma^* \vDash \varphi_n \to \chi, \qquad \Delta^* \vDash \neg\chi.$$

Likewise, if $\neg\varphi_n \notin \Gamma^*$, there is $\chi' \in \mathcal{L}'_0$ such that both:

$$\Gamma^* \vDash \neg\varphi_n \to \chi', \qquad \Delta^* \vDash \neg\chi'.$$

By propositional logic, $\Gamma^* \vDash \chi \vee \chi'$ and $\Delta^* \vDash \neg(\chi \vee \chi')$, so $\chi \vee \chi'$ separates $\Gamma^*$and $\Delta^*$. A similar argument establishes that $\Delta^*$ is maximal.

Finally, we show that $\Gamma^* \cap \Delta^*$ is maximally consistent in $\mathcal{L}'_0$. It is obviously consistent, since it is the intersection of consistent sets. To show maximality, let $\sigma \in \mathcal{L}'_0$. Now, $\Gamma^*$ is maximal in $\mathcal{L}'_1 \supseteq \mathcal{L}'_0$, and similarly $\Delta^*$ is maximal in $\mathcal{L}'_2 \supseteq \mathcal{L}'_0$. It follows that either $\sigma \in \Gamma^*$ or $\neg\sigma \in \Gamma^*$, and either $\sigma \in \Delta^*$ or $\neg\sigma \in \Delta^*$. If $\sigma \in \Gamma^*$ and $\neg\sigma \in \Delta^*$ then $\sigma$ would separate $\Gamma^*$ and $\Delta^*$; and if $\neg\sigma \in \Gamma^*$ and $\sigma \in \Delta^*$ then $\Gamma^*$ and $\Delta^*$ would be separated by $\neg\sigma$. Hence, either $\sigma \in \Gamma^* \cap \Delta^*$ or $\neg\sigma \in \Gamma^* \cap \Delta^*$, and $\Gamma^* \cap \Delta^*$ is maximal.

Since $\Gamma^*$ is maximally consistent, it has a model $\mathfrak{M}'_1$ whose domain $|\mathfrak{M}'_1|$ comprises all and only the elements $c^{\mathfrak{M}'_1}$ interpreting the constant symbols—just like in the proof of the completeness theorem (Theorem 19.18). Similarly,

$\Delta^*$ has a model $\mathfrak{M}'_2$ whose domain $|\mathfrak{M}'_2|$ is given by the interpretations $c^{\mathfrak{M}'_2}$ of the constant symbols.

Let $\mathfrak{M}_1$ be obtained from $\mathfrak{M}'_1$ by dropping interpretations for constant symbols, function symbols, and predicate symbols in $\mathcal{L}'_1 \setminus \mathcal{L}'_0$, and similarly for $\mathfrak{M}_2$. Then the map $h\colon M_1 \to M_2$ defined by $h(c^{\mathfrak{M}'_1}) = c^{\mathfrak{M}'_2}$ is an isomorphism in $\mathcal{L}'_0$, because $\Gamma^* \cap \Delta^*$ is maximally consistent in $\mathcal{L}'_0$, as shown. This follows because any $\mathcal{L}'_0$-sentence either belongs to both $\Gamma^*$ and $\Delta^*$, or to neither: so $c^{\mathfrak{M}'_1} \in P^{\mathfrak{M}'_1}$ if and only if $P(c) \in \Gamma^*$ if and only if $P(c) \in \Delta^*$ if and only if $c^{\mathfrak{M}'_2} \in P^{\mathfrak{M}'_2}$. The other conditions satisfied by isomorphisms can be established similarly.

Let us now define a model $\mathfrak{M}$ for the language $\mathcal{L}_1 \cup \mathcal{L}_2$ as follows:

1. The domain $|\mathfrak{M}|$ is just $|\mathfrak{M}_2|$, i.e., the set of all elements $c^{\mathfrak{M}'_2}$;
2. If a predicate symbol $P$ is in $\mathcal{L}_2 \setminus \mathcal{L}_1$ then $P^{\mathfrak{M}} = P^{\mathfrak{M}'_2}$;
3. If a predicate $P$ is in $\mathcal{L}_1 \setminus \mathcal{L}_2$ then $P^{\mathfrak{M}} = h(P^{\mathfrak{M}'_2})$, i.e., $\langle c_1^{\mathfrak{M}'_2}, \ldots, c_n^{\mathfrak{M}'_2}\rangle \in P^{\mathfrak{M}}$ if and only if $\langle c_1^{\mathfrak{M}'_1}, \ldots, c_n^{\mathfrak{M}'_1}\rangle \in P^{\mathfrak{M}'_1}$.
4. If a predicate symbol $P$ is in $\mathcal{L}_0$ then $P^{\mathfrak{M}} = P^{\mathfrak{M}'_2} = h(P^{\mathfrak{M}'_1})$.
5. Function symbols of $\mathcal{L}_1 \cup \mathcal{L}_2$, including constant symbols, are handled similarly.

Finally, one shows by induction on formulas that $\mathfrak{M}$ agrees with $\mathfrak{M}'_1$ on all formulas of $\mathcal{L}'_1$ and with $\mathfrak{M}'_2$ on all formulas of $\mathcal{L}'_2$. In particular, $\mathfrak{M} \vDash \Gamma^* \cup \Delta^*$, whence $\mathfrak{M} \vDash \varphi$ and $\mathfrak{M} \vDash \neg\psi$, and $\nvdash \varphi \to \psi$. This concludes the proof of Craig's Interpolation Theorem. □

## 23.4 The Definability Theorem

One important application of the interpolation theorem is Beth's definability theorem. To define an $n$-place relation $R$ we can give a formula $\chi$ with $n$ free variables which does not involve $R$. This would be an *explicit* definition of $R$ in terms of $\chi$. We can then say also that a theory $\Sigma(P)$ in a language containing the $n$-place predicate symbol $P$ explicitly defines $P$ if it contains (or at least entails) a formalized explicit definition, i.e.,

$$\Sigma(P) \vDash \forall x_1 \ldots \forall x_n\, (P(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n)).$$

But an explicit definition is only one way of defining—in the sense of determining completely—a relation. A theory may also be such that the interpretation of $P$ is fixed by the interpretation of the rest of the language in any model. The definability theorem states that whenever a theory fixes the interpretation of $P$ in this way—whenever it *implicitly defines* $P$—then it also explicitly defines it.

**Definition 23.5.** Suppose $\mathcal{L}$ is a language not containing the predicate symbol $P$. A set $\Sigma(P)$ of sentences of $\mathcal{L} \cup \{P\}$ *explicitly defines* $P$ if and only if there is a formula $\chi(x_1, \ldots, x_n)$ of $\mathcal{L}$ such that

$$\Sigma(P) \vDash \forall x_1 \ldots \forall x_n \, (P(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n)).$$

**Definition 23.6.** Suppose $\mathcal{L}$ is a language not containing the predicate symbols $P$ and $P'$. A set $\Sigma(P)$ of sentences of $\mathcal{L} \cup \{P\}$ *implicitly defines* $P$ if and only if

$$\Sigma(P) \cup \Sigma(P') \vDash \forall x_1 \ldots \forall x_n \, (P(x_1, \ldots, x_n) \leftrightarrow P'(x_1, \ldots, x_n)),$$

where $\Sigma(P')$ is the result of uniformly replacing $P$ with $P'$ in $\Sigma(P)$.

In other words, for any model $\mathfrak{M}$ and $R, R' \subseteq |\mathfrak{M}|^n$, if both $(\mathfrak{M}, R) \vDash \Sigma(P)$ and $(\mathfrak{M}, R') \vDash \Sigma(P')$, then $R = R'$; where $(\mathfrak{M}, R)$ is the structure $\mathfrak{M}'$ for the expansion of $\mathcal{L}$ to $\mathcal{L} \cup \{P\}$ such that $P^{\mathfrak{M}'} = R$, and similarly for $(\mathfrak{M}, R')$.

**Theorem 23.7** (Beth Definability Theorem)**.** *A set $\Sigma(P)$ of $\mathcal{L} \cup \{P\}$-formulas implicitly defines $P$ if and only $\Sigma(P)$ explicitly defines $P$.*

*Proof.* If $\Sigma(P)$ explicitly defines $P$ then both

$$\begin{aligned} \Sigma(P) \vDash \quad & \forall x_1 \ldots \forall x_n \, (P(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n)) \\ \Sigma(P') \vDash \quad & \forall x_1 \ldots \forall x_n \, (P'(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n)) \end{aligned}$$

and the conclusion follows. For the converse: assume that $\Sigma(P)$ implicitly defines $P$. First, we add constant symbols $c_1, \ldots, c_n$ to $\mathcal{L}$. Then

$$\Sigma(P) \cup \Sigma(P') \vDash P(c_1, \ldots, c_n) \to P'(c_1, \ldots, c_n).$$

By compactness, there are finite sets $\Delta_0 \subseteq \Sigma(P)$ and $\Delta_1 \subseteq \Sigma(P')$ such that

$$\Delta_0 \cup \Delta_1 \vDash P(c_1, \ldots, c_n) \to P'(c_1, \ldots, c_n).$$

Let $\theta(P)$ be the conjunction of all sentences $\varphi(P)$ such that either $\varphi(P) \in \Delta_0$ or $\varphi(P') \in \Delta_1$ and let $\theta(P')$ be the conjunction of all sentences $\varphi(P')$ such that either $\varphi(P) \in \Delta_0$ or $\varphi(P') \in \Delta_1$. Then $\theta(P) \wedge \theta(P') \vDash P(c_1, \ldots, c_n) \to P'c_1 \ldots c_n$. We can re-arrange this so that each predicate symbol occurs on one side of $\vDash$:

$$\theta(P) \wedge P(c_1, \ldots, c_n) \vDash \theta(P') \to P'(c_1, \ldots, c_n).$$

By Craig's Interpolation Theorem there is a sentence $\chi(c_1, \ldots, c_n)$ not containing $P$ or $P'$ such that:

$$\theta(P) \wedge P(c_1, \ldots, c_n) \vDash \chi(c_1, \ldots, c_n); \quad \chi(c_1, \ldots, c_n) \vDash \theta(P') \to P'(c_1, \ldots, c_n).$$

From the former of these two entailments we have: $\theta(P) \vDash P(c_1, \ldots, c_n) \to \chi(c_1, \ldots, c_n)$. And from the latter, since an $\mathcal{L} \cup \{P\}$-model $(\mathfrak{M}, R) \vDash \varphi(P)$

if and only if the corresponding $\mathcal{L} \cup \{P'\}$-model $(\mathfrak{M}, R) \models \varphi(P')$, we have $\chi(c_1, \ldots, c_n) \vDash \theta(P) \to P(c_1, \ldots, c_n)$, from which:

$$\theta(P) \vDash \chi(c_1, \ldots, c_n) \to P(c_1, \ldots, c_n).$$

Putting the two together, $\theta(P) \vDash P(c_1, \ldots, c_n) \leftrightarrow \chi(c_1, \ldots, c_n)$, and by monotony and generalization also

$$\Sigma(P) \vDash \forall x_1 \ldots \forall x_n \, (P(x_1, \ldots, x_n) \leftrightarrow \chi(x_1, \ldots, x_n)). \qquad \square$$

# Chapter 24

# Lindström's Theorem

## 24.1 Introduction

In this chapter we aim to prove Lindström's characterization of first-order logic as the maximal logic for which (given certain further constraints) the Compactness and the Downward Löwenheim-Skolem theorems hold (Theorem 19.21 and Theorem 19.30). First, we need a more general characterization of the general class of logics to which the theorem applies. We will restrict ourselves to *relational* languages, i.e., languages which only contain predicate symbols and individual constants, but no function symbols.

## 24.2 Abstract Logics

**Definition 24.1.** An *abstract logic* is a pair $\langle L, \models_L \rangle$, where $L$ is a function that assigns to each language $\mathcal{L}$ a set $L(\mathcal{L})$ of sentences, and $\models_L$ is a relation between structures for the language $\mathcal{L}$ and elements of $L(\mathcal{L})$. In particular, $\langle F, \models \rangle$ is ordinary first-order logic, i.e., $F$ is the function assigning to the language $\mathcal{L}$ the set of first-order sentences built from the constants in $\mathcal{L}$, and $\models$ is the satisfaction relation of first-order logic.

Notice that we are still employing the same notion of structure for a given language as for first-order logic, but we do not presuppose that sentences are build up from the basic symbols in $\mathcal{L}$ in the usual way, nor that the relation $\models_L$ is recursively defined in the same way as for first-order logic. So for instance the definition, being completely general, is intended to capture the case where sentences in $\langle L, \models_L \rangle$ contain infinitely long conjunctions or disjunction, or quantifiers other than $\exists$ and $\forall$ (e.g., "there are infinitely many $x$ such that ..."), or perhaps infinitely long quantifier prefixes. To emphasize that "sentences" in $L(\mathcal{L})$ need not be ordinary sentences of first-order logic, in this chapter we use variables $\alpha$, $\beta$, ... to range over them, and reserve $\varphi$, $\psi$, ... for ordinary first-order formulas.

**Definition 24.2.** Let $\mathrm{Mod}_L(\alpha)$ denote the class $\{\mathfrak{M} : \mathfrak{M} \models_L \alpha\}$. If the language needs to be made explicit, we write $\mathrm{Mod}_L^{\mathcal{L}}(\alpha)$. Two structures $\mathfrak{M}$ and $\mathfrak{N}$ for $\mathcal{L}$ are *elementarily equivalent in* $\langle L, \models_L \rangle$, written $\mathfrak{M} \equiv_L \mathfrak{N}$, if the same sentences from $L(\mathcal{L})$ are true in each.

**Definition 24.3.** An abstract logic $\langle L, \models_L \rangle$ for the language $\mathcal{L}$ is *normal* if it satisfies the following properties:

1. (*L-Monotony*) For languages $\mathcal{L}$ and $\mathcal{L}'$, if $\mathcal{L} \subseteq \mathcal{L}'$, then $L(\mathcal{L}) \subseteq L(\mathcal{L}')$.

2. (*Expansion Property*) For each $\alpha \in L(\mathcal{L})$ there is a *finite* subset $\mathcal{L}'$ of $\mathcal{L}$ such that the relation $\mathfrak{M} \models_L \alpha$ depends only on the reduct of $\mathfrak{M}$ to $\mathcal{L}'$; i.e., if $\mathfrak{M}$ and $\mathfrak{N}$ have the same reduct to $\mathcal{L}'$ then $\mathfrak{M} \models_L \alpha$ if and only if $\mathfrak{N} \models_L \alpha$.

3. (*Isomorphism Property*) If $\mathfrak{M} \models_L \alpha$ and $\mathfrak{M} \simeq \mathfrak{N}$ then also $\mathfrak{N} \models_L \alpha$.

4. (*Renaming Property*) The relation $\models_L$ is preserved under renaming: if the language $\mathcal{L}'$ is obtained from $\mathcal{L}$ by replacing each symbol $P$ by a symbol $P'$ of the same arity and each constant $c$ by a distinct constant $c'$, then for each structure $\mathfrak{M}$ and sentence $\alpha$, $\mathfrak{M} \models_L \alpha$ if and only if $\mathfrak{M}' \models_L \alpha'$, where $\mathfrak{M}'$ is the $\mathcal{L}'$-structure corresponding to $\mathcal{L}$ and $\alpha' \in L(\mathcal{L}')$.

5. (*Boolean Property*) The abstract logic $\langle L, \models_L \rangle$ is closed under the Boolean connectives in the sense that for each $\alpha \in L(\mathcal{L})$ there is a $\beta \in L(\mathcal{L})$ such that $\mathfrak{M} \models_L \beta$ if and only if $\mathfrak{M} \not\models_L \alpha$, and for each $\alpha$ and $\beta$ there is a $\gamma$ such that $\mathrm{Mod}_L(\gamma) = \mathrm{Mod}_L(\alpha) \cap \mathrm{Mod}_L(\beta)$. Similarly for atomic formulas and the other connectives.

6. (*Quantifier Property*) For each constant $c$ in $\mathcal{L}$ and $\alpha \in L(\mathcal{L})$ there is a $\beta \in L(\mathcal{L})$ such that

$$\mathrm{Mod}_L^{\mathcal{L}'}(\beta) = \{\mathfrak{M} : (\mathfrak{M}, a)\} \in \mathrm{Mod}_L^{\mathcal{L}}(\alpha) \text{ for some } a \in |\mathfrak{M}|\},$$

   where $\mathcal{L}' = \mathcal{L} \setminus \{c\}$ and $(\mathfrak{M}, a)$ is the expansion of $\mathfrak{M}$ to $\mathcal{L}$ assigning $a$ to $c$.

7. (*Relativization Property*) Given a sentence $\alpha \in L(\mathcal{L})$ and symbols $R$, $c_1$, $\ldots$, $c_n$ not in $\mathcal{L}$, there is a sentence $\beta \in L(\mathcal{L} \cup \{R, c_1, \ldots, c_n\})$ called the *relativization* of $\alpha$ to $R(x, c_1, \ldots c_n)$, such that for each structure $\mathfrak{M}$:

$$(\mathfrak{M}, X, b_1, \ldots, b_n) \models_L \beta) \text{ if and only if } \mathfrak{N} \models_L \alpha,$$

   where $\mathfrak{N}$ is the substructure of $\mathfrak{M}$ with domain $|\mathfrak{N}| = \{a \in |\mathfrak{M}| : R^{\mathfrak{M}}(a, b_1, \ldots, b_n)\}$ (see Remark 1), and $(\mathfrak{M}, X, b_1, \ldots, b_n)$ is the expansion of $\mathfrak{M}$ interpreting $R$, $c_1$, $\ldots$, $c_n$ by $X$, $b_1$, $\ldots$, $b_n$, respectively (with $X \subseteq M^{n+1}$).

**Definition 24.4.** Given two abstract logics $\langle L_1, \models_{L_1} \rangle$ and $\langle L_2, \models_{L_2} \rangle$ we say that the latter is *at least as expressive* as the former, written $\langle L_1, \models_{L_1} \rangle \leq \langle L_2, \models_{L_2} \rangle$, if for each language $\mathcal{L}$ and sentence $\alpha \in L_1(\mathcal{L})$ there is a sentence $\beta \in L_2(\mathcal{L})$ such that $\mathrm{Mod}^{\mathcal{L}}_{L_1}(\alpha) = \mathrm{Mod}^{\mathcal{L}}_{L_2}(\beta)$. The logics $\langle L_1, \models_{L_1} \rangle$ and $\langle L_2, \models_{L_2} \rangle$ are *equivalent* if $\langle L_1, \models_{L_1} \rangle \leq \langle L_2, \models_{L_2} \rangle$ and $\langle L_2, \models_{L_2} \rangle \leq \langle L_1, \models_{L_1} \rangle$.

*Remark* 5. First-order logic, i.e., the abstract logic $\langle F, \models \rangle$, is normal. In fact, the above properties are mostly straightforward for first-order logic. We just remark that the expansion property comes down to extensionality, and that the relativization of a sentence $\alpha$ to $R(x, c_1, \ldots, c_n)$ is obtained by replacing each subformula $\forall x\, \beta$ by $\forall x\, (R(x, c_1, \ldots, c_n) \to \beta)$. Moreover, if $\langle L, \models_L \rangle$ is normal, then $\langle F, \models \rangle \leq \langle L, \models_L \rangle$, as can be can shown by induction on first-order formulas. Accordingly, with no loss in generality, we can assume that every first-order sentence belongs to every normal logic.

## 24.3 Compactness and Löwenheim-Skolem Properties

We now give the obvious extensions of compactness and Löwenheim-Skolem to the case of abstract logics.

**Definition 24.5.** An abstract logic $\langle L, \models_L \rangle$ has the *Compactness Property* if each set $\Gamma$ of $L(\mathcal{L})$-sentences is satisfiable whenever each finite $\Gamma_0 \subseteq \Gamma$ is satisfiable.

**Definition 24.6.** $\langle L, \models_L \rangle$ has the *Downward Löwenheim-Skolem property* if any satisfiable $\Gamma$ has an enumerable model.

The notion of partial isomorphism from Definition 21.15 is purely "algebraic" (i.e., given without reference to the sentences of the language but only to the constants provided by the language $\mathcal{L}$ of the structures), and hence it applies to the case of abstract logics. In case of first-order logic, we know from Theorem 21.17 that if two structures are partially isomorphic then they are elementarily equivalent. That proof does not carry over to abstract logics, for induction on formulas need not be available for arbitrary $\alpha \in L(\mathcal{L})$, but the theorem is true nonetheless, provided the Löwenheim-Skolem property holds.

**Theorem 24.7.** *Suppose $\langle L, \models_L \rangle$ is a normal logic with the Löwenheim-Skolem property. Then any two structures that are partially isomorphic are elementarily equivalent in $\langle L, \models_L \rangle$.*

*Proof.* Suppose $\mathfrak{M} \simeq_p \mathfrak{N}$, but for some $\alpha$ also $\mathfrak{M} \models_L \alpha$ while $\mathfrak{N} \not\models_L \alpha$. By the Isomorphism Property we can assume that $|\mathfrak{M}|$ and $|\mathfrak{N}|$ are disjoint, and by the Expansion Property we can assume that $\alpha \in L(\mathcal{L})$ for a finite language $\mathcal{L}$. Let $\mathcal{I}$ be a set of partial isomorphisms between $\mathfrak{M}$ and $\mathfrak{N}$, and with no loss of generality also assume that if $p \in \mathcal{I}$ and $q \subseteq p$ then also $q \in \mathcal{I}$.

$|\mathfrak{M}|^{<\omega}$ is the set of finite sequences of elements of $|\mathfrak{M}|$. Let $S$ be the ternary relation over $|\mathfrak{M}|^{<\omega}$ representing concatenation, i.e., if $\mathbf{a}, \mathbf{b}, \mathbf{c} \in |\mathfrak{M}|^{<\omega}$ then $S(\mathbf{a}, \mathbf{b}, \mathbf{c})$ holds if and only if $\mathbf{c}$ is the concatenation of $\mathbf{a}$ and $\mathbf{b}$; and let $T$ be the ternary relation such that $T(\mathbf{a}, b, \mathbf{c})$ holds for $b \in M$ and $\mathbf{a}, \mathbf{c} \in |\mathfrak{M}|^{<\omega}$ if and only if $\mathbf{a} = a_1, \ldots a_n$ and $\mathbf{c} = a_1, \ldots a_n, b$. Pick new 3-place predicate symbols $P$ and $Q$ and form the structure $\mathfrak{M}^*$ having the universe $|\mathfrak{M}| \cup |\mathfrak{M}|^{<\omega}$, having $\mathfrak{M}$ as a substructure, and interpreting $P$ and $Q$ by the concatenation relations $S$ and $T$ (so $\mathfrak{M}^*$ is in the language $\mathcal{L} \cup \{P, Q\}$).

Define $|\mathfrak{N}|^{<\omega}$, $S'$, $T'$, $P'$, $Q'$ and $\mathfrak{N}^*$ analogously. Since by hypothesis $\mathfrak{M} \simeq_p \mathfrak{N}$, there is a relation $I$ between $|\mathfrak{M}|^{<\omega}$ and $|\mathfrak{N}|^{<\omega}$ such that $I(\mathbf{a}, \mathbf{b})$ holds if and only if $\mathbf{a}$ and $\mathbf{b}$ are isomorphic and satisfy the back-and-forth condition of Definition 21.15. Now, let $\mathfrak{M}$ be the structure whose domain is the union of the domains of $\mathfrak{M}^*$ and $\mathfrak{N}^*$, having $\mathfrak{M}^*$ and $\mathfrak{N}^*$ as substructures, in the language with one extra binary predicate symbol $R$ interpreted by the relation $I$ and predicate symbols denoting the domains $|\mathfrak{M}|^*$ and $|\mathfrak{N}|*$.

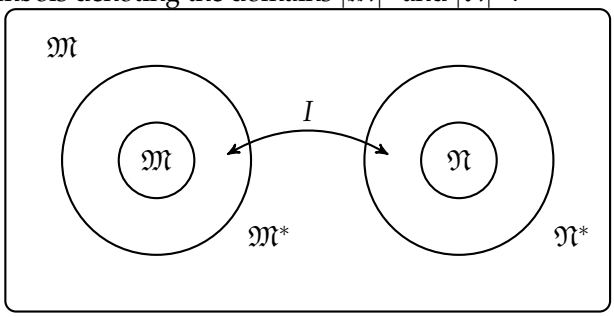



Figure 24.1: The structure $\mathfrak{M}$ with the internal partial isomorphism.

The crucial observation is that in the language of the structure $\mathfrak{M}$ there is a *first-order* sentence $\theta_1$ true in $\mathfrak{M}$ saying that $\mathfrak{M} \models_L \alpha$ and $\mathfrak{N} \not\models_L \alpha$ (this requires the Relativization Property), as well as a *first-order* sentence $\theta_2$ true in $\mathfrak{M}$ saying that $\mathfrak{M} \simeq_p \mathfrak{N}$ via the partial isomorphism $I$. By the Löwenheim-Skolem Property, $\theta_1$ and $\theta_2$ are jointly true in an enumerable model $\mathfrak{M}_0$ containing partially isomorphic substructures $\mathfrak{M}_0$ and $\mathfrak{N}_0$ such that $\mathfrak{M}_0 \models_L \alpha$ and $\mathfrak{N}_0 \not\models_L \alpha$. But enumerable partially isomorphic structures are in fact isomorphic by Theorem 21.16, contradicting the Isomorphism Property of normal abstract logics. □

## 24.4 Lindström's Theorem

**Lemma 24.8.** *Suppose $\alpha \in L(\mathcal{L})$, with $\mathcal{L}$ finite, and assume also that there is an $n \in \mathbb{N}$ such that for any two structures $\mathfrak{M}$ and $\mathfrak{N}$, if $\mathfrak{M} \equiv_n \mathfrak{N}$ and $\mathfrak{M} \models_L \alpha$ then also $\mathfrak{N} \models_L \alpha$. Then $\alpha$ is equivalent to a first-order sentence, i.e., there is a first-order $\theta$ such that $\mathrm{Mod}_L(\alpha) = \mathrm{Mod}_L(\theta)$.*

*Proof.* Let $n$ be such that any two $n$-equivalent structures $\mathfrak{M}$ and $\mathfrak{N}$ agree on the value assigned to $\alpha$. Recall Proposition 21.19: there are only finitely many first-order sentences in a finite language that have quantifier rank no greater than $n$, up to logical equivalence. Now, for each fixed structure $\mathfrak{M}$ let $\theta_{\mathfrak{M}}$ be the conjunction of all first-order sentences $\alpha$ true in $\mathfrak{M}$ with $\mathrm{qr}(\alpha) \leq n$ (this conjunction is finite), so that $\mathfrak{N} \models \theta_{\mathfrak{M}}$ if and only if $\mathfrak{N} \equiv_n \mathfrak{M}$. Then put $\theta = \bigvee\{\theta_{\mathfrak{M}} : \mathfrak{M} \models_L \alpha\}$; this disjunction is also finite (up to logical equivalence).

The conclusion $\mathrm{Mod}_L(\alpha) = \mathrm{Mod}_L(\theta)$ follows. In fact, if $\mathfrak{N} \models_L \theta$ then for some $\mathfrak{M} \models_L \alpha$ we have $\mathfrak{N} \models \theta_{\mathfrak{M}}$, whence also $\mathfrak{N} \models_L \alpha$ (by the hypothesis of the lemma). Conversely, if $\mathfrak{N} \models_L \alpha$ then $\theta_{\mathfrak{N}}$ is a disjunct in $\theta$, and since $\mathfrak{N} \models \theta_{\mathfrak{N}}$, also $\mathfrak{N} \models_L \theta$. □

**Theorem 24.9** (Lindström's Theorem)**.** *Suppose $\langle L, \models_L\rangle$ has the Compactness and the Löwenheim-Skolem Properties. Then $\langle L, \models_L\rangle \leq \langle F, \models\rangle$ (so $\langle L, \models_L\rangle$ is equivalent to first-order logic).*

*Proof.* By Lemma 24.8, it suffices to show that for any $\alpha \in L(\mathcal{L})$, with $\mathcal{L}$ finite, there is $n \in \mathbb{N}$ such that for any two structures $\mathfrak{M}$ and $\mathfrak{N}$: if $\mathfrak{M} \equiv_n \mathfrak{N}$ then $\mathfrak{M}$ and $\mathfrak{N}$ agree on $\alpha$. For then $\alpha$ is equivalent to a first-order sentence, from which $\langle L, \models_L\rangle \leq \langle F, \models\rangle$ follows. Since we are working in a finite, purely relational language, by Theorem 21.23 we can replace the statement that $\mathfrak{M} \equiv_n \mathfrak{N}$ by the corresponding algebraic statement that $I_n(\emptyset, \emptyset)$.

Given $\alpha$, suppose towards a contradiction that for each $n$ there are structures $\mathfrak{M}_n$ and $\mathfrak{N}_n$ such that $I_n(\emptyset, \emptyset)$, but (say) $\mathfrak{M}_n \models_L \alpha$ whereas $\mathfrak{N}_n \not\models_L \alpha$. By the Isomorphism Property we can assume that all the $\mathfrak{M}_n$'s interpret the constants of the language by the same objects; furthermore, since there are only finitely many atomic sentences in the language, we may also assume that they satisfy the same atomic sentences (we can take a subsequence of the $\mathfrak{M}$'s otherwise). Let $\mathfrak{M}$ be the union of all the $\mathfrak{M}_n$'s, i.e., the unique minimal structure having each $\mathfrak{M}_n$ as a substructure. As in the proof of Theorem 24.7, let $\mathfrak{M}^*$ be the extension of $\mathfrak{M}$ with domain $|\mathfrak{M}| \cup |\mathfrak{M}|^{<\omega}$, in the expanded language comprising the concatenation predicates $P$ and $Q$.

Similarly, define $\mathfrak{N}_n$, $\mathfrak{N}$ and $\mathfrak{N}^*$. Now let $\mathfrak{M}$ be the structure whose domain comprises the domains of $\mathfrak{M}^*$ and $\mathfrak{N}^*$ as well as the natural numbers $\mathbb{N}$ along with their natural ordering $\leq$, in the language with extra predicates representing the domains $|\mathfrak{M}|$, $|\mathfrak{N}|$, $|\mathfrak{M}|^{<\omega}$ and $|\mathfrak{N}|^{<\omega}$ as well as predicates coding the domains of $\mathfrak{M}_n$ and $\mathfrak{N}_n$ in the sense that:

$$|\mathfrak{M}_n| = \{a \in |\mathfrak{M}| : R(a, n)\}; \qquad |\mathfrak{N}_n| = \{a \in |\mathfrak{N}| : S(a, n)\};$$
$$|\mathfrak{M}|_n^{<\omega} = \{a \in |\mathfrak{M}|^{<\omega} : R(a, n)\}; \qquad |\mathfrak{N}|_n^{<\omega} = \{a \in |\mathfrak{N}|^{<\omega} : S(a, n)\}.$$

The structure $\mathfrak{M}$ also has a ternary relation $J$ such that $J(n, \mathbf{a}, \mathbf{b})$ holds if and only if $I_n(\mathbf{a}, \mathbf{b})$.

Now there is a sentence $\theta$ in the language $\mathcal{L}$ augmented by $R$, $S$, $J$, etc., saying that $\leq$ is a discrete linear ordering with first but no last element and

such that $\mathfrak{M}_n \models \alpha$, $\mathfrak{N}_n \not\models \alpha$, and for each $n$ in the ordering, $J(n, \mathbf{a}, \mathbf{b})$ holds if and only if $I_n(\mathbf{a}, \mathbf{b})$.

Using the Compactness Property, we can find a model $\mathfrak{M}^*$ of $\theta$ in which the ordering contains a non-standard element $n^*$. In particular then $\mathfrak{M}^*$ will contain substructures $\mathfrak{M}_{n^*}$ and $\mathfrak{N}_{n^*}$ such that $\mathfrak{M}_{n^*} \models_L \alpha$ and $\mathfrak{N}_{n^*} \not\models_L \alpha$. But now we can define a set $\mathcal{I}$ of pairs of $k$-tuples from $|\mathfrak{M}_{n^*}|$ and $|\mathfrak{N}_{n^*}|$ by putting $\langle \mathbf{a}, \mathbf{b} \rangle \in \mathcal{I}$ if and only if $J(n^* - k, \mathbf{a}, \mathbf{b})$, where $k$ is the length of $\mathbf{a}$ and $\mathbf{b}$. Since $n^*$ is non-standard, for each standard $k$ we have that $n^* - k > 0$, and the set $\mathcal{I}$ witnesses the fact that $\mathfrak{M}_{n^*} \simeq_p \mathfrak{N}_{n^*}$. But by Theorem 24.7, $\mathfrak{M}_{n^*}$ is $L$-equivalent to $\mathfrak{N}_{n^*}$, a contradiction. □

# Part V

# Computability

This part is based on Jeremy Avigad's notes on computability theory. Only the chapter on recursive functions contains exercises yet, and everything could stand to be expanded with motivation, examples, details, and exercises.

# Chapter 25

# Recursive Functions

These are Jeremy Avigad's notes on recursive functions, revised and expanded by Richard Zach. This chapter does contain some exercises, and can be included independently to provide the basis for a discussion of arithmetization of syntax.

## 25.1 Introduction

In order to develop a mathematical theory of computability, one has to, first of all, develop a *model* of computability. We now think of computability as the kind of thing that computers do, and computers work with symbols. But at the beginning of the development of theories of computability, the paradigmatic example of computation was *numerical* computation. Mathematicians were always interested in number-theoretic functions, i.e., functions $f : \mathbb{N}^n \to \mathbb{N}$ that can be computed. So it is not surprising that at the beginning of the theory of computability, it was such functions that were studied. The most familiar examples of computable numerical functions, such as addition, multiplication, exponentiation (of natural numbers) share an interesting feature: they can be defined *recursively*. It is thus quite natural to attempt a general definition of *computable function* on the basis of recursive definitions. Among the many possible ways to define number-theoretic functions recursively, one particulalry simple pattern of definition here becomes central: so-called *primitive recursion*.

In addition to computable functions, we might be interested in computable sets and relations. A set is computable if we can compute the answer to whether or not a given number is an element of the set, and a relation is computable iff we can compute whether or not a tuple $\langle n_1, \ldots, n_k \rangle$ is an element of the relation. By considering the *characteristic function* of a set or relation, discussion of computable sets and relations can be subsumed under that of

computable functions. Thus we can define primitive recursive relations as well, e.g., the relation "$n$ evenly divides $m$" is a primitive recursive relation.

Primitive recursive functions—those that can be defined using just primitive recursion—are not, however, the only computable number-theoretic functions. Many generalizations of primitive recursion have been considered, but the most powerful and widely-accepted additional way of computing functions is by unbounded search. This leads to the definition of *partial recursive functions*, and a related definition to *general recursive functions*. General recursive functions are computable and total, and the definition characterizes exactly the partial recursive functions that happen to be total. Recursive functions can simulate every other model of computation (Turing machines, lambda calculus, etc.) and so represent one of the many accepted models of computation.

## 25.2 Primitive Recursion

A characteristic of the natural numbers is that every natural number can be reached from 0 by applying the successor operation "$+1$" finitely many times—any natural number is either 0 or the successor of ... the successor of 0. One way to specify a function $f\colon \mathbb{N} \to \mathbb{N}$ that makes use of this fact is this: (a) specify what the value of $f$ is for argument 0, and (b) also specify how to, given the value of $f(x)$, compute the value of $f(x+1)$. For (a) tells us directly what $f(0)$ is, so $f$ is defined for 0. Now, using the instruction given by (b) for $x = 0$, we can compute $f(1) = f(0+1)$ from $f(0)$. Using the same instructions for $x = 1$, we compute $f(2) = f(1+1)$ from $f(1)$, and so on. For every natural number $x$, we'll eventually reach the step where we define $f(x)$ from $f(x+1)$, and so $f(x)$ is defined for all $x \in \mathbb{N}$.

For instance, suppose we specify $h\colon \mathbb{N} \to \mathbb{N}$ by the following two equations:

$$\begin{aligned} h(0) &= 1 \\ h(x+1) &= 2 \cdot h(x). \end{aligned}$$

If we already know how to multiply, then these equations give us the information required for (a) and (b) above. Successively the second equation, we get that

$$\begin{aligned} h(1) &= 2 \cdot h(0) = 2, \\ h(2) &= 2 \cdot h(1) = 2 \cdot 2, \\ h(3) &= 2 \cdot h(2) = 2 \cdot 2 \cdot 2, \\ &\vdots \end{aligned}$$

We see that the function $h$ we have specified is $h(x) = 2^x$.

The characteristic feature of the natural numbers guarantees that there is only one function $d$ that meets these two criteria. A pair of equations like these is called a *definition by primitive recursion* of the function $d$. It is so-called because we define $f$ "recursively," i.e., the definition, specifically the second equation, involves $f$ itself on the right-hand-side. It is "primitive" because in defining $f(x+1)$ we only use the value $f(x)$, i.e., the immediately preceding value. This is the simplest way of defining a function on $\mathbb{N}$ recursively.

We can define even more fundamental functions like addition and multiplication by primitive recursion. In these cases, however, the functions in question are 2-place. We fix one of the argument places, and use the other for the recursion. E.g, to define $\text{add}(x, y)$ we can fix $x$ and define the value first for $y = 0$ and then for $y + 1$ in terms of $y$. Since $x$ is fixed, it will appear on the left and on the right side of the defining equations.

$$\begin{aligned}\text{add}(x, 0) &= x\\ \text{add}(x, y+1) &= \text{add}(x, y) + 1\end{aligned}$$

These equations specify the value of add for all $x$ *and* $y$. To find $\text{add}(2, 3)$, for instance, we apply the defining equations for $x = 2$, using the first to find $\text{add}(2, 0) = 2$, then using the second to successively find $\text{add}(2, 1) = 2 + 1 = 3$, $\text{add}(2, 2) = 3 + 1 = 4$, $\text{add}(2, 3) = 4 + 1 = 5$.

In the definition of add we used $+$ on the right-hand-side of the second equation, but only to add 1. In other words, we used the successor function $\text{succ}(z) = z + 1$ and applied it to the previous value $\text{add}(x, y)$ to define $\text{add}(x, y + 1)$. So we can think of the recursive definition as given in terms of a single function which we apply to the previous value. However, it doesn't hurt—and sometimes is necessary—to allow the function to depend not just on the previous value but also on $x$ and $y$. Consider:

$$\begin{aligned}\text{mult}(x, 0) &= 0\\ \text{mult}(x, y+1) &= \text{add}(\text{mult}(x, y), x).\end{aligned}$$

This is a primitive recursive definition of a function mult by applying the function add to both the preceding value $\text{mult}(x, y)$ and the first argument $x$. It also defines the function $\text{mult}(x, y)$ for all arguments $x$ and $y$. For instance, $\text{mult}(2, 3)$ is determined by successively computing $\text{mult}(2, 0)$, $\text{mult}(2, 1)$, $\text{mult}(2, 2)$, and $\text{mult}(2, 3)$:

$$\begin{aligned}&\text{mult}(2, 0) = 0\\ &\text{mult}(2, 1) = \text{mult}(2, 0 + 1) = \text{add}(\text{mult}(2, 0), 2) = \text{add}(0, 2) = 2\\ &\text{mult}(2, 2) = \text{mult}(2, 1 + 1) = \text{add}(\text{mult}(2, 1), 2) = \text{add}(2, 2) = 4\\ &\text{mult}(2, 3) = \text{mult}(2, 2 + 1) = \text{add}(\text{mult}(2, 2), 2) = \text{add}(4, 2) = 6.\end{aligned}$$

The general pattern then is this: to give a primitive recursive definition of a function $h(x_0, \ldots, x_{k-1}, y)$, we provide two equations. The first defines the

value of $h(x_0, \ldots, x_{k-1}, 0)$ without reference to $f$. The second defines the value of $h(x_0, \ldots, x_{k-1}, y+1)$ in terms of $h(x_0, \ldots, x_{k-1}, y)$, the other arguments $x_0$, $\ldots$, $x_{k-1}$, and $y$. Only the immediately preceding value of $h$ may be used in that second equation. If we think of the operations given by the right-hand-sides of these two equations as themselves being functions $f$ and $g$, then the pattern to define a new function $h$ by primitive recursion is this:

$$\begin{aligned} h(x_0, \ldots, x_{k-1}, 0) &= f(x_0, \ldots, x_k) \\ h(x_0, \ldots, x_{k-1}, y+1) &= g(x_0, \ldots, x_{k-1}, y, h(x_0, \ldots, x_{k-1}, y)). \end{aligned}$$

In the case of add, we have $k = 0$ and $f(x_0) = x_0$ (the identity function), and $g(x_0, y, z) = z + 1$ (the 3-place function that returns the successor of its third argument):

$$\begin{aligned} \text{add}(x_0, 0) &= f(x_0) = x_0 \\ \text{add}(x_0, y+1) &= g(x_0, y, \text{add}(x_0, y)) = \text{succ}(\text{add}(x_0, y)) \end{aligned}$$

In the case of mult, we have $f(x_0) = 0$ (the constant function always returning 0) and $g(x_0, y, z) = \text{add}(z, x_0)$ (the 3-place function that returns the sum of its last and first argument):

$$\begin{aligned} \text{mult}(x_0, 0) &= f(x_0) = 0 \\ \text{mult}(x_0, y+1) &= g(x_0, y, \text{mult}(x_0, y)) = \text{add}(\text{mult}(x_0, y), x_0). \end{aligned}$$

## 25.3 Composition

If $f$ and $g$ are two one-place functions of natural numbers, we can compose them: $h(x) = g(f(x))$. The new function $h(x)$ is then defined by *composition* from the functions $f$ and $g$. We'd like to generalize this to functions of more than one argument.

Here's one way of doing this: suppose $f$ is a $k$-place function, and $g_0$, $\ldots$, $g_{k-1}$ are $k$ functions which are all $n$-place. Then we can define a new $n$-place function $h$ as follows:

$$h(x_0, \ldots, x_{n-1}) = f(g_0(x_0, \ldots, x_{n-1}), \ldots, g_{k-1}(x_0, \ldots, x_{n-1})).$$

If $f$ and all $g_i$ are computable, so is $h$: To compute $h(x_0, \ldots, x_{n-1})$, first compute the values $y_i = g_i(x_0, \ldots, x_{n-1})$ for each $i = 0, \ldots, k-1$. Then feed these values into $f$ to compute $h(x_0, \ldots, x_{k-1}) = f(y_0, \ldots, y_{k-1})$.

This may seem like an overly restrictive characterization of what happens when we compute a new function using some existing ones. For one thing, sometimes we do not use all the arguments of a function, as when we defined $g(x, y, z) = \text{succ}(z)$ for use in the primitive recursive definition of add. Suppose we are allowed use of the following functions:

$$P_i^n(x_0, \ldots, x_{n-1}) = x_i.$$

The functions $P_i^k$ are called *projection* functions: $P_i^n$ is an $n$-place function. Then $g$ can be defined as

$$g(x, y, z) = \text{succ}(P_2^3)$$

Here the role of $f$ is played by the 1-place function succ, so $k = 1$. And we have one 3-place function $P_2^3$ which plays the role of $g_0$. The result is a 3-place function that returns the successor of the third argument.

The projection functions also allow us to define new functions by reordering or identifying arguments. For instance, the function $h(x) = \text{add}(x, x)$ can be defined as

$$h(x_0) = \text{add}(P_0^1(x_0), P_0^1(x_0))$$

Here $k = 2$, $n = 1$, the role of $f(y_0, y_1)$ is played by add, and the roles of $g_0(x_0)$ and $g_1(x_0)$ are both played by $P_0^1(x_0)$, the one-place projection function (aka the identity function).

If $f(y_0, y_1)$ is a function we already have, we can define the function $h(x_0, x_1) = f(x_1, x_0)$ by

$$h(x_0, x_1) = f(P_1^2(x_0, x_1), P_0^2(x_0, x_1)).$$

Here $k = 2$, $n = 2$, and the roles of $g_0$ and $g_1$ are played by $P_1^2$ and $P_0^2$, respectively.

You may also worry that $g_0, \ldots, g_{k-1}$ are all required to have the same arity $n$. (Remember that the *arity* of a function is the number of arguments; an $n$-place function has arity $n$.) But adding the projection functions provides the desired flexibility. For example, suppose $f$ and $g$ are 3-place functions and $h$ is the 2-place function defined by

$$h(x, y) = f(x, g(x, x, y), y).$$

The definition of $h$ can be rewritten with the projection functions, as

$$h(x, y) = f(P_0^2(x, y), g(P_0^2(x, y), P_0^2(x, y), P_1^2(x, y)), P_1^2(x, y)).$$

Then $h$ is the composition of $f$ with $P_0^2$, $l$, and $P_1^2$, where

$$l(x, y) = g(P_0^2(x, y), P_0^2(x, y), P_1^2(x, y)),$$

i.e., $l$ is the composition of $g$ with $P_0^2$, $P_0^2$, and $P_1^2$.

## 25.4 Primitive Recursion Functions

Let us record again how we can define new functions from existing ones using primitive recursion and composition.

**Definition 25.1.** Suppose $f$ is a $k$-place function ($k \geq 1$) and $g$ is a $(k+2)$-place function. The function defined by *primitive recursion from $f$ and $g$* is the $(k+1)$-place function $h$ defined by the equations

$$\begin{aligned} h(x_0, \ldots, x_{k-1}, y) &= f(x_0, \ldots, x_{k-1}) \\ h(x_0, \ldots, x_{k-1}, y+1) &= g(x_0, \ldots, x_{k-1}, y, h(x_0, \ldots, x_{k-1}, y)) \end{aligned}$$

**Definition 25.2.** Suppose $f$ is a $k$-place function, and $g_0, \ldots, g_{k-1}$ are $k$ functions which are all $n$-place. The function defined by *composition from $f$ and $g_0, \ldots, g_{k-1}$* is the $n$-place function $h$ defined by

$$h(x_0, \ldots, x_{n-1}) = f(g_0(x_0, \ldots, x_{n-1}), \ldots, g_{k-1}(x_0, \ldots, x_{n-1})).$$

In addition to succ and the projection functions

$$P_i^n(x_0, \ldots, x_{n-1}) = x_i,$$

for each natural number $n$ and $i < n$, we will include among the primitive recursive functions the function $\text{zero}(x) = 0$.

**Definition 25.3.** The set of primitive recursive functions is the set of functions from $\mathbb{N}^n$ to $\mathbb{N}$, defined inductively by the following clauses:

1. zero is primitive recursive.
2. succ is primitive recursive.
3. Each projection function $P_i^n$ is primitive recursive.
4. If $f$ is a $k$-place primitive recursive function and $g_0, \ldots, g_{k-1}$ are $n$-place primitive recursive functions, then the composition of $f$ with $g_0, \ldots, g_{k-1}$ is primitive recursive.
5. If $f$ is a $k$-place primitive recursive function and $g$ is a $k+2$-place primitive recursive function, then the function defined by primitive recursion from $f$ and $g$ is primitive recursive.

Put more concisely, the set of primitive recursive functions is the smallest set containing zero, succ, and the projection functions $P_j^n$, and which is closed under composition and primitive recursion.

Another way of describing the set of primitive recursive functions is by defining it in terms of "stages." Let $S_0$ denote the set of starting functions: zero, succ, and the projections. These are the primitive recursive functions of stage 0. Once a stage $S_i$ has been defined, let $S_{i+1}$ be the set of all functions you get by applying a single instance of composition or primitive recursion to functions already in $S_i$. Then

$$S = \bigcup_{i \in \mathbb{N}} S_i$$

is the set of all primitive recursive functions

Let us verify that add is a primitive recursive function.

**Proposition 25.4.** *The addition function* $\text{add}(x, y) = x + y$ *is primitive recursive.*

*Proof.* We already have a primitive recursive definition of add in terms of two functions $f$ and $g$ which matches the format of Definition 25.1:

$$\begin{aligned}\text{add}(x_0, 0) &= f(x_0) = x_0\\ \text{add}(x_0, y+1) &= g(x_0, y, \text{add}(x_0, y)) = \text{succ}(\text{add}(x_0, y))\end{aligned}$$

So add is primitive recursive provided $f$ and $g$ are as well. $f(x_0) = x_0 = P^1_0(x_0)$, and the projection functions count as primitive recursive, so $f$ is primitive recursive. The function $g$ is the three-place function $g(x_0, y, z)$ defined by

$$g(x_0, y, z) = \text{succ}(z).$$

This does not yet tell us that $g$ is primitive recursive, since $g$ and succ are not quite the same function: succ is one-place, and $g$ has to be three-place. But we can define $g$ "officially" by composition as

$$g(x_0, y, z) = \text{succ}(P^3_2(x_0, y, z))$$

Since succ and $P^3_2$ count as primitive recursive functions, $g$ does as well, since it can be defined by composition from primitive recursive functions. □

**Proposition 25.5.** *The multiplication function* $\text{mult}(x, y) = x \cdot y$ *is primitive recursive.*

*Proof.* Exercise. □

**Example 25.6.** Here's our very first example of a primitive recursive definition:

$$\begin{aligned}h(0) &= 1\\ h(y+1) &= 2 \cdot h(y).\end{aligned}$$

This function cannot fit into the form required by Definition 25.1, since $k = 0$. The definition also involves the constants 1 and 2. To get around the first problem, let's introduce a dummy argument and define the function $h'$:

$$\begin{aligned}h'(x_0, 0) &= f(x_0) = 1\\ h'(x_0, y+1) &= g(x_0, y, h'(x_0, y)) = 2 \cdot h'(x_0, y).\end{aligned}$$

The function $f(x_0) = 1$ can be defined from succ and zero by composition: $f(x_0) = \text{succ}(\text{zero}(x_0))$. The function $g$ can be defined by composition from $g'(z) = 2 \cdot z$ and projections:

$$g(x_0, y, z) = g'(P^3_2(x_0, y, z))$$

and $g'$ in turn can be defined by composition as

$$g'(z) = \text{mult}(g''(z), P^1_0(z))$$

and

$$g''(z) = \text{succ}(f(z)),$$

where $f$ is as above: $f(z) = \text{succ}(\text{zero}(z))$. Now that we have $h'$ we can use composition again to let $h(y) = h'(P_0^1(y), P_0^1(y))$. This shows that $h$ can be defined from the basic functions using a sequence of compositions and primitive recursions, so $h$ is primitive recursive.

## 25.5 Primitive Recursion Notations

One advantage to having the precise inductive description of the primitive recursive functions is that we can be systematic in describing them. For example, we can assign a "notation" to each such function, as follows. Use symbols zero, succ, and $P_i^n$ for zero, successor, and the projections. Now suppose $f$ is defined by composition from a $k$-place function $h$ and $n$-place functions $g_0$, ..., $g_{k-1}$, and we have assigned notations $H$, $G_0$, ..., $G_{k-1}$ to the latter functions. Then, using a new symbol $\text{Comp}_{k,n}$, we can denote the function $f$ by $\text{Comp}_{k,n}[H, G_0, \ldots, G_{k-1}]$. For the functions defined by primitive recursion, we can use analogous notations of the form $\text{Rec}_k[G, H]$, where $k+1$ is the arity of the function being defined. With this setup, we can denote the addition function by

$$\text{Rec}_2[P_0^1, \text{Comp}_{1,3}[\text{succ}, P_2^3]].$$

Having these notations sometimes proves useful.

## 25.6 Primitive Recursive Functions are Computable

Suppose a function $h$ is defined by primitive recursion

$$\begin{aligned} h(\vec{x}, 0) &= f(\vec{x}) \\ h(\vec{x}, y) &= g(\vec{x}, y, h(\vec{x}, y)) \end{aligned}$$

and suppose the functions $f$ and $g$ are computable. (We use $\vec{x}$ to abbreviate $x_0$, ..., $x_{k-1}$.) Then $h(\vec{x}, 0)$ can obviously be computed, since it is just $f(\vec{x})$ which we assume is computable. $h(\vec{x}, 1)$ can then also be computed, since $1 = 0 + 1$ and so $h(\vec{x}, 1)$ is just

$$h(\vec{x}, 1) = g(\vec{x}, 0, h(\vec{x}, 0)) = g(\vec{x}, 0, f(\vec{x})).$$

We can go on in this way and compute

$$\begin{aligned} h(\vec{x}, 2) &= g(\vec{x}, 1, h(\vec{x}, 1)) = g(\vec{x}, 1, g(\vec{x}, 0, f(\vec{x}))) \\ h(\vec{x}, 3) &= g(\vec{x}, 2, h(\vec{x}, 2)) = g(\vec{x}, 2, g(\vec{x}, 1, g(\vec{x}, 0, f(\vec{x})))) \\ h(\vec{x}, 4) &= g(\vec{x}, 3, h(\vec{x}, 3)) = g(\vec{x}, 3, g(\vec{x}, 2, g(\vec{x}, 1, g(\vec{x}, 0, f(\vec{x}))))) \\ &\vdots \end{aligned}$$

Thus, to compute $h(\vec{x}, y)$ in general, successively compute $h(\vec{x}, 0)$, $h(\vec{x}, 1)$, ..., until we reach $h(\vec{x}, y)$.

Thus, a primitive recursive definition yields a new computable function if the functions $f$ and $g$ are computable. Composition of functions also results in a computable function if the functions $f$ and $g_i$ are computable.

Since the basic functions zero, succ, and $P_i^n$ are computable, and composition and primitive recursion yield computable functions from computable functions, this means that every primitive recursive function is computable.

## 25.7 Examples of Primitive Recursive Functions

We already have some examples of primitive recursive functions: the addition and multiplication functions add and mult. The identity function $\mathrm{id}(x) = x$ is primitive recursive, since it is just $P_0^1$. The constant functions $\mathrm{const}_n(x) = n$ are primitive recursive since they can be defined from zero and succ by successive composition. This is useful when we want to use constants in primitive recursive definitions, e.g., if we want to define the function $f(x) = 2 \cdot x$ can obtain it by composition from $\mathrm{const}_n(x)$ and multiplication as $f(x) = \mathrm{mult}(\mathrm{const}_2(x), P_0^1(x))$. We'll make use of this trick from now on.

**Proposition 25.7.** *The exponentiation function* $\exp(x, y) = x^y$ *is primitive recursive.*

*Proof.* We can define exp primitive recursively as

$$\begin{aligned} \exp(x, 0) &= 1 \\ \exp(x, y + 1) &= \mathrm{mult}(x, \exp(x, y)). \end{aligned}$$

Strictly speaking, this is not a recursive definition from primitive recursive functions. Officially, though, we have:

$$\begin{aligned} \exp(x, 0) &= f(x) \\ \exp(x, y + 1) &= g(x, y, \exp(x, y)). \end{aligned}$$

where

$$\begin{aligned} f(x) &= \mathrm{succ}(\mathrm{zero}(x)) = 1 \\ g(x, y, z) &= \mathrm{mult}(P_0^3(x, y, z), P_2^3(x, y, z) = x \cdot z \end{aligned}$$

and so $f$ and $g$ are defined from primitive recursive functions by composition. □

**Proposition 25.8.** *The predecessor function* $\mathrm{pred}(y)$ *defined by*

$$\mathrm{pred}(y) = \begin{cases} 0 & \text{if } y = 0 \\ y - 1 & \text{otherwise} \end{cases}$$

*is primitive recursive.*

*Proof.* Note that

$$\begin{aligned}\text{pred}(0) &= 0\\ \text{pred}(y+1) &= y\end{aligned}$$

This is almost a primitive recursive definition. It does not, strictly speaking, fit into the pattern of definition by primitive recursion, since that pattern requires at least one extra argument $x$. It is also odd in that it does not actually use $\text{pred}(y)$ in the definition of $\text{pred}(y+1)$. But we can first define $\text{pred}'(x,y)$ by

$$\begin{aligned}\text{pred}'(x,0) &= \text{zero}(x) = 0\\ \text{pred}'(x,y+1) &= P^3_1(x,y,\text{pred}'(x,y)) = y\end{aligned}$$

and then define pred from it by composition, e.g., as $\text{pred}(x) = \text{pred}'(\text{zero}(x), P^1_0(x))$. □

**Proposition 25.9.** *The factorial function* $\text{fac}(x) = x! = 1 \cdot 2 \cdot 3 \cdot \cdots \cdot x$ *is primitive recursive.*

*Proof.* The obvious primitive recursive definition is

$$\begin{aligned}\text{fac}(0) &= 1\\ \text{fac}(y+1) &= !y \cdot (y+1)\end{aligned}$$

Officially, we have to first define a two-place function $h$

$$\begin{aligned}h(x,0) &= \text{const}_1(x)\\ h(x,y) &= g(x,y,h(x,y))\end{aligned}$$

where $g(x,y,z) = \text{mult}(P^3_2(x,y,z), \text{succ}(P^3_1(x,y,z)$ and then let

$$\text{fac}(y) = h(P^1_0(y), P^1_0(y))$$

From now on we'll be a bit more laissez-faire and not give the official definitions by composition and primitive recursion. □

**Proposition 25.10.** *Truncated subtraction,* $x \dot{-} y$*, defined by*

$$x \dot{-} y = \begin{cases} 0 & \text{if } x > y\\ x - y & \text{otherwise}\end{cases}$$

*is primitive recursive.*

*Proof.* We have

$$\begin{aligned} x \dot{-} 0 &= x \\ x \dot{-} (y+1) &= \text{pred}(x \dot{-} y) \end{aligned}$$

□

**Proposition 25.11.** *The distance between $x$ and $y$, $|x-y|$, is primitive recursive.*

*Proof.* We have $|x-y| = (x \dot{-} y) + (y \dot{-} x)$, so the distance can be defined by composition from $+$ and $\dot{-}$, which are primitive recursive. □

**Proposition 25.12.** *The maximum of $x$ and $y$, $\max(x,y)$, is primitive recursive.*

*Proof.* We can define $\max(x,y)$ by composition from $+$ and $\dot{-}$ by

$$\max(x,y) = x + (y \dot{-} x).$$

If $x$ is the maximum, i.e., $x \geq y$, then $y \dot{-} x = 0$, so $x + (y \dot{-} x) = x + 0 = x$. If $y$ is the maximum, then $y \dot{-} x = y - x$, and so $x + (y \dot{-} x) = x + (y - x) = y$. □

**Proposition 25.13.** *The minimum of $x$ and $y$, $\min(x,y)$, is primitive recursive.*

*Proof.* Prove Proposition 25.13. □

**Proposition 25.14.** *The set of primitive recursive functions is closed under the following two operations:*

1. *Finite sums: if $f(\vec{x}, z)$ is primitive recursive, then so is the function*

$$g(\vec{x}, y) = \sum_{z=0}^{y} f(\vec{x}, z).$$

2. *Finite products: if $f(\vec{x}, z)$ is primitive recursive, then so is the function*

$$h(\vec{x}, y) = \prod_{z=0}^{y} f(\vec{x}, z).$$

*Proof.* For example, finite sums are defined recursively by the equations

$$\begin{aligned} g(\vec{x}, 0) &= f(\vec{x}, 0) \\ g(\vec{x}, y+1) &= g(\vec{x}, y) + f(\vec{x}, y+1). \end{aligned}$$

□

## 25.8 Primitive Recursive Relations

**Definition 25.15.** A relation $R(\vec{x})$ is said to be primitive recursive if its characteristic function,

$$\chi_R(\vec{x}) = \begin{cases} 1 & \text{if } R(\vec{x}) \\ 0 & \text{otherwise} \end{cases}$$

is primitive recursive.

In other words, when one speaks of a primitive recursive relation $R(\vec{x})$, one is referring to a relation of the form $\chi_R(\vec{x}) = 1$, where $\chi_R$ is a primitive recursive function which, on any input, returns either 1 or 0. For example, the relation IsZero$(x)$, which holds if and only if $x = 0$, corresponds to the function $\chi_{\text{IsZero}}$, defined using primitive recursion by

$$\chi_{\text{IsZero}}(0) = 1, \quad \chi_{\text{IsZero}}(x+1) = 0.$$

It should be clear that one can compose relations with other primitive recursive functions. So the following are also primitive recursive:

1. The equality relation, $x = y$, defined by IsZero$(|x - y|)$
2. The less-than relation, $x \leq y$, defined by IsZero$(x \dot{-} y)$

**Proposition 25.16.** *The set of primitive recursive relations is closed under boolean operations, that is, if $P(\vec{x})$ and $Q(\vec{x})$ are primitive, so are*

1. $\neg R(\vec{x})$
2. $P(\vec{x}) \wedge Q(\vec{x})$
3. $P(\vec{x}) \vee Q(\vec{x})$
4. $P(\vec{x}) \to Q(\vec{x})$

*Proof.* Suppose $P(\vec{x})$ and $Q(\vec{x})$ are primitive recursive, i.e., their characteristic functions $\chi_P$ and $\chi_Q$ are. We have to show that the characteristic functions of $\neg R(\vec{x})$, etc., are also primitive recursive.

$$\chi_{\neg P}(\vec{x}) = \begin{cases} 0 & \text{if } \chi_P(\vec{x}) = 1 \\ 1 & \text{otherwise} \end{cases}$$

We can define $\chi_{\neg P}(\vec{x})$ as $1 \dot{-} \chi_P(\vec{x})$.

$$\chi_{P \wedge Q}(\vec{x}) = \begin{cases} 1 & \text{if } \chi_P(\vec{x}) = \chi_Q(\vec{x}) = 1 \\ 0 & \text{otherwise} \end{cases}$$

We can define $\chi_{P \wedge Q}(\vec{x})$ as $\chi_P(\vec{x}) \cdot \chi_Q(\vec{x})$ or as $\min(\chi_P(\vec{x}), \chi_Q(\vec{x}))$.

Similarly, $\chi_{P \vee Q}(\vec{x}) = \max(\chi_P(\vec{x}), \chi_Q(\vec{x}))$ and $\chi_{P \vee Q}(\vec{x}) = \max(1 \dot{-} \chi_P(\vec{x}), \chi_Q(\vec{x}))$. □

**Proposition 25.17.** *The set of primitive recursive relations is closed under bounded quantification, i.e., if $R(\vec{x}, z)$ is a primitive recursive relation, then so are the relations $(\forall z < y)\ R(\vec{x}, z)$ and $(\exists z < y)\ R(\vec{x}, z)$.*

*($(\forall z < y)\ R(\vec{x}, z)$ holds of $\vec{x}$ and $y$ if and only if $R(\vec{x}, z)$ holds for every $z$ less than $y$, and similarly for $(\exists z < y)\ R(\vec{x}, z)$.)*

*Proof.* By convention, we take $(\forall z < 0)\ R(\vec{x}, z)$ to be true (for the trivial reason that there are no $z$ less than 0) and $(\exists z < 0)\ R(\vec{x}, z)$ to be false. A universal quantifier functions just like a finite product or iterated minimum, i.e., if $P(\vec{x}, y) \Leftrightarrow (\forall z < y)\ R(\vec{x}, z)$ then $\chi_P(\vec{x}, y)$ can be defined by

$$\begin{aligned} \chi_P(\vec{x}, 0) &= 1 \\ \chi_P(\vec{x}, y+1) &= \min(\chi_P(\vec{x}, y), \chi_R(\vec{x}, y+1))). \end{aligned}$$

Bounded existential quantification can similarly be defined using max. Alternatively, it can be defined from bounded universal quantification, using the equivalence $(\exists z < y)\ R(\vec{x}, z) \leftrightarrow \neg(\forall z < y)\ \neg R(\vec{x}, z)$. Note that, for example, a bounded quantifier of the form $(\exists x \leq y)\ \ldots x \ldots$ is equivalent to $(\exists x < y + 1)\ \ldots x \ldots$. □

Another useful primitive recursive function is the conditional function, $\mathrm{cond}(x, y, z)$, defined by

$$\mathrm{cond}(x, y, z) = \begin{cases} y & \text{if } x = 0 \\ z & \text{otherwise.} \end{cases}$$

This is defined recursively by

$$\mathrm{cond}(0, y, z) = y, \quad \mathrm{cond}(x+1, y, z) = z.$$

One can use this to justify definitions of primitive recursive functions by cases from primitive recursive relations:

**Proposition 25.18.** *If $g_0(\vec{x}), \ldots, g_m(\vec{x})$ are functions, and $R_1(\vec{x}), \ldots, R_{m-1}(\vec{x})$ are primitive recursive relations, then the function $f$ defined by*

$$f(\vec{x}) = \begin{cases} g_0(\vec{x}) & \textit{if } R_0(\vec{x}) \\ g_1(\vec{x}) & \textit{if } R_1(\vec{x}) \textit{ and not } R_0(\vec{x}) \\ \vdots & \\ g_{m-1}(\vec{x}) & \textit{if } R_{m-1}(\vec{x}) \textit{ and none of the previous hold} \\ g_m(\vec{x}) & \textit{otherwise} \end{cases}$$

*is also primitive recursive.*

*Proof.* When $m = 1$, this is just the function defined by

$$f(\vec{x}) = \mathrm{cond}(\chi_{\neg R_0}(\vec{x}), g_0(\vec{x}), g_1(\vec{x})).$$

For $m$ greater than 1, one can just compose definitions of this form. □

## 25.9 Bounded Minimization

It is often useful to define a function as the least number satisfying some property or relation $P$. If $P$ is decidable, we can compute this function simply by trying out all the possible numbers, $0, 1, 2, \ldots$, until we find the least one satisfying $P$. This kind of unbounded search takes us out of the realm of primitive recursive functions. However, if we're only interested in the least number *less than some independently given bound*, we stay primitive recursive. In other words, and a bit more generally, suppose we have a primitive recursive relation $R(x, z)$. Consider the function that maps $x$ and $y$ to the least $z < y$ such that $R(x, z)$. It, too, can be computed, by testing whether $R(x, 0)$, $R(x, 1)$, $\ldots$, $R(x, y - 1)$. But why is it primitive recursive?

**Proposition 25.19.** *If $R(\vec{x}, z)$ is primitive recursive, so is the function $m_R(\vec{x}, y)$ which returns the least $z$ less than $y$ such that $R(\vec{x}, z)$ holds, if there is one, and $y$ otherwise. We will write the function $m_R$ as*

$$(\min z < y)\, R(\vec{x}, z),$$

*Proof.* Note than there can be no $z < 0$ such that $R(\vec{x}, z)$ since there is no $z < 0$ at all. So $m_R(\vec{x}, 0) = 0$.

In case the bound is of the form $y + 1$ we have three cases: (a) There is a $z < y$ such that $R(\vec{x}, z)$, in which case $m_R(\vec{x}, z) = m_R(\vec{x}, y)$. (b) There is no such $z < y$ but $R(\vec{x}, y)$ holds, then $m_R(\vec{x}, y + 1) = y$. (c) There is no $z < y + 1$ such that $R(\vec{x}, z)$, then $m_R(\vec{z}, y + 1) = y + 1$. Note that there is a $z < y$ such that $R(\vec{x}, z)$ iff $m_R(\vec{x}, y) \neq y$. So,

$$m_R(\vec{x}, 0) = 0$$

$$m_R(\vec{x}, y + 1) = \begin{cases} m_R(\vec{z}, y) & \text{if } m_R(\vec{x}, y) \neq y \\ y & \text{if } m_R(\vec{x}, y) = y \text{ and } R(\vec{x}, y) \\ y + 1 & \text{otherwise.} \end{cases}$$

□

## 25.10 Primes

Bounded quantification and bounded minimization provide us with a good deal of machinery to show that natural functions and relations are primitive recursive. For example, consider the relation "$x$ divides $y$", written $x \mid y$. The relation $x \mid y$ holds if division of $y$ by $x$ is possible without remainder, i.e., if $y$ is an integer multiple of $x$. (If it doesn't hold, i.e., the remainder when dividing $x$ by $y$ is $> 0$, we write $x \nmid y$.) In other words, $x \mid y$ iff for some $z$, $x \cdot z = y$. Obviously, any such $z$, if it exists, must be $\leq y$. So, we have that

$x \mid y$ iff for some $z \leq y$, $x \cdot z = y$. We can define the relation $x \mid y$ by bounded existential quantification from $=$ and multiplication by

$$x \mid y \Leftrightarrow (\exists z \leq y)\ (x \cdot z) = y.$$

We've thus shown that $x \mid y$ is primitive recursive.

A natural number $x$ is *prime* if it is neither 0 nor 1 and is only divisible by 1 and itself. In other words, prime numbers are such that, whenever $y \mid x$, either $y = 1$ or $y = x$. To test if $x$ is prime, we only have to check if $y \mid x$ for all $y \leq x$, since if $y > x$, then automatically $y \nmid x$. So, the relation $\text{Prime}(x)$, which holds iff $x$ is prime, can be defined by

$$\text{Prime}(x) \Leftrightarrow x \geq 2 \wedge (\forall y \leq x)\ (y \mid x \rightarrow y = 1 \vee y = x)$$

and is thus primitive recursive.

The primes are 2, 3, 5, 7, 11, etc. Consider the function $p(x)$ which returns the $x$th prime in that sequence, i.e., $p(0) = 2$, $p(1) = 3$, $p(2) = 5$, etc. (For convenience we will often write $p(x)$ as $p_x$ ($p_0 = 2$, $p_1 = 3$, etc.)

If we had a function nextPrime(x), which returns the first prime number larger than $x$, $p$ can be easily defined using primitive recursion:

$$\begin{aligned} p(0) &= 2 \\ p(x+1) &= \text{nextPrime}(p(x)) \end{aligned}$$

Since $\text{nextPrime}(x)$ is the least $y$ such that $y > x$ and $y$ is prime, it can be easily computed by unbounded search. But it can also be defined by bounded minimization, thanks to a result due to Euclid: there is always a prime number between $x$ and $x! + 1$.

$$\text{nextPrime(x)} = (\min y \leq x! + 1)\ (y > x \wedge \text{Prime}(y)).$$

This shows, that $\text{nextPrime}(x)$ and hence $p(x)$ are (not just computable but) primitive recursive.

(If you're curious, here's a quick proof of Euclid's theorem. Suppose $p_n$ is the largest prime $\leq x$ and consider the product $p = p_0 \cdot p_1 \cdot \cdots \cdot p_n$ of all primes $\leq x$. Either $p + 1$ is prime or there is a prime between $x$ and $p + 1$. Why? Suppose $p + 1$ is not prime. Then some prime number $q \mid p + 1$ where $q < p + 1$. None of the primes $\leq x$ divide $p + 1$. (By definition of $p$, each of the primes $p_i \leq x$ divides $p$, i.e., with remainder 0. So, each of the primes $p_i \leq x$ divides $p + 1$ with remainder 1, and so $p_i \nmid p + 1$.) Hence, $q$ is a prime $> x$ and $< p + 1$. And $p \leq x!$, so there is a prime $> x$ and $\leq x! + 1$.)

## 25.11 Sequences

The set of primitive recursive functions is remarkably robust. But we will be able to do even more once we have developed a adequate means of handling

*sequences*. We will identify finite sequences of natural numbers with natural numbers in the following way: the sequence $\langle a_0, a_1, a_2, \ldots, a_k \rangle$ corresponds to the number

$$p_0^{a_0+1} \cdot p_1^{a_1+1} \cdot p_2^{a_2+1} \cdot \ldots \cdot p_k^{a_k+1}.$$

We add one to the exponents to guarantee that, for example, the sequences $\langle 2, 7, 3 \rangle$ and $\langle 2, 7, 3, 0, 0 \rangle$ have distinct numeric codes. We can take both 0 and 1 to code the empty sequence; for concreteness, let $\Lambda$ denote 0.

The reason that this coding of sequences works is the so-called Fundamental Theorem of Arithmetic: every natural number $n \geq 2$ can be written in one and only one way in the form

$$n = p_0^{a_0} \cdot p_1^{a_1} \cdot \ldots \cdot p_k^{a_k}$$

with $a_k \geq 1$. This guarantees that the mapping $\langle\rangle(a_0, \ldots, a_k) = \langle a_0, \ldots, a_k \rangle$ is injective: different sequences are mapped to different numbers; to each number only at most one sequence corresponds.

We'll now show that the operations of determining the length of a sequence, determining its $i$th element, appending an element to a sequence, and concatenating two sequences, are all primitive recursive.

**Proposition 25.20.** *The function* $\mathrm{len}(s)$*, which returns the length of the sequence* $s$*, is primitive recursive.*

*Proof.* Let $R(i, s)$ be the relation defined by

$$R(i, s) \text{ iff } p_i \mid s \wedge p_{i+1} \nmid s.$$

$R$ is clearly primitive recursive. Whenever $s$ is the code of a non-empty sequence, i.e.,

$$s = p_0^{a_0+1} \cdot \ldots \cdot p_k^{a_k+1},$$

$R(i, s)$ holds if $p_i$ is the largest prime such that $p_i \mid s$, i.e., $i = k$. The length of $s$ thus is $i + 1$ iff $p_i$ is the largest prime that divides $s$, so we can let

$$\mathrm{len}(s) = \begin{cases} 0 & \text{if } s = 0 \text{ or } s = 1 \\ 1 + (\min i < s)\, R(i, s) & \text{otherwise} \end{cases}$$

We can use bounded minimization, since there is only one $i$ that satisfies $R(s, i)$ when $s$ is a code of a sequence, and if $i$ exists it is less than $s$ itself. □

**Proposition 25.21.** *The function* $\mathrm{append}(s, a)$*, which returns the result of appending* $a$ *to the sequence* $s$*, is primitive recursive.*

*Proof.* append can be defined by:

$$\mathrm{append}(s, a) = \begin{cases} 2^{a+1} & \text{if } s = 0 \text{ or } s = 1 \\ s \cdot p_{\mathrm{len}(s)}^{a+1} & \text{otherwise.} \end{cases}$$

□

**Proposition 25.22.** *The function* element$(s, i)$*, which returns the ith element of* $s$ *(where the initial element is called the* 0*th), or* 0 *if* $i$ *is greater than or equal to the length of* $s$*, is primitive recursive.*

*Proof.* Note that $a$ is the $i$th element of $s$ iff $p_i^{a+1}$ is the largest power of $p_i$ that divides $s$, i.e., $p_i^{a+1} \mid s$ but $p_i^{a+2} \nmid s$. So:

$$\text{element}(s, i) = \begin{cases} 0 & \text{if } i \geq \text{len}(s) \\ (\min a < s)\,(p_i^{a+2} \nmid s) & \text{otherwise.} \end{cases}$$

□

Instead of using the official names for the functions defined above, we introduce a more compact notation. We will use $(s)_i$ instead of element$(s, i)$, and $\langle s_0, \ldots, s_k \rangle$ to abbreviate

$$\text{append}(\text{append}(\ldots \text{append}(\Lambda, s_0) \ldots), s_k).$$

Note that if $s$ has length $k$, the elements of $s$ are $(s)_0, \ldots, (s)_{k-1}$.

**Proposition 25.23.** *The function* concat$(s, t)$*, which concatenates two sequences, is primitive recursive.*

*Proof.* We want a function concat with the property that

$$\text{concat}(\langle a_0, \ldots, a_k \rangle, \langle b_0, \ldots, b_l \rangle) = \langle a_0, \ldots, a_k, b_0, \ldots, b_l \rangle.$$

We'll use a "helper" function hconcat$(s, t, n)$ which concatenates the first $n$ symbols of $t$ to $s$. This function can be defined by primitive recursion as follows:

$$\begin{aligned} \text{hconcat}(s, t, 0) &= s \\ \text{hconcat}(s, t, n+1) &= \text{append}(\text{hconcat}(s, t, n), (t)_n) \end{aligned}$$

Then we can define concat by

$$\text{concat}(s, t) = \text{hconcat}(s, t, \text{len}(t)).$$

□

We will write $s \frown t$ instead of concat$(s, t)$.

It will be useful for us to be able to bound the numeric code of a sequence in terms of its length and its largest element. Suppose $s$ is a sequence of length $k$, each element of which is less than equal to some number $x$. Then $s$ has at most $k$ prime factors, each at most $p_{k-1}$, and each raised to at most $x + 1$ in the prime factorization of $s$. In other words, if we define

$$\text{sequenceBound}(x, k) = p_{k-1}^{k \cdot (x+1)},$$

then the numeric code of the sequence $s$ described above is at most sequenceBound$(x, k)$.

Having such a bound on sequences gives us a way of defining new functions using bounded search. For example, we can define concat using bounded search. All we need to do is write down a primitive recursive *specification* of the object (number of the concatenated sequence) we are looking for, and a bound on how far to look. The following works:

$$\begin{aligned}\text{concat}(s,t) = (&\min v < \text{sequenceBound}(s+t, \text{len}(s)+\text{len}(t))) \\ &(\text{len}(v) = \text{len}(s)+\text{len}(t) \wedge \\ &\quad (\forall i < \text{len}(s))\ ((v)_i = (s)_i) \wedge \\ &\quad (\forall j < \text{len}(t))\ ((v)_{\text{len}(s)+j} = (t)_j))\end{aligned}$$

**Proposition 25.24.** *The function* subseq$(s, i, n)$ *which returns the subsequence of $s$ of length $n$ beginning at the $i$th element, is primitive recursive.*

*Proof.* Exercise. □

## 25.12 Trees

Sometimes it is useful to represent trees as natural numbers, just like we can represent sequences by numbers and properties of and operations on them by primitive recursive relations and functions on their codes. We'll use sequences and their codes to do this. A tree can be either a single node (possibly with a label) or else a node (possibly with a label) connected to a number of subtrees. The node is called the *root* of the tree, and the subtrees it is connected to its *immediate subtrees*.

We code trees recursively as a sequence $\langle k, d_1, \ldots, d_k\rangle$, where $k$ is the number of immediate subtrees and $d_1, \ldots, d_k$ the codes of the immediate subtrees. If the nodes have labels, they can be included after the immediate subtrees. So a tree consisting just of a single node with label $l$ would be coded by $\langle 0, l\rangle$, and a tree consisting of a root (labelled $l_1$) connected to two single nodes (labelled $l_2$, $l_3$) would be coded by $\langle 2, \langle 0, l_2\rangle, \langle 0, l_3\rangle, l_1\rangle$.

**Proposition 25.25.** *The function* SubtreeSeq$(t)$*, which returns the code of a sequence the elements of which are the codes of all subtrees of the tree with code $t$, is primitive recursive.*

*Proof.* First note that ISubtrees$(t)$ = subseq$(t, 1, (t)_0)$ is primitive recursive and returns the codes of the immediate subtrees of a tree $t$. Now we can define a helper function hSubtreeSeq$(t, n)$ which computes the sequence of all subtrees which are $n$ nodes remove from the root. The sequence of subtrees of $t$ which is 0 nodes removed from the root—in other words, begins at the root of $t$—is the sequence consisting just of $t$. To obtain a sequence of all level $n+1$ subtrees of $t$, we concatenate the level $n$ subtrees with a sequence consisting

of all immediate subtrees of the level $n$ subtrees. To get a list of all these, note that if $f(x)$ is a primitive recursive function returning codes of sequences, then $g_f(s,k) = f((s)_0) \frown \ldots \frown f((s)_k)$ is also primivive recursive:

$$\begin{aligned} g(s,0) &= f((s)_0) \\ g(s,k+1) &= g(s,k) \frown f((s)_{k+1}) \end{aligned}$$

For instance, if $s$ is a sequence of trees, then $h(s) = g_{\text{ISubtrees}}(s, \text{len}(s))$ gives the sequence of the immediate subtrees of the elements of $s$. We can use it to define hSubtreeSeq by

$$\begin{aligned} \text{hSubtreeSeq}(t,0) &= \langle t \rangle \\ \text{hSubtreeSeq}(t,n+1) &= \text{hSubtreeSeq}(t,n) \frown h(\text{hSubtree}(t,n)). \end{aligned}$$

The maximum level of subtrees in a tree coded by $t$, i.e., the maximum distance between the root and a leaf node, is bounded by the code $t$. So a sequence of codes of all subtrees of the tree coded by $t$ is given by $\text{hSubtreeSeq}(t,t)$. □

## 25.13 Other Recursions

Using pairing and sequencing, we can justify more exotic (and useful) forms of primitive recursion. For example, it is often useful to define two functions simultaneously, such as in the following definition:

$$\begin{aligned} h_0(\vec{x},0) &= f_0(\vec{x}) \\ h_1(\vec{x},0) &= f_1(\vec{x}) \\ h_0(\vec{x},y+1) &= g_0(\vec{x},y,h_0(\vec{x},y),h_1(\vec{x},y)) \\ h_1(\vec{x},y+1) &= g_1(\vec{x},y,h_0(\vec{x},y),h_1(\vec{x},y)) \end{aligned}$$

This is an instance of *simultaneous recursion*. Another useful way of defining functions is to give the value of $h(\vec{x},y+1)$ in terms of *all* the values $h(\vec{x},0)$, $\ldots$, $h(\vec{x},y)$, as in the following definition:

$$\begin{aligned} h(\vec{x},0) &= f(\vec{x}) \\ h(\vec{x},y+1) &= g(\vec{x},y,\langle h(\vec{x},0),\ldots,h(\vec{x},y)\rangle). \end{aligned}$$

The following schema captures this idea more succinctly:

$$h(\vec{x},y) = g(\vec{x},y,\langle h(\vec{x},0),\ldots,h(\vec{x},y-1)\rangle)$$

with the understanding that the last argument to $g$ is just the empty sequence when $y$ is 0. In either formulation, the idea is that in computing the "successor step," the function $h$ can make use of the entire sequence of values computed

so far. This is known as a *course-of-values* recursion. For a particular example, it can be used to justify the following type of definition:

$$h(\vec{x}, y) = \begin{cases} g(\vec{x}, y, h(\vec{x}, k(\vec{x}, y))) & \text{if } k(\vec{x}, y) < y \\ f(\vec{x}) & \text{otherwise} \end{cases}$$

In other words, the value of $h$ at $y$ can be computed in terms of the value of $h$ at *any* previous value, given by $k$.

You should think about how to obtain these functions using ordinary primitive recursion. One final version of primitive recursion is more flexible in that one is allowed to change the *parameters* (side values) along the way:

$$\begin{aligned} h(\vec{x}, y) &= f(\vec{x}) \\ h(\vec{x}, y+1) &= g(\vec{x}, y, h(k(\vec{x}), y)) \end{aligned}$$

This, too, can be simulated with ordinary primitive recursion. (Doing so is tricky. For a hint, try unwinding the computation by hand.)

## 25.14 Non-Primitive Recursive Functions

The primitive recursive functions do not exhaust the intuitively computable functions. It should be intuitively clear that we can make a list of all the unary primitive recursive functions, $f_0$, $f_1$, $f_2$, ... such that we can effectively compute the value of $f_x$ on input $y$; in other words, the function $g(x, y)$, defined by

$$g(x, y) = f_x(y)$$

is computable. But then so is the function

$$\begin{aligned} h(x) &= g(x, x) + 1 \\ &= f_x(x) + 1. \end{aligned}$$

For each primitive recursive function $f_i$, the value of $h$ and $f_i$ differ at $i$. So $h$ is computable, but not primitive recursive; and one can say the same about $g$. This is an "effective" version of Cantor's diagonalization argument.

One can provide more explicit examples of computable functions that are not primitive recursive. For example, let the notation $g^n(x)$ denote $g(g(\ldots g(x)))$, with $n$ $g$'s in all; and define a sequence $g_0, g_1, \ldots$ of functions by

$$\begin{aligned} g_0(x) &= x + 1 \\ g_{n+1}(x) &= g_n^x(x) \end{aligned}$$

You can confirm that each function $g_n$ is primitive recursive. Each successive function grows much faster than the one before; $g_1(x)$ is equal to $2x$, $g_2(x)$ is equal to $2^x \cdot x$, and $g_3(x)$ grows roughly like an exponential stack of $x$ 2's.

Ackermann's function is essentially the function $G(x) = g_x(x)$, and one can show that this grows faster than any primitive recursive function.

Let us return to the issue of enumerating the primitive recursive functions. Remember that we have assigned symbolic notations to each primitive recursive function; so it suffices to enumerate notations. We can assign a natural number $\#(F)$ to each notation $F$, recursively, as follows:

$$\begin{aligned}
\#(0) &= \langle 0 \rangle \\
\#(S) &= \langle 1 \rangle \\
\#(P_i^n) &= \langle 2, n, i \rangle \\
\#(\mathrm{Comp}_{k,l}[H, G_0, \ldots, G_{k-1}]) &= \langle 3, k, l, \#(H), \#(G_0), \ldots, \#(G_{k-1}) \rangle \\
\#(\mathrm{Rec}_l[G, H]) &= \langle 4, l, \#(G), \#(H) \rangle
\end{aligned}$$

Here we are using the fact that every sequence of numbers can be viewed as a natural number, using the codes from the last section. The upshot is that every code is assigned a natural number. Of course, some sequences (and hence some numbers) do not correspond to notations; but we can let $f_i$ be the unary primitive recursive function with notation coded as $i$, if $i$ codes such a notation; and the constant 0 function otherwise. The net result is that we have an explicit way of enumerating the unary primitive recursive functions.

(In fact, some functions, like the constant zero function, will appear more than once on the list. This is not just an artifact of our coding, but also a result of the fact that the constant zero function has more than one notation. We will later see that one can not computably avoid these repetitions; for example, there is no computable function that decides whether or not a given notation represents the constant zero function.)

We can now take the function $g(x, y)$ to be given by $f_x(y)$, where $f_x$ refers to the enumeration we have just described. How do we know that $g(x, y)$ is computable? Intuitively, this is clear: to compute $g(x, y)$, first "unpack" $x$, and see if it is a notation for a unary function. If it is, compute the value of that function on input $y$.

You may already be convinced that (with some work!) one can write a program (say, in Java or C++) that does this; and now we can appeal to the Church-Turing thesis, which says that anything that, intuitively, is computable can be computed by a Turing machine.

Of course, a more direct way to show that $g(x, y)$ is computable is to describe a Turing machine that computes it, explicitly. This would, in particular, avoid the Church-Turing thesis and appeals to intuition. Soon we will have built up enough machinery to show that $g(x, y)$ is computable, appealing to a model of computation that can be *simulated* on a Turing machine: namely, the recursive functions.

## 25.15 Partial Recursive Functions

To motivate the definition of the recursive functions, note that our proof that there are computable functions that are not primitive recursive actually establishes much more. The argument was simple: all we used was the fact was that it is possible to enumerate functions $f_0, f_1, \ldots$ such that, as a function of $x$ and $y$, $f_x(y)$ is computable. So the argument applies to *any class of functions that can be enumerated in such a way*. This puts us in a bind: we would like to describe the computable functions explicitly; but any explicit description of a collection of computable functions cannot be exhaustive!

The way out is to allow *partial* functions to come into play. We will see that it *is* possible to enumerate the partial computable functions. In fact, we already pretty much know that this is the case, since it is possible to enumerate Turing machines in a systematic way. We will come back to our diagonal argument later, and explore why it does not go through when partial functions are included.

The question is now this: what do we need to add to the primitive recursive functions to obtain all the partial recursive functions? We need to do two things:

1. Modify our definition of the primitive recursive functions to allow for partial functions as well.
2. *Add* something to the definition, so that some new partial functions are included.

The first is easy. As before, we will start with zero, successor, and projections, and close under composition and primitive recursion. The only difference is that we have to modify the definitions of composition and primitive recursion to allow for the possibility that some of the terms in the definition are not defined. If $f$ and $g$ are partial functions, we will write $f(x) \downarrow$ to mean that $f$ is defined at $x$, i.e., $x$ is in the domain of $f$; and $f(x) \uparrow$ to mean the opposite, i.e., that $f$ is not defined at $x$. We will use $f(x) \simeq g(x)$ to mean that either $f(x)$ and $g(x)$ are both undefined, or they are both defined and equal. We will use these notations for more complicated terms as well. We will adopt the convention that if $h$ and $g_0, \ldots, g_k$ all are partial functions, then

$$h(g_0(\vec{x}), \ldots, g_k(\vec{x}))$$

is defined if and only if each $g_i$ is defined at $\vec{x}$, and $h$ is defined at $g_0(\vec{x})$, $\ldots$, $g_k(\vec{x})$. With this understanding, the definitions of composition and primitive recursion for partial functions is just as above, except that we have to replace "$=$" by "$\simeq$".

What we will add to the definition of the primitive recursive functions to obtain partial functions is the *unbounded search operator*. If $f(x, \vec{z})$ is any partial function on the natural numbers, define $\mu x\, f(x, \vec{z})$ to be

the least $x$ such that $f(0, \vec{z}), f(1, \vec{z}), \ldots, f(x, \vec{z})$ are all defined, and $f(x, \vec{z}) = 0$, if such an $x$ exists

with the understanding that $\mu x\ f(x, \vec{z})$ is undefined otherwise. This defines $\mu x\ f(x, \vec{z})$ uniquely.

Note that our definition makes no reference to Turing machines, or algorithms, or any specific computational model. But like composition and primitive recursion, there is an operational, computational intuition behind unbounded search. When it comes to the computability of a partial function, arguments where the function is undefined correspond to inputs for which the computation does not halt. The procedure for computing $\mu x\ f(x, \vec{z})$ will amount to this: compute $f(0, \vec{z}), f(1, \vec{z}), f(2, \vec{z})$ until a value of 0 is returned. If any of the intermediate computations do not halt, however, neither does the computation of $\mu x\ f(x, \vec{z})$.

If $R(x, \vec{z})$ is any relation, $\mu x\ R(x, \vec{z})$ is defined to be $\mu x\ (1 \dot{-} \chi_R(x, \vec{z}))$. In other words, $\mu x\ R(x, \vec{z})$ returns the least value of $x$ such that $R(x, \vec{z})$ holds. So, if $f(x, \vec{z})$ is a total function, $\mu x\ f(x, \vec{z})$ is the same as $\mu x\ (f(x, \vec{z}) = 0)$. But note that our original definition is more general, since it allows for the possibility that $f(x, \vec{z})$ is not everywhere defined (whereas, in contrast, the characteristic function of a relation is always total).

**Definition 25.26.** The set of *partial recursive functions* is the smallest set of partial functions from the natural numbers to the natural numbers (of various arities) containing zero, successor, and projections, and closed under composition, primitive recursion, and unbounded search.

Of course, some of the partial recursive functions will happen to be total, i.e., defined for every argument.

**Definition 25.27.** The set of *recursive functions* is the set of partial recursive functions that are total.

A recursive function is sometimes called "total recursive" to emphasize that it is defined everywhere.

## 25.16 The Normal Form Theorem

**Theorem 25.28** (Kleene's Normal Form Theorem)**.** *There is a primitive recursive relation $T(e, x, s)$ and a primitive recursive function $U(s)$, with the following property: if $f$ is any partial recursive function, then for some $e$,*

$$f(x) \simeq U(\mu s\ T(e, x, s))$$

*for every $x$.*

The proof of the normal form theorem is involved, but the basic idea is simple. Every partial recursive function has an *index* $e$, intuitively, a number coding its program or definition. If $f(x) \downarrow$, the computation can be recorded systematically and coded by some number $s$, and that $s$ codes the computation of $f$ on input $x$ can be checked primitive recursively using only $x$ and the definition $e$. This means that $T$ is primitive recursive. Given the full record of the computation $s$, the "upshot" of $s$ is the value of $f(x)$, and it can be obtained from $s$ primitive recursively as well.

The normal form theorem shows that only a single unbounded search is required for the definition of any partial recursive function. We can use the numbers $e$ as "names" of partial recursive functions, and write $\varphi_e$ for the function $f$ defined by the equation in the theorem. Note that any partial recursive function can have more than one index—in fact, every partial recursive function has infinitely many indices.

## 25.17 The Halting Problem

The *halting problem* in general is the problem of deciding, given the specification $e$ (e.g., program) of a computable function and a number $n$, whether the computation of the function on input $n$ halts, i.e., produces a result. Famously, Alan Turing proved that this problem itself cannot be solved by a computable function, i.e., the function

$$h(e, n) = \begin{cases} 1 & \text{if computation } e \text{ halts on input } n \\ 0 & \text{otherwise,} \end{cases}$$

is not computable.

In the context of partial recursive functions, the role of the specification of a program may be played by the index $e$ given in Kleene's normal form theorem. If $f$ is a partial recursive function, any $e$ for which the equation in the normal form theorem holds, is an index of $f$. Given a number $e$, the normal form theorem states that

$$\varphi_e(x) \simeq U(\mu s\ T(e, x, s))$$

is partial recursive, and for every partial recursive $f\colon \mathbb{N} \to \mathbb{N}$, there is an $e \in \mathbb{N}$ such that $\varphi_e(x) \simeq f(x)$ for all $x \in \mathbb{N}$. In fact, for each such $f$ there is not just one, but infinitely many such $e$. The *halting function* $h$ is defined by

$$h(e, x) = \begin{cases} 1 & \text{if } \varphi_e(x) \downarrow \\ 0 & \text{otherwise.} \end{cases}$$

Note that $h(e, x) = 0$ if $\varphi_e(x) \uparrow$, but also when $e$ is not the index of a partial recursive function at all.

**Theorem 25.29.** *The halting function h is not partial recursive.*

*Proof.* If $h$ were partial recursive, we could define

$$d(y) = \begin{cases} 1 & \text{if } h(y,y) = 0 \\ \mu x\ x \neq x & \text{otherwise.} \end{cases}$$

From this definition it follows that

1. $d(y) \downarrow$ iff $\varphi_y(y) \uparrow$ or $y$ is not the index of a partial recursive function.
2. $d(y) \uparrow$ iff $\varphi_y(y) \downarrow$.

If $h$ were partial recursive, then $d$ would be partial recursive as well. Thus, by the Kleene normal form theorem, it has an index $e_d$. Consider the value of $h(e_d, e_d)$. There are two possible cases, 0 and 1.

1. If $h(e_d, e_d) = 1$ then $\varphi_{e_d}(e_d) \downarrow$. But $\varphi_{e_d} \simeq d$, and $d(e_d)$ is defined iff $h(e_d, e_d) = 0$. So $h(e_d, e_d) \neq 1$.
2. If $h(e_d, e_d) = 0$ then either $e_d$ is not the index of a partial recursive function, or it is and $\varphi_{e_d}(e_d) \uparrow$. But again, $\varphi_{e_d} \simeq d$, and $d(e_d)$ is undefined iff $\varphi_{e_d}(e_d) \downarrow$.

The upshot is that $e_d$ cannot, after all, be the index of a partial recursive function. But if $h$ were partial recursive, $d$ would be too, and so our definition of $e_d$ as an index of it would be admissible. We must conclude that $h$ cannot be partial recursive. □

## 25.18 General Recursive Functions

There is another way to obtain a set of total functions. Say a total function $f(x, \vec{z})$ is *regular* if for every sequence of natural numbers $\vec{z}$, there is an $x$ such that $f(x, \vec{z}) = 0$. In other words, the regular functions are exactly those functions to which one can apply unbounded search, and end up with a total function. One can, conservatively, restrict unbounded search to regular functions:

**Definition 25.30.** The set of *general recursive functions* is the smallest set of functions from the natural numbers to the natural numbers (of various arities) containing zero, successor, and projections, and closed under composition, primitive recursion, and unbounded search applied to *regular* functions.

Clearly every general recursive function is total. The difference between Definition 25.30 and Definition 25.27 is that in the latter one is allowed to use partial recursive functions along the way; the only requirement is that the function you end up with at the end is total. So the word "general," a historic relic, is a misnomer; on the surface, Definition 25.30 is *less* general

than Definition 25.27. But, fortunately, the difference is illusory; though the definitions are different, the set of general recursive functions and the set of recursive functions are one and the same.

## Problems

**Problem 25.1.** Prove Proposition 25.5 by showing that the primitive recursive definition of mult is can be put into the form required by Definition 25.1 and showing that the corresponding functions $f$ and $g$ are primitive recursive.

**Problem 25.2.** Give the complete primitive recursive notation for mult.

**Problem 25.3.** Show that

$$f(x,y) = 2^{(2^{\cdot^{\cdot^{\cdot^{2^x}}}})} \Big\} y \text{ 2's}$$

is primitive recursive.

**Problem 25.4.** Show that integer division $d(x,y) = \lfloor x/y \rfloor$ (i.e., division, where you disregard everything after the decimal point) is primitive recursive. When $y = 0$, we stipulate $d(x,y) = 0$. Give an explicit definition of $d$ using primitive recursion and composition.

**Problem 25.5.** Suppose $R(\vec{x}, z)$ is primitive recursive. Define the function $m'_R(\vec{x}, y)$ which returns the least $z$ less than $y$ such that $R(\vec{x}, z)$ holds, if there is one, and 0 otherwise, by primitive recursion from $\chi_R$.

**Problem 25.6.** Define integer division $d(x,y)$ using bounded minimization.

**Problem 25.7.** Show that there is a primitive recursive function sconcat$(s)$ with the property that

$$\text{sconcat}(\langle s_0, \ldots, s_k \rangle) = s_0 \frown \ldots \frown s_k.$$

**Problem 25.8.** Show that there is a primitive recursive function tail$(s)$ with the property that

$$\begin{aligned} \text{tail}(\Lambda) &= 0 \text{ and} \\ \text{tail}(\langle s_0, \ldots, s_k \rangle) &= \langle s_1, \ldots, s_k \rangle. \end{aligned}$$

**Problem 25.9.** Prove Proposition 25.24.

**Problem 25.10.** The definition of hSubtreeSeq in the proof of Proposition 25.25 in general includes repetitions. Give an alternative definition which guarantees that the code of a subtree occurs only once in the resulting list.

# Chapter 26

# The Lambda Calculus

This chapter needs to be expanded (issue #66).

## 26.1 Introduction

The lambda calculus was originally designed by Alonzo Church in the early 1930s as a basis for constructive logic, and *not* as a model of the computable functions. But soon after the Turing computable functions, the recursive functions, and the general recursive functions were shown to be equivalent, lambda computability was added to the list. The fact that this initially came as a small surprise makes the characterization all the more interesting.

Lambda notation is a convenient way of referring to a function directly by a symbolic expression which defines it, instead of defining a name for it. Instead of saying "let $f$ be the function defined by $f(x) = x + 3$," one can say, "let $f$ be the function $\lambda x.\,(x + 3)$." In other words, $\lambda x.\,(x + 3)$ is just a *name* for the function that adds three to its argument. In this expression, $x$ is a dummy variable, or a placeholder: the same function can just as well be denoted by $\lambda y.\,(y + 3)$. The notation works even with other parameters around. For example, suppose $g(x, y)$ is a function of two variables, and $k$ is a natural number. Then $\lambda x.\,g(x, k)$ is the function which maps any $x$ to $g(x, k)$.

This way of defining a function from a symbolic expression is known as *lambda abstraction*. The flip side of lambda abstraction is *application*: assuming one has a function $f$ (say, defined on the natural numbers), one can *apply* it to any value, like 2. In conventional notation, of course, we write $f(2)$ for the result.

What happens when you combine lambda abstraction with application? Then the resulting expression can be simplified, by "plugging" the applicand in for the abstracted variable. For example,

$$(\lambda x.\,(x + 3))(2)$$

can be simplified to $2 + 3$.

Up to this point, we have done nothing but introduce new notations for conventional notions. The lambda calculus, however, represents a more radical departure from the set-theoretic viewpoint. In this framework:

1. Everything denotes a function.
2. Functions can be defined using lambda abstraction.
3. Anything can be applied to anything else.

For example, if $F$ is a term in the lambda calculus, $F(F)$ is always assumed to be meaningful. This liberal framework is known as the *untyped* lambda calculus, where "untyped" means "no restriction on what can be applied to what."

There is also a *typed* lambda calculus, which is an important variation on the untyped version. Although in many ways the typed lambda calculus is similar to the untyped one, it is much easier to reconcile with a classical set-theoretic framework, and has some very different properties.

Research on the lambda calculus has proved to be central in theoretical computer science, and in the design of programming languages. LISP, designed by John McCarthy in the 1950s, is an early example of a language that was influenced by these ideas.

## 26.2 The Syntax of the Lambda Calculus

One starts with a sequence of variables $x$, $y$, $z$, ... and some constant symbols $a$, $b$, $c$, .... The set of terms is defined inductively, as follows:

1. Each variable is a term.
2. Each constant is a term.
3. If $M$ and $N$ are terms, so is $(MN)$.
4. If $M$ is a term and $x$ is a variable, then $(\lambda x.\, M)$ is a term.

The system without any constants at all is called the *pure* lambda calculus.

We will follow a few notational conventions:

1. When parentheses are left out, application takes place from left to right. For example, if $M$, $N$, $P$, and $Q$ are terms, then $MNPQ$ abbreviates $(((MN)P)Q)$.
2. Again, when parentheses are left out, lambda abstraction is to be given the widest scope possible. From example, $\lambda x.\, MNP$ is read $\lambda x.\, (MNP)$.

3. A lambda can be used to abstract multiple variables. For example, $\lambda xyz.\, M$ is short for $\lambda x.\, \lambda y.\, \lambda z.\, M$.

For example,

$$\lambda xy.\, xxyx\lambda z.\, xz$$

abbreviates

$$\lambda x.\, \lambda y.\, ((((xx)y)x)\lambda z.\, (xz)).$$

You should memorize these conventions. They will drive you crazy at first, but you will get used to them, and after a while they will drive you less crazy than having to deal with a morass of parentheses.

Two terms that differ only in the names of the bound variables are called $\alpha$-equivalent; for example, $\lambda x.\, x$ and $\lambda y.\, y$. It will be convenient to think of these as being the "same" term; in other words, when we say that $M$ and $N$ are the same, we also mean "up to renamings of the bound variables." Variables that are in the scope of a $\lambda$ are called "bound", while others are called "free." There are no free variables in the previous example; but in

$$(\lambda z.\, yz)x$$

$y$ and $x$ are free, and $z$ is bound.

## 26.3 Reduction of Lambda Terms

What can one do with lambda terms? Simplify them. If $M$ and $N$ are any lambda terms and $x$ is any variable, we can use $M[N/x]$ to denote the result of substituting $N$ for $x$ in $M$, after renaming any bound variables of $M$ that would interfere with the free variables of $N$ after the substitution. For example,

$$(\lambda w.\, xxw)[yyz/x] = \lambda w.\, (yyz)(yyz)w.$$

Alternative notations for substitution are $[N/x]M$, $M[N/x]$, and also $M[x/N]$. Beware!

Intuitively, $(\lambda x.\, M)N$ and $M[N/x]$ have the same meaning; the act of replacing the first term by the second is called $\beta$-conversion. More generally, if it is possible convert a term $P$ to $P'$ by $\beta$-conversion of some subterm, one says *$P$ $\beta$-reduces to $P'$ in one step*. If $P$ can be converted to $P'$ with any number of one-step reductions (possibly none), then *$P$ $\beta$-reduces* to $P'$. A term that cannot be $\beta$-reduced any further is called *$\beta$-irreducible*, or $\beta$-normal. I will say "reduces" instead of "$\beta$-reduces," etc., when the context is clear.

Let us consider some examples.

1. We have

$$\begin{aligned}(\lambda x.\, xxy)\lambda z.\, z &\triangleright_1 (\lambda z.\, z)(\lambda z.\, z)y \\ &\triangleright_1 (\lambda z.\, z)y \\ &\triangleright_1 y\end{aligned}$$

2. "Simplifying" a term can make it more complex:

$$\begin{aligned}(\lambda x.\, xxy)(\lambda x.\, xxy) &\triangleright_1 (\lambda x.\, xxy)(\lambda x.\, xxy)y \\ &\triangleright_1 (\lambda x.\, xxy)(\lambda x.\, xxy)yy \\ &\triangleright_1 \ldots\end{aligned}$$

3. It can also leave a term unchanged:

$$(\lambda x.\, xx)(\lambda x.\, xx) \triangleright_1 (\lambda x.\, xx)(\lambda x.\, xx)$$

4. Also, some terms can be reduced in more than one way; for example,

$$(\lambda x.\, (\lambda y.\, yx)z)v \triangleright_1 (\lambda y.\, yv)z$$

   by contracting the outermost application; and

$$(\lambda x.\, (\lambda y.\, yx)z)v \triangleright_1 (\lambda x.\, zx)v$$

   by contracting the innermost one. Note, in this case, however, that both terms further reduce to the same term, $zv$.

The final outcome in the last example is not a coincidence, but rather illustrates a deep and important property of the lambda calculus, known as the "Church-Rosser property."

## 26.4 The Church-Rosser Property

**Theorem 26.1.** *Let $M$, $N_1$, and $N_2$ be terms, such that $M \triangleright N_1$ and $M \triangleright N_2$. Then there is a term $P$ such that $N_1 \triangleright P$ and $N_2 \triangleright P$.*

**Corollary 26.2.** *Suppose $M$ can be reduced to normal form. Then this normal form is unique.*

*Proof.* If $M \triangleright N_1$ and $M \triangleright N_2$, by the previous theorem there is a term $P$ such that $N_1$ and $N_2$ both reduce to $P$. If $N_1$ and $N_2$ are both in normal form, this can only happen if $N_1 = P = N_2$. □

Finally, we will say that two terms $M$ and $N$ are *β-equivalent*, or just *equivalent*, if they reduce to a common term; in other words, if there is some $P$ such that $M \triangleright P$ and $N \triangleright P$. This is written $M \equiv N$. Using Theorem 26.1, you can check that $\equiv$ is an equivalence relation, with the additional property that for every $M$ and $N$, if $M \triangleright N$ or $N \triangleright M$, then $M \equiv N$. (In fact, one can show that $\equiv$ is the *smallest* equivalence relation having this property.)

## 26.5 Representability by Lambda Terms

How can the lambda calculus serve as a model of computation? At first, it is not even clear how to make sense of this statement. To talk about computability on the natural numbers, we need to find a suitable representation for such numbers. Here is one that works surprisingly well.

**Definition 26.3.** For each natural number $n$, define the *numeral* $\overline{n}$ to be the lambda term $\lambda x.\, \lambda y.\, (x(x(x(\ldots x(y)))))$, where there are $n$ $x$'s in all.

The terms $\overline{n}$ are "iterators": on input $f$, $\overline{n}$ returns the function mapping $y$ to $f^n(y)$. Note that each numeral is normal. We can now say what it means for a lambda term to "compute" a function on the natural numbers.

**Definition 26.4.** Let $f(x_0, \ldots, x_{n-1})$ be an $n$-ary partial function from $\mathbb{N}$ to $\mathbb{N}$. We say a lambda term $X$ *represents* $f$ if for every sequence of natural numbers $m_0, \ldots, m_{n-1}$,

$$X\overline{m_0}\overline{m_1}\ldots\overline{m_{n-1}} \triangleright \overline{f(m_0, m_1, \ldots, m_{n-1})}$$

if $f(m_0, \ldots, m_{n-1})$ is defined, and $X\overline{m_0}\overline{m_1}\ldots\overline{m_{n-1}}$ has no normal form otherwise.

**Theorem 26.5.** *A function $f$ is a partial computable function if and only if it is represented by a lambda term.*

This theorem is somewhat striking. As a model of computation, the lambda calculus is a rather simple calculus; the only operations are lambda abstraction and application! From these meager resources, however, it is possible to implement any computational procedure.

## 26.6 Lambda Representable Functions are Computable

**Theorem 26.6.** *If a partial function $f$ is represented by a lambda term, it is computable.*

*Proof.* Suppose a function $f$, is represented by a lambda term $X$. Let us describe an informal procedure to compute $f$. On input $m_0$, …, $m_{n-1}$, write down the term $X\overline{m}_0 \ldots \overline{m}_{n-1}$. Build a tree, first writing down all the one-step reductions of the original term; below that, write all the one-step reductions of those (i.e., the two-step reductions of the original term); and keep going. If you ever reach a numeral, return that as the answer; otherwise, the function is undefined.

An appeal to Church's thesis tells us that this function is computable. A better way to prove the theorem would be to give a recursive description of this search procedure. For example, one could define a sequence primitive recursive functions and relations, "IsASubterm," "Substitute," "ReducesToInOneStep,"

"ReductionSequence," "Numeral," etc. The partial recursive procedure for computing $f(m_0, \ldots, m_{n-1})$ is then to search for a sequence of one-step reductions starting with $X\overline{m_0}\ldots\overline{m_{n-1}}$ and ending with a numeral, and return the number corresponding to that numeral. The details are long and tedious but otherwise routine. □

## 26.7 Computable Functions are Lambda Representable

**Theorem 26.7.** *Every computable partial function if representable by a lambda term.*

*Proof.* Wwe need to show that every partial computable function $f$ is represented by a lambda term $\overline{f}$. By Kleene's normal form theorem, it suffices to show that every primitive recursive function is represented by a lambda term, and then that the functions so represented are closed under suitable compositions and unbounded search. To show that every primitive recursive function is represented by a lambda term, it suffices to show that the initial functions are represented, and that the partial functions that are represented by lambda terms are closed under composition, primitive recursion, and unbounded search. □

We will use a more conventional notation to make the rest of the proof more readable. For example, we will write $M(x, y, z)$ instead of $Mxyz$. While this is suggestive, you should remember that terms in the untyped lambda calculus do not have associated arities; so, for the same term $M$, it makes just as much sense to write $M(x, y)$ and $M(x, y, z, w)$. But using this notation indicates that we are treating $M$ as a function of three variables, and helps make the intentions behind the definitions clearer. In a similar way, we will say "define $M$ by $M(x, y, z) = \ldots$" instead of "define $M$ by $M = \lambda x.\, \lambda y.\, \lambda z.\, \ldots$."

## 26.8 The Basic Primitive Recursive Functions are Lambda Representable

**Lemma 26.8.** *The functions* 0*, S, and* $P_i^n$ *are lambda representable.*

*Proof.* Zero, $\overline{0}$, is just $\lambda x.\, \lambda y.\, y$.

The successor function $\overline{S}$, is defined by $\overline{S}(u) = \lambda x.\, \lambda y.\, x(uxy)$. You should think about why this works; for each numeral $\overline{n}$, thought of as an iterator, and each function $f$, $S(\overline{n}, f)$ is a function that, on input $y$, applies $f$ $n$ times starting with $y$, and then applies it once more.

There is nothing to say about projections: $\overline{P_i^n}(x_0, \ldots, x_{n-1}) = x_i$. In other words, by our conventions, $\overline{P_i^n}$ is the lambda term $\lambda x_0.\, \ldots \lambda x_{n-1}.\, x_i$. □

## 26.9 Lambda Representable Functions Closed under Composition

**Lemma 26.9.** *The lambda representable functions are closed under composition.*

*Proof.* Suppose $f$ is defined by composition from $h, g_0, \ldots, g_{k-1}$. Assuming $h$, $g_0, \ldots, g_{k-1}$ are represented by $\overline{h}$, $\overline{g_0}, \ldots, \overline{g_{k-1}}$, respectively, we need to find a term $\overline{f}$ representing $f$. But we can simply define $\overline{f}$ by

$$\overline{f}(x_0, \ldots, x_{l-1}) = \overline{h}(\overline{g_0}(x_0, \ldots, x_{l-1}), \ldots, \overline{g_{k-1}}(x_0, \ldots, x_{l-1})).$$

In other words, the language of the lambda calculus is well suited to represent composition. □

## 26.10 Lambda Representable Functions Closed under Primitive Recursion

When it comes to primitive recursion, we finally need to do some work. We will have to proceed in stages. As before, on the assumption that we already have terms $\overline{g}$ and $\overline{h}$ representing functions $g$ and $h$, respectively, we want a term $\overline{f}$ representing the function $f$ defined by

$$\begin{aligned} f(0, \vec{z}) &= g(\vec{z}) \\ f(x+1, \vec{z}) &= h(z, f(x, \vec{z}), \vec{z}). \end{aligned}$$

So, in general, given lambda terms $G'$ and $H'$, it suffices to find a term $F$ such that

$$\begin{aligned} F(\overline{0}, \vec{z}) &\equiv G'(\vec{z}) \\ F(\overline{n+1}, \vec{z}) &\equiv H'(\overline{n}, F(\overline{n}, \vec{z}), \vec{z}) \end{aligned}$$

for every natural number $n$; the fact that $G'$ and $H'$ represent $g$ and $h$ means that whenever we plug in numerals $\overline{m}$ for $\vec{z}$, $F(\overline{n+1}, \overline{m})$ will normalize to the right answer.

But for this, it suffices to find a term $F$ satisfying

$$\begin{aligned} F(\overline{0}) &\equiv G \\ F(\overline{n+1}) &\equiv H(\overline{n}, F(\overline{n})) \end{aligned}$$

for every natural number $n$, where

$$\begin{aligned} G &= \lambda \vec{z}.\, G'(\vec{z}) \text{ and} \\ H(u, v) &= \lambda \vec{z}.\, H'(u, v(u, \vec{z}), \vec{z}). \end{aligned}$$

In other words, with lambda trickery, we can avoid having to worry about the extra parameters $\vec{z}$—they just get absorbed in the lambda notation.

Before we define the term $F$, we need a mechanism for handling ordered pairs. This is provided by the next lemma.

**Lemma 26.10.** *There is a lambda term $D$ such that for each pair of lambda terms $M$ and $N$, $D(M,N)(\overline{0}) \triangleright M$ and $D(M,N)(\overline{1}) \triangleright N$.*

*Proof.* First, define the lambda term $K$ by

$$K(y) = \lambda x.\, y.$$

In other words, $K$ is the term $\lambda y.\, \lambda x.\, y$. Looking at it differently, for every $M$, $K(M)$ is a constant function that returns $M$ on any input.

Now define $D(x,y,z)$ by $D(x,y,z) = z(K(y))x$. Then we have

$$D(M,N,\overline{0}) \triangleright \overline{0}(K(N))M \triangleright M \text{ and}$$
$$D(M,N,\overline{1}) \triangleright \overline{1}(K(N))M \triangleright K(N)M \triangleright N,$$

as required. □

The idea is that $D(M,N)$ represents the pair $\langle M,N\rangle$, and if $P$ is assumed to represent such a pair, $P(\overline{0})$ and $P(\overline{1})$ represent the left and right projections, $(P)_0$ and $(P)_1$. We will use the latter notations.

**Lemma 26.11.** *The lambda representable functions are closed under primitive recursion.*

*Proof.* We need to show that given any terms, $G$ and $H$, we can find a term $F$ such that

$$F(\overline{0}) \equiv G$$
$$F(\overline{n+1}) \equiv H(\overline{n}, F(\overline{n}))$$

for every natural number $n$. The idea is roughly to compute sequences of *pairs*

$$\langle \overline{0}, F(\overline{0})\rangle, \langle \overline{1}, F(\overline{1})\rangle, \ldots,$$

using numerals as iterators. Notice that the first pair is just $\langle \overline{0}, G\rangle$. Given a pair $\langle \overline{n}, F(\overline{n})\rangle$, the next pair, $\langle \overline{n+1}, F(\overline{n+1})\rangle$ is supposed to be equivalent to $\langle \overline{n+1}, H(\overline{n}, F(\overline{n}))\rangle$. We will design a lambda term $T$ that makes this one-step transition.

The details are as follows. Define $T(u)$ by

$$T(u) = \langle S((u)_0), H((u)_0, (u)_1)\rangle.$$

Now it is easy to verify that for any number $n$,

$$T(\langle \overline{n}, M\rangle) \triangleright \langle \overline{n+1}, H(\overline{n}, M)\rangle.$$

As suggested above, given $G$ and $H$, define $F(u)$ by

$$F(u) = (u(T, \langle \overline{0}, G \rangle))_1.$$

In other words, on input $\overline{n}$, $F$ iterates $T$ $n$ times on $\langle \overline{0}, G \rangle$, and then returns the second component. To start with, we have

1. $\overline{0}(T, \langle \overline{0}, G \rangle) \equiv \langle \overline{0}, G \rangle$
2. $F(\overline{0}) \equiv G$

By induction on $n$, we can show that for each natural number one has the following:

1. $\overline{n+1}(T, \langle \overline{0}, G \rangle) \equiv \langle \overline{n+1}, F(\overline{n+1}) \rangle$
2. $F(\overline{n+1}) \equiv H(\overline{n}, F(\overline{n}))$

For the second clause, we have

$$\begin{aligned} F(\overline{n+1}) &\triangleright (\overline{n+1}(T, \langle \overline{0}, G \rangle))_1 \\ &\equiv (T(\overline{n}(T, \langle \overline{0}, G \rangle)))_1 \\ &\equiv (T(\langle \overline{n}, F(\overline{n}) \rangle))_1 \\ &\equiv (\langle \overline{n+1}, H(\overline{n}, F(\overline{n})) \rangle)_1 \\ &\equiv H(\overline{n}, F(\overline{n})). \end{aligned}$$

Here we have used the induction hypothesis on the second-to-last line. For the first clause, we have

$$\begin{aligned} \overline{n+1}(T, \langle \overline{0}, G \rangle) &\equiv T(\overline{n}(T, \langle \overline{0}, G \rangle)) \\ &\equiv T(\langle \overline{n}, F(\overline{n}) \rangle) \\ &\equiv \langle \overline{n+1}, H(\overline{n}, F(\overline{n})) \rangle \\ &\equiv \langle \overline{n+1}, F(\overline{n+1}) \rangle. \end{aligned}$$

Here we have used the second clause in the last line. So we have shown $F(\overline{0}) \equiv G$ and, for every $n$, $F(\overline{n+1}) \equiv H(\overline{n}, F(\overline{n}))$, which is exactly what we needed. □

## 26.11 Fixed-Point Combinators

Suppose you have a lambda term $g$, and you want another term $k$ with the property that $k$ is $\beta$-equivalent to $gk$. Define terms

$$\mathrm{diag}(x) = xx$$

and

$$l(x) = g(\mathrm{diag}(x))$$

using our notational conventions; in other words, $l$ is the term $\lambda x.\, g(xx)$. Let $k$ be the term $ll$. Then we have

$$\begin{aligned} k &= (\lambda x.\, g(xx))(\lambda x.\, g(xx)) \\ &\triangleright g((\lambda x.\, g(xx))(\lambda x.\, g(xx))) \\ &= gk. \end{aligned}$$

If one takes

$$Y = \lambda g.\, ((\lambda x.\, g(xx))(\lambda x.\, g(xx)))$$

then $Yg$ and $g(Yg)$ reduce to a common term; so $Yg \equiv_\beta g(Yg)$. This is known as "Curry's combinator." If instead one takes

$$Y = (\lambda xg.\, g(xxg))(\lambda xg.\, g(xxg))$$

then in fact $Yg$ reduces to $g(Yg)$, which is a stronger statement. This latter version of $Y$ is known as "Turing's combinator."

## 26.12 Lambda Representable Functions Closed under Minimization

**Lemma 26.12.** *Suppose $f(x, y)$ is primitive recursive. Let $g$ be defined by*

$$g(x) \simeq \mu y\, f(x, y).$$

*Then $g$ is represented by a lambda term.*

*Proof.* The idea is roughly as follows. Given $x$, we will use the fixed-point lambda term $Y$ to define a function $h_x(n)$ which searches for a $y$ starting at $n$; then $g(x)$ is just $h_x(0)$. The function $h_x$ can be expressed as the solution of a fixed-point equation:

$$h_x(n) \simeq \begin{cases} n & \text{if } f(x, n) = 0 \\ h_x(n+1) & \text{otherwise.} \end{cases}$$

Here are the details. Since $f$ is primitive recursive, it is represented by some term $F$. Remember that we also have a lambda term $D$, such that $D(M, N, \bar{0}) \triangleright M$ and $D(M, N, \bar{1}) \triangleright N$. Fixing $x$ for the moment, to represent $h_x$ we want to find a term $H$ (depending on $x$) satisfying

$$H(\overline{n}) \equiv D(\overline{n}, H(S(\overline{n})), F(x, \overline{n})).$$

We can do this using the fixed-point term $Y$. First, let $U$ be the term

$$\lambda h.\, \lambda z.\, D(z, (h(Sz)), F(x, z)),$$

and then let $H$ be the term $YU$. Notice that the only free variable in $H$ is $x$. Let us show that $H$ satisfies the equation above.

By the definition of $Y$, we have

$$H = YU \equiv U(YU) = U(H).$$

In particular, for each natural number $n$, we have

$$\begin{aligned} H(\overline{n}) &\equiv U(H, \overline{n}) \\ &\quad \triangleright D(\overline{n}, H(S(\overline{n})), F(x, \overline{n})), \end{aligned}$$

as required. Notice that if you substitute a numeral $\overline{m}$ for $x$ in the last line, the expression reduces to $\overline{n}$ if $F(\overline{m}, \overline{n})$ reduces to $\overline{0}$, and it reduces to $H(S(\overline{n}))$ if $F(\overline{m}, \overline{n})$ reduces to any other numeral.

To finish off the proof, let $G$ be $\lambda x. H(\overline{0})$. Then $G$ represents $g$; in other words, for every $m$, $G(\overline{m})$ reduces to reduces to $\overline{g(m)}$, if $g(m)$ is defined, and has no normal form otherwise. □

# Chapter 27

# Computability Theory

Material in this chapter should be reviewed and expanded. In paticular, there are no exercises yet.

## 27.1 Introduction

The branch of logic known as *Computability Theory* deals with issues having to do with the computability, or relative computability, of functions and sets. It is a evidence of Kleene's influence that the subject used to be known as *Recursion Theory*, and today, both names are commonly used.

Let us call a function $f\colon \mathbb{N} \nrightarrow \mathbb{N}$ *partial computable* if it can be computed in some model of computation. If $f$ is total we will simply say that $f$ is *computable*. A relation $R$ with computable characteristic function $\chi_R$ is also called computable. If $f$ and $g$ are partial functions, we will write $f(x)\downarrow$ to mean that $f$ is defined at $x$, i.e., $x$ is in the domain of $f$; and $f(x)\uparrow$ to mean the opposite, i.e., that $f$ is not defined at $x$. We will use $f(x) \simeq g(x)$ to mean that either $f(x)$ and $g(x)$ are both undefined, or they are both defined and equal.

One can explore the subject without having to refer to a specific model of computation. To do this, one shows that there is a universal partial computable function, $\mathrm{Un}(k, x)$. This allows us to enumerate the partial computable functions. We will adopt the notation $\varphi_k$ to denote the $k$-th unary partial computable function, defined by $\varphi_k(x) \simeq \mathrm{Un}(k, x)$. (Kleene used $\{k\}$ for this purpose, but this notation has not been used as much recently.) Slightly more generally, we can uniformly enumerate the partial computable functions of arbitrary arities, and we will use $\varphi^n_k$ to denote the $k$-th $n$-ary partial recursive function.

Recall that if $f(\vec{x}, y)$ is a total or partial function, then $\mu y\ f(\vec{x}, y)$ is the function of $\vec{x}$ that returns the least $y$ such that $f(\vec{x}, y) = 0$, assuming that all of $f(\vec{x}, 0), \ldots, f(\vec{x}, y-1)$ are defined; if there is no such $y$, $\mu y\ f(\vec{x}, y)$ is undefined.

If $R(\vec{x}, y)$ is a relation, $\mu y\ R(\vec{x}, y)$ is defined to be the least $y$ such that $R(\vec{x}, y)$ is true; in other words, the least $y$ such that *one minus* the characteristic function of $R$ is equal to zero at $\vec{x}, y$.

To show that a function is computable, there are two ways one can proceed:

1. Rigorously: describe a Turing machine or partial recursive function explicitly, and show that it computes the function you have in mind;
2. Informally: describe an algorithm that computes it, and appeal to Church's thesis.

There is no fine line between the two; a detailed description of an algorithm should provide enough information so that it is relatively clear how one could, in principle, design the right Turing machine or sequence of partial recursive definitions. Fully rigorous definitions are unlikely to be informative, and we will try to find a happy medium between these two approaches; in short, we will try to find intuitive yet rigorous proofs that the precise definitions could be obtained.

## 27.2 Coding Computations

In every model of computation, it is possible to do the following:

1. Describe the *definitions* of computable functions in a systematic way. For instance, you can think of Turing machine specifications, recursive definitions, or programs in a programming language as providing these definitions.
2. Describe the complete record of the computation of a function given by some definition for a given input. For instance, a Turing machine computation can be described by the sequence of configurations (state of the machine, contents of the tape) for each step of computation.
3. Test whether a putative record of a computation is in fact the record of how a computable function with a given definition would be computed for a given input.
4. Extract from such a description of the complete record of a computation the value of the function for a given input. For instance, the contents of the tape in the very last step of a halting Turing machine computation is the value.

Using coding, it is possible to assign to each description of a computable function a numerical *index* in such a way that the instructions can be recovered from the index in a computable way. Similarly, the complete record of a computation can be coded by a single number as well. The resulting arithmetical

relation "$s$ codes the record of computation of the function with index $e$ for input $x$" and the function "output of computation sequence with code $s$" are then computable; in fact, they are primitive recursive.

This fundamental fact is very powerful, and allows us to prove a number of striking and important results about computability, independently of the model of computation chosen.

## 27.3 The Normal Form Theorem

**Theorem 27.1** (Kleene's Normal Form Theorem)**.** *There are a primitive recursive relation $T(k, x, s)$ and a primitive recursive function $U(s)$, with the following property: if $f$ is any partial computable function, then for some $k$,*

$$f(x) \simeq U(\mu s\ T(k, x, s))$$

*for every $x$.*

*Proof Sketch.* For any model of computation one can rigorously define a description of the computable function $f$ and code such description using a natural number $k$. One can also rigorously define a notion of "computation sequence" which records the process of computing the function with index $k$ for input $x$. These computation sequences can likewise be coded as numbers $s$. This can be done in such a way that (a) it is decidable whether a number $s$ codes the computation sequence of the function with index $k$ on input $x$ and (b) what the end result of the computation sequence coded by $s$ is. In fact, the relation in (a) and the function in (b) are primitive recursive. □

In order to give a rigorous proof of the Normal Form Theorem, we would have to fix a model of computation and carry out the coding of descriptions of computable functions and of computation sequences in detail, and verify that the relation $T$ and function $U$ are primitive recursive. For most applications, it suffices that $T$ and $U$ are computable and that $U$ is total.

It is probably best to remember the proof of the normal form theorem in slogan form: $\mu s\ T(k, x, s)$ searches for a computation sequence of the function with index $k$ on input $x$, and $U$ returns the output of the computation sequence if one can be found.

$T$ and $U$ can be used to define the enumeration $\varphi_0$, $\varphi_1$, $\varphi_2$, .... From now on, we will assume that we have fixed a suitable choice of $T$ and $U$, and take the equation

$$\varphi_e(x) \simeq U(\mu s\ T(e, x, s))$$

to be the *definition* of $\varphi_e$.

Here is another useful fact:

**Theorem 27.2.** *Every partial computable function has infinitely many indices.*

Again, this is intuitively clear. Given any (description of) a computable function, one can come up with a different description which computes the same function (input-output pair) but does so, e.g., by first doing something that has no effect on the computation (say, test if $0 = 0$, or count to 5, etc.). The index of the altered description will always be different from the original index. Both are indices of the same function, just computed slightly differently.

## 27.4 The *s-m-n* Theorem

The next theorem is known as the "*s-m-n* theorem," for a reason that will be clear in a moment. The hard part is understanding just what the theorem says; once you understand the statement, it will seem fairly obvious.

**Theorem 27.3.** *For each pair of natural numbers $n$ and $m$, there is a primitive recursive function $s^m_n$ such that for every sequence $x$, $a_0$, ..., $a_{m-1}$, $y_0$ ,..., $y_{n-1}$, we have*

$$\varphi^n_{s^m_n(x,a_0,\ldots,a_{m-1})}(y_0,\ldots,y_{n-1}) \simeq \varphi^{m+n}_x(a_0,\ldots,a_{m-1},y_0,\ldots,y_{n-1}).$$

It is helpful to think of $s^m_n$ as acting on *programs*. That is, $s^m_n$ takes a program, $x$, for an $(m+n)$-ary function, as well as fixed inputs $a_0, \ldots, a_{m-1}$; and it returns a program, $s^m_n(x, a_0, \ldots, a_{m-1})$, for the $n$-ary function of the remaining arguments. It you think of $x$ as the description of a Turing machine, then $s^m_n(x, a_0, \ldots, a_{m-1})$ is the Turing machine that, on input $y_0, \ldots, y_{n-1}$, prepends $a_0, \ldots, a_{m-1}$ to the input string, and runs $x$. Each $s^m_n$ is then just a primitive recursive function that finds a code for the appropriate Turing machine.

## 27.5 The Universal Partial Computable Function

**Theorem 27.4.** *There is a universal partial computable function* $\mathrm{Un}(k, x)$*. In other words, there is a function* $\mathrm{Un}(k, x)$ *such that:*

1. $\mathrm{Un}(k, x)$ *is partial computable.*
2. *If $f(x)$ is any partial computable function, then there is a natural number $k$ such that $f(x) \simeq \mathrm{Un}(k, x)$ for every $x$.*

*Proof.* Let $\mathrm{Un}(k, x) \simeq U(\mu s\ T(k, x, s))$ in Kleene's normal form theorem. □

This is just a precise way of saying that we have an effective enumeration of the partial computable functions; the idea is that if we write $f_k$ for the function defined by $f_k(x) = \mathrm{Un}(k, x)$, then the sequence $f_0$, $f_1$, $f_2$, ... includes all the partial computable functions, with the property that $f_k(x)$ can be computed "uniformly" in $k$ and $x$. For simplicity, we am using a binary function that is universal for unary functions, but by coding sequences of numbers we can easily generalize this to more arguments. For example, note that

if $f(x, y, z)$ is a 3-place partial recursive function, then the function $g(x) \simeq f((x)_0, (x)_1, (x)_2)$ is a unary recursive function.

## 27.6 No Universal Computable Function

**Theorem 27.5.** *There is no universal computable function. In other words, the universal function* $\mathrm{Un}'(k, x) = \varphi_k(x)$ *is not computable.*

*Proof.* This theorem says that there is no *total* computable function that is universal for the total computable functions. The proof is a simple diagonalization: if $\mathrm{Un}'(k, x)$ were total and computable, then

$$d(x) = \mathrm{Un}'(x, x) + 1$$

would also be total and computable. However, for every $k$, $d(k)$ is not equal to $\mathrm{Un}'(k, k)$. □

Theorem Theorem 27.4 above shows that we can get around this diagonalization argument, but only at the expense of allowing partial functions. It is worth trying to understand what goes wrong with the diagonalization argument, when we try to apply it in the partial case. In particular, the function $h(x) = \mathrm{Un}(x, x) + 1$ *is* partial recursive. Suppose $h$ is the $k$-th function in the enumeration; what can we say about $h(k)$?

## 27.7 The Halting Problem

Since, in our construction, $\mathrm{Un}(k, x)$ is defined if and only if the computation of the function coded by $k$ produces a value for input $x$, it is natural to ask if we can decide whether this is the case. And in fact, it is not. For the Turing machine model of computation, this means that whether a given Turing machine halts on a given input is computationally undecidable. The following theorem is therefore known as the "undecidability of the halting problem." I will provide two proofs below. The first continues the thread of our previous discussion, while the second is more direct.

**Theorem 27.6.** *Let*

$$h(k, x) = \begin{cases} 1 & \textit{if } \mathrm{Un}(k, x) \textit{ is defined} \\ 0 & \textit{otherwise.} \end{cases}$$

*Then $h$ is not computable.*

*Proof.* If $h$ were computable, we would have a universal computable function, as follows. Suppose $h$ is computable, and define

$$\mathrm{Un}'(k, x) = \begin{cases} fnUn(k, x) & \text{if } h(k, x) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

But now $\mathrm{Un}'(k, x)$ is a total function, and is computable if $h$ is. For instance, we could define $g$ using primitive recursion, by

$$g(0, k, x) \simeq 0$$
$$g(y + 1, k, x) \simeq \mathrm{Un}(k, x);$$

then

$$\mathrm{Un}'(k, x) \simeq g(h(k, x), k, x).$$

And since $\mathrm{Un}'(k, x)$ agrees with $\mathrm{Un}(k, x)$ wherever the latter is defined, $\mathrm{Un}'$ is universal for those partial computable functions that happen to be total. But this contradicts Theorem 27.5. □

*Proof.* Suppose $h(k, x)$ were computable. Define the function $g$ by

$$g(x) = \begin{cases} 0 & \text{if } h(x, x) = 0 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

The function $g$ is partial computable; for example, one can define it as $\mu y\, h(x, x) = 0$. So, for some $k$, $g(x) \simeq \mathrm{Un}(k, x)$ for every $x$. Is $g$ defined at $k$? If it is, then, by the definition of $g$, $h(k, k) = 0$. By the definition of $f$, this means that $\mathrm{Un}(k, k)$ is undefined; but by our assumption that $g(k) \simeq \mathrm{Un}(k, x)$ for every $x$, this means that $g(k)$ is undefined, a contradiction. On the other hand, if $g(k)$ is undefined, then $h(k, k) \neq 0$, and so $h(k, k) = 1$. But this means that $\mathrm{Un}(k, k)$ is defined, i.e., that $g(k)$ is defined. □

We can describe this argument in terms of Turing machines. Suppose there were a Turing machine $H$ that took as input a description of a Turing machine $K$ and an input $x$, and decided whether or not $K$ halts on input $x$. Then we could build another Turing machine $G$ which takes a single input $x$, calls $H$ to decide if machine $x$ halts on input $x$, and does the opposite. In other words, if $H$ reports that $x$ halts on input $x$, $G$ goes into an infinite loop, and if $H$ reports that $x$ doesn't halt on input $x$, then $G$ just halts. Does $G$ halt on input $G$? The argument above shows that it does if and only if it doesn't—a contradiction. So our supposition that there is a such Turing machine $H$, is false.

## 27.8 Comparison with Russell's Paradox

It is instructive to compare and contrast the arguments in this section with Russell's paradox:

1. Russell's paradox: let $S = \{x : x \notin x\}$. Then $x \in S$ if and only if $x \notin S$, a contradiction.

   *Conclusion:* There is no such set $S$. Assuming the existence of a "set of all sets" is inconsistent with the other axioms of set theory.

2. A modification of Russell's paradox: let $F$ be the "function" from the set of all functions to $\{0,1\}$, defined by

$$F(f) = \begin{cases} 1 & \text{if } f \text{ is in the domain of } f, \text{ and } f(f) = 0 \\ 0 & \text{otherwise} \end{cases}$$

   A similar argument shows that $F(F) = 0$ if and only if $F(F) = 1$, a contradiction.

   *Conclusion:* $F$ is not a function. The "set of all functions" is too big to be the domain of a function.

3. The diagonalization argument: let $f_0$, $f_1$, ...be the enumeration of the partial computable functions, and let $G\colon \mathbb{N} \to \{0,1\}$ be defined by

$$G(x) = \begin{cases} 1 & \text{if } f_x(x) \downarrow = 0 \\ 0 & \text{otherwise} \end{cases}$$

   If $G$ is computable, then it is the function $f_k$ for some $k$. But then $G(k) = 1$ if and only if $G(k) = 0$, a contradiction.

   *Conclusion:* $G$ is not computable. Note that according to the axioms of set theory, $G$ is still a function; there is no paradox here, just a clarification.

That talk of partial functions, computable functions, partial computable functions, and so on can be confusing. The set of all partial functions from $\mathbb{N}$ to $\mathbb{N}$ is a big collection of objects. Some of them are total, some of them are computable, some are both total and computable, and some are neither. Keep in mind that when we say "function," by default, we mean a total function. Thus we have:

1. computable functions
2. partial computable functions that are not total
3. functions that are not computable
4. partial functions that are neither total nor computable

To sort this out, it might help to draw a big square representing all the partial functions from $\mathbb{N}$ to $\mathbb{N}$, and then mark off two overlapping regions, corresponding to the total functions and the computable partial functions, respectively. It is a good exercise to see if you can describe an object in each of the resulting regions in the diagram.

## 27.9 Computable Sets

We can extend the notion of computability from computable functions to computable sets:

**Definition 27.7.** Let $S$ be a set of natural numbers. Then $S$ is *computable* iff its characteristic function is. In other words, $S$ is computable iff the function

$$\chi_S(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$$

is computable. Similarly, a relation $R(x_0, \ldots, x_{k-1})$ is computable if and only if its characteristic function is.

Computable sets are also called *decidable*.

Notice that we now have a number of notions of computability: for partial functions, for functions, and for sets. Do not get them confused! The Turing machine computing a partial function returns the output of the function, for input values at which the function is defined; the Turing machine computing a set returns either 1 or 0, after deciding whether or not the input value is in the set or not.

## 27.10 Computably Enumerable Sets

**Definition 27.8.** A set is *computably enumerable* if it is empty or the range of a computable function.

**Historical Remarks** Computably enumarable sets are also called *recursively enumerable* instead. This is the original terminology, and today both are commonly used, as well as the abbreviations "c.e." and "r.e."

You should think about what the definition means, and why the terminology is appropriate. The idea is that if $S$ is the range of the computable function $f$, then

$$S = \{f(0), f(1), f(2), \ldots\},$$

and so $f$ can be seen as "enumerating" the elements of $S$. Note that according to the definition, $f$ need not be an increasing function, i.e., the enumeration need not be in increasing order. In fact, $f$ need not even be injective, so that the constant function $f(x) = 0$ enumerates the set $\{0\}$.

Any computable set is computably enumerable. To see this, suppose $S$ is computable. If $S$ is empty, then by definition it is computably enumerable. Otherwise, let $a$ be any element of $S$. Define $f$ by

$$f(x) = \begin{cases} x & \text{if } \chi_S(x) = 1 \\ a & \text{otherwise.} \end{cases}$$

Then $f$ is a computable function, and $S$ is the range of $f$.

## 27.11 Equivalent Defininitions of Computably Enumerable Sets

The following gives a number of important equivalent statements of what it means to be computably enumerable.

**Theorem 27.9.** *Let $S$ be a set of natural numbers. Then the following are equivalent:*

1. *$S$ is computably enumerable.*
2. *$S$ is the range of a* partial *computable function.*
3. *$S$ is empty or the range of a primitive recursive function.*
4. *$S$ is the* domain *of a partial computable function.*

The first three clauses say that we can equivalently take any non-empty computably enumerable set to be enumerated by either a computable function, a partial computable function, or a primitive recursive function. The fourth clause tells us that if $S$ is computably enumerable, then for some index $e$,

$$S = \{x : \varphi_e(x) \downarrow\}.$$

In other words, $S$ is the set of inputs on for which the computation of $\varphi_e$ halts. For that reason, computably enumerable sets are sometimes called *semi-decidable*: if a number is in the set, you eventually get a "yes," but if it isn't, you never get a "no"!

*Proof.* Since every primitive recursive function is computable and every computable function is partial computable, (3) implies (1) and (1) implies (2). (Note that if $S$ is empty, $S$ is the range of the partial computable function that is nowhere defined.) If we show that (2) implies (3), we will have shown the first three clauses equivalent.

So, suppose $S$ is the range of the partial computable function $\varphi_e$. If $S$ is empty, we are done. Otherwise, let $a$ be any element of $S$. By Kleene's normal form theorem, we can write

$$\varphi_e(x) = U(\mu s\ T(e, x, s)).$$

In particular, $\varphi_e(x) \downarrow$ and $= y$ if and only if there is an $s$ such that $T(e, x, s)$ and $U(s) = y$. Define $f(z)$ by

$$f(z) = \begin{cases} U((z)_1) & \text{if } T(e, (z)_0, (z)_1) \\ a & \text{otherwise.} \end{cases}$$

Then $f$ is primitive recursive, because $T$ and $U$ are. Expressed in terms of Turing machines, if $z$ codes a pair $\langle (z)_0, (z)_1 \rangle$ such that $(z)_1$ is a halting computation of machine $e$ on input $(z)_0$, then $f$ returns the output of the computation;

otherwise, it returns $a$.We need to show that $S$ is the range of $f$, i.e., for any natural number $y$, $y \in S$ if and only if it is in the range of $f$. In the forwards direction, suppose $y \in S$. Then $y$ is in the range of $\varphi_e$, so for some $x$ and $s$, $T(e, x, s)$ and $U(s) = y$; but then $y = f(\langle x, s \rangle)$. Conversely, suppose $y$ is in the range of $f$. Then either $y = a$, or for some $z$, $T(e, (z)_0, (z)_1)$ and $U((z)_1) = y$. Since, in the latter case, $\varphi_e(x) \downarrow = y$, either way, $y$ is in $S$.

(The notation $\varphi_e(x) \downarrow = y$ means "$\varphi_e(x)$ is defined and equal to $y$." We could just as well use $\varphi_e(x) = y$, but the extra arrow is sometimes helpful in reminding us that we are dealing with a partial function.)

To finish up the proof of Theorem 27.9, it suffices to show that (1) and (4) are equivalent. First, let us show that (1) implies (4). Suppose $S$ is the range of a computable function $f$, i.e.,

$$S = \{y : \text{for some } x, f(x) = y\}.$$

Let

$$g(y) = \mu x \, f(x) = y.$$

Then $g$ is a partial computable function, and $g(y)$ is defined if and only if for some $x$, $f(x) = y$. In other words, the domain of $g$ is the range of $f$. Expressed in terms of Turing machines: given a Turing machine $F$ that enumerates the elements of $S$, let $G$ be the Turing machine that semi-decides $S$ by searching through the outputs of $F$ to see if a given element is in the set.

Finally, to show (4) implies (1), suppose that $S$ is the domain of the partial computable function $\varphi_e$, i.e.,

$$S = \{x : \varphi_e(x) \downarrow\}.$$

If $S$ is empty, we are done; otherwise, let $a$ be any element of $S$. Define $f$ by

$$f(z) = \begin{cases} (z)_0 & \text{if } T(e, (z)_0, (z)_1) \\ a & \text{otherwise.} \end{cases}$$

Then, as above, a number $x$ is in the range of $f$ if and only if $\varphi_e(x) \downarrow$, i.e., if and only if $x \in S$. Expressed in terms of Turing machines: given a machine $M_e$ that semi-decides $S$, enumerate the elements of $S$ by running through all possible Turing machine computations, and returning the inputs that correspond to halting computations. □

The fourth clause of Theorem 27.9 provides us with a convenient way of enumerating the computably enumerable sets: for each $e$, let $W_e$ denote the domain of $\varphi_e$. Then if $A$ is any computably enumerable set, $A = W_e$, for some $e$.

The following provides yet another characterization of the computably enumerable sets.

**Theorem 27.10.** *A set $S$ is computably enumerable if and only if there is a computable relation $R(x, y)$ such that*

$$S = \{x : \exists y\, R(x, y)\}.$$

*Proof.* In the forward direction, suppose $S$ is computably enumerable. Then for some $e$, $S = W_e$. For this value of $e$ we can write $S$ as

$$S = \{x : \exists y\, T(e, x, y)\}.$$

In the reverse direction, suppose $S = \{x : \exists y\, R(x, y)\}$. Define $f$ by

$$f(x) \simeq \mu y\ AtomRx, y.$$

Then $f$ is partial computable, and $S$ is the domain of $f$. □

## 27.12 Computably Enumerable Sets are Closed under Union and Intersection

The following theorem gives some closure properties on the set of computably enumerable sets.

**Theorem 27.11.** *Suppose $A$ and $B$ are computably enumerable. Then so are $A \cap B$ and $A \cup B$.*

*Proof.* Theorem 27.9 allows us to use various characterizations of the computably enumerable sets. By way of illustration, we will provide a few different proofs.

For the first proof, suppose $A$ is enumerated by a computable function $f$, and $B$ is enumerated by a computable function $g$. Let

$$h(x) = \mu y\ (f(y) = x \vee g(y) = x) \text{ and}$$
$$j(x) = \mu y\ (f((y)_0) = x \wedge g((y)_1) = x).$$

Then $A \cup B$ is the domain of $h$, and $A \cap B$ is the domain of $j$.

Here is what is going on, in computational terms: given procedures that enumerate $A$ and $B$, we can semi-decide if an element $x$ is in $A \cup B$ by looking for $x$ in either enumeration; and we can semi-decide if an element $x$ is in $A \cap B$ for looking for $x$ in both enumerations at the same time.

For the second proof, suppose again that $A$ is enumerated by $f$ and $B$ is enumerated by $g$. Let

$$k(x) = \begin{cases} f(x/2) & \text{if } x \text{ is even} \\ g((x-1)/2) & \text{if } x \text{ is odd.} \end{cases}$$

Then $k$ enumerates $A \cup B$; the idea is that $k$ just alternates between the enumerations offered by $f$ and $g$. Enumerating $A \cap B$ is tricker. If $A \cap B$ is empty, it

is trivially computably enumerable. Otherwise, let $c$ be any element of $A \cap B$, and define $l$ by

$$l(x) = \begin{cases} f((x)_0) & \text{if } f((x)_0) = g((x)_1) \\ c & \text{otherwise.} \end{cases}$$

In computational terms, $l$ runs through pairs of elements in the enumerations of $f$ and $g$, and outputs every match it finds; otherwise, it just stalls by outputting $c$.

For the last proof, suppose $A$ is the *domain* of the partial function $m(x)$ and $B$ is the domain of the partial function $n(x)$. Then $A \cap B$ is the domain of the partial function $m(x) + n(x)$.

In computational terms, if $A$ is the set of values for which $m$ halts and $B$ is the set of values for which $n$ halts, $A \cap B$ is the set of values for which both procedures halt.

Expressing $A \cup B$ as a set of halting values is more difficult, because one has to simulate $m$ and $n$ in parallel. Let $d$ be an index for $m$ and let $e$ be an index for $n$; in other words, $m = \varphi_d$ and $n = \varphi_e$. Then $A \cup B$ is the domain of the function

$$p(x) = \mu y \ (T(d, x, y) \vee T(e, x, y)).$$

In computational terms, on input $x$, $p$ searches for either a halting computation for $m$ or a halting computation for $n$, and halts if it finds either one. □

## 27.13 Computably Enumerable Sets not Closed under Complement

Suppose $A$ is computably enumerable. Is the complement of $A$, $\overline{A} = \mathbb{N} \setminus A$, necessarily computably enumerable as well? The following theorem and corollary show that the answer is "no."

**Theorem 27.12.** *Let $A$ be any set of natural numbers. Then $A$ is computable if and only if both $A$ and $\overline{A}$ are computably enumerable.*

*Proof.* The forwards direction is easy: if $A$ is computable, then $\overline{A}$ is computable as well ($\chi_A = 1 \dot{-} \chi_{\overline{A}}$), and so both are computably enumerable.

In the other direction, suppose $A$ and $\overline{A}$ are both computably enumerable. Let $A$ be the domain of $\varphi_d$, and let $\overline{A}$ be the domain of $\varphi_e$. Define $h$ by

$$h(x) = \mu s \ (T(d, x, s) \vee T(e, x, s)).$$

In other words, on input $x$, $h$ searches for either a halting computation of $\varphi_d$ or a halting computation of $\varphi_e$. Now, if $x \in A$, it will succeed in the first case, and if $x \in \overline{A}$, it will succeed in the second case. So, $h$ is a total computable

function. But now we have that for every $x$, $x \in A$ if and only if $T(e, x, h(x))$, i.e., if $\varphi_e$ is the one that is defined. Since $T(e, x, h(x))$ is a computable relation, $A$ is computable. □

It is easier to understand what is going on in informal computational terms: to decide $A$, on input $x$ search for halting computations of $\varphi_e$ and $\varphi_f$. One of them is bound to halt; if it is $\varphi_e$, then $x$ is in $A$, and otherwise, $x$ is in $\overline{A}$.

**Corollary 27.13.** *$\overline{K_0}$ is not computably enumerable.*

*Proof.* We know that $K_0$ is computably enumerable, but not computable. If $\overline{K_0}$ were computably enumerable, then $K_0$ would be computable by Theorem 27.12. □

## 27.14 Reducibility

We now know that there is at least one set, $K_0$, that is computably enumerable but not computable. It should be clear that there are others. The method of reducibility provides a powerful method of showing that other sets have these properties, without constantly having to return to first principles.

Generally speaking, a "reduction" of a set $A$ to a set $B$ is a method of transforming answers to whether or not elements are in $B$ into answers as to whether or not elements are in $A$. We will focus on a notion called "many-one reducibility," but there are many other notions of reducibility available, with varying properties. Notions of reducibility are also central to the study of computational complexity, where efficiency issues have to be considered as well. For example, a set is said to be "NP-complete" if it is in NP and every NP problem can be reduced to it, using a notion of reduction that is similar to the one described below, only with the added requirement that the reduction can be computed in polynomial time.

We have already used this notion implicitly. Define the set $K$ by

$$K = \{x : \varphi_x(x) \downarrow\},$$

i.e., $K = \{x : x \in W_x\}$. Our proof that the halting problem in unsolvable, Theorem 27.6, shows most directly that $K$ is not computable. Recall that $K_0$ is the set

$$K_0 = \{\langle e, x\rangle : \varphi_e(x) \downarrow\}.$$

i.e. $K_0 = \{\langle x, e\rangle : x \in W_e\}$. It is easy to extend any proof of the uncomputability of $K$ to the uncomputability of $K_0$: if $K_0$ were computable, we could decide whether or not an element $x$ is in $K$ simply by asking whether or not the pair $\langle x, x\rangle$ is in $K_0$. The function $f$ which maps $x$ to $\langle x, x\rangle$ is an example of a *reduction* of $K$ to $K_0$.

**Definition 27.14.** Let $A$ and $B$ be sets. Then $A$ is said to be *many-one reducible* to $B$, written $A \leq_m B$, if there is a computable function $f$ such that for every natural number $x$,

$$x \in A \quad \text{if and only if} \quad f(x) \in B.$$

If $A$ is many-one reducible to $B$ and vice-versa, then $A$ and $B$ are said to be *many-one equivalent*, written $A \equiv_m B$.

If the function $f$ in the definition above happens to be injective, $A$ is said to be *one-one reducible* to $B$. Most of the reductions described below meet this stronger requirement, but we will not use this fact.

It is true, but by no means obvious, that one-one reducibility really is a stronger requirement than many-one reducibility. In other words, there are infinite sets $A$ and $B$ such that $A$ is many-one reducible to $B$ but not one-one reducible to $B$.

## 27.15 Properties of Reducibility

The intuition behind writing $A \leq_m B$ is that $A$ is "no harder than" $B$. The following two propositions support this intuition.

**Proposition 27.15.** *If $A \leq_m B$ and $B \leq_m C$, then $A \leq_m C$.*

*Proof.* Composing a reduction of $A$ to $B$ with a reduction of $B$ to $C$ yields a reduction of $A$ to $C$. (You should check the details!) □

**Proposition 27.16.** *Let $A$ and $B$ be any sets, and suppose $A$ is many-one reducible to $B$.*

1. *If $B$ is computably enumerable, so is $A$.*
2. *If $B$ is computable, so is $A$.*

*Proof.* Let $f$ be a many-one reduction from $A$ to $B$. For the first claim, just check that if $B$ is the domain of a partial function $g$, then $A$ is the domain of $g \circ f$:

$$\begin{aligned} x \in A &\text{iff } f(x) \in B \\ &\text{iff } g(f(x)) \downarrow . \end{aligned}$$

For the second claim, remember that if $B$ is computable then $B$ and $\overline{B}$ are computably enumerable. It is not hard to check that $f$ is also a many-one reduction of $\overline{A}$ to $\overline{B}$, so, by the first part of this proof, $A$ and $\overline{A}$ are computably enumerable. So $A$ is computable as well. (Alternatively, you can check that $\chi_A = \chi_B \circ f$; so if $\chi_B$ is computable, then so is $\chi_A$.) □

A more general notion of reducibility called *Turing reducibility* is useful in other contexts, especially for proving undecidability results. Note that by Corollary 27.13, the complement of $K_0$ is not reducible to $K_0$, since it is not computably enumerable. But, intuitively, if you knew the answers to questions about $K_0$, you would know the answer to questions about its complement as well. A set $A$ is said to be Turing reducible to $B$ if one can determine answers to questions in $A$ using a computable procedure that can ask questions about $B$. This is more liberal than many-one reducibility, in which (1) you are only allowed to ask one question about $B$, and (2) a "yes" answer has to translate to a "yes" answer to the question about $A$, and similarly for "no." It is still the case that if $A$ is Turing reducible to $B$ and $B$ is computable then $A$ is computable as well (though, as we have seen, the analogous statement does not hold for computable enumerability).

You should think about the various notions of reducibility we have discussed, and understand the distinctions between them. We will, however, only deal with many-one reducibility in this chapter. Incidentally, both types of reducibility discussed in the last paragraph have analogues in computational complexity, with the added requirement that the Turing machines run in polynomial time: the complexity version of many-one reducibility is known as *Karp reducibility*, while the complexity version of Turing reducibility is known as *Cook reducibility*.

## 27.16 Complete Computably Enumerable Sets

**Definition 27.17.** A set $A$ is a *complete computably enumerable set* (under many-one reducibility) if

1. $A$ is computably enumerable, and
2. for any other computably enumerable set $B$, $B \leq_m A$.

In other words, complete computably enumerable sets are the "hardest" computably enumerable sets possible; they allow one to answer questions about *any* computably enumerable set.

**Theorem 27.18.** *$K$, $K_0$, and $K_1$ are all complete computably enumerable sets.*

*Proof.* To see that $K_0$ is complete, let $B$ be any computably enumerable set. Then for some index $e$,

$$B = W_e = \{x : \varphi_e(x) \downarrow\}.$$

Let $f$ be the function $f(x) = \langle e, x \rangle$. Then for every natural number $x$, $x \in B$ if and only if $f(x) \in K_0$. In other words, $f$ reduces $B$ to $K_0$.

To see that $K_1$ is complete, note that in the proof of Proposition 27.19 we reduced $K_0$ to it. So, by Proposition 27.15, any computably enumerable set can be reduced to $K_1$ as well.

$K$ can be reduced to $K_0$ in much the same way. □

So, it turns out that all the examples of computably enumerable sets that we have considered so far are either computable, or complete. This should seem strange! Are there any examples of computably enumerable sets that are neither computable nor complete? The answer is yes, but it wasn't until the middle of the 1950s that this was established by Friedberg and Muchnik, independently.

## 27.17 An Example of Reducibility

Let us consider an application of Proposition 27.16.

**Proposition 27.19.** *Let*

$$K_1 = \{e : \varphi_e(0) \downarrow\}.$$

*Then $K_1$ is computably enumerable but not computable.*

*Proof.* Since $K_1 = \{e : \exists s\, T(e, 0, s)\}$, $K_1$ is computably enumerable by Theorem 27.10.

To show that $K_1$ is not computable, let us show that $K_0$ is reducible to it.

This is a little bit tricky, since using $K_1$ we can only ask questions about computations that start with a particular input, 0. Suppose you have a smart friend who can answer questions of this type (friends like this are known as "oracles"). Then suppose someone comes up to you and asks you whether or not $\langle e, x\rangle$ is in $K_0$, that is, whether or not machine $e$ halts on input $x$. One thing you can do is build another machine, $e_x$, that, for *any* input, ignores that input and instead runs $e$ on input $x$. Then clearly the question as to whether machine $e$ halts on input $x$ is equivalent to the question as to whether machine $e_x$ halts on input 0 (or any other input). So, then you ask your friend whether this new machine, $e_x$, halts on input 0; your friend's answer to the modified question provides the answer to the original one. This provides the desired reduction of $K_0$ to $K_1$.

Using the universal partial computable function, let $f$ be the 3-ary function defined by

$$f(x, y, z) \simeq \varphi_x(y).$$

Note that $f$ ignores its third input entirely. Pick an index $e$ such that $f = \varphi^3_e$; so we have

$$\varphi^3_e(x, y, z) \simeq \varphi_x(y).$$

By the *s-m-n* theorem, there is a function $s(e, x, y)$ such that, for every $z$,

$$\begin{aligned}\varphi_{s(e,x,y)}(z) &\simeq \varphi_e^3(x, y, z)\\ &\simeq \varphi_x(y).\end{aligned}$$

In terms of the informal argument above, $s(e, x, y)$ is an index for the machine that, for any input $z$, ignores that input and computes $\varphi_x(y)$.

In particular, we have

$$\varphi_{s(e,x,y)}(0)\downarrow \quad \text{if and only if} \quad \varphi_x(y)\downarrow.$$

In other words, $\langle x, y\rangle \in K_0$ if and only if $s(e, x, y) \in K_1$. So the function $g$ defined by

$$g(w) = s(e, (w)_0, (w)_1)$$

is a reduction of $K_0$ to $K_1$. □

## 27.18 Totality is Undecidable

Let us consider one more example of using the *s-m-n* theorem to show that something is noncomputable. Let Tot be the set of indices of total computable functions, i.e.

$$\text{Tot} = \{x : \text{for every } y,\ \varphi_x(y)\downarrow\}.$$

**Proposition 27.20.** Tot *is not computable.*

*Proof.* To see that Tot is not computable, it suffices to show that $K$ is reducible to it. Let $h(x, y)$ be defined by

$$h(x, y) \simeq \begin{cases} 0 & \text{if } x \in K\\ \text{undefined} & \text{otherwise}\end{cases}$$

Note that $h(x, y)$ does not depend on $y$ at all. It should not be hard to see that $h$ is partial computable: on input $x, y$, the we compute $h$ by first simulating the function $\varphi_x$ on input $x$; if this computation halts, $h(x, y)$ outputs 0 and halts. So $h(x, y)$ is just $Z(\mu s\ T(x, x, s))$, where $Z$ is the constant zero function.

Using the *s-m-n* theorem, there is a primitive recursive function $k(x)$ such that for every $x$ and $y$,

$$\varphi_{k(x)}(y) = \begin{cases} 0 & \text{if } x \in K\\ \text{undefined} & \text{otherwise}\end{cases}$$

So $\varphi_{k(x)}$ is total if $x \in K$, and undefined otherwise. Thus, $k$ is a reduction of $K$ to Tot. □

It turns out that Tot is not even computably enumerable—its complexity lies further up on the "arithmetic hierarchy." But we will not worry about this strengthening here.

## 27.19 Rice's Theorem

If you think about it, you will see that the specifics of Tot do not play into the proof of Proposition 27.20. We designed $h(x, y)$ to act like the constant function $j(y) = 0$ exactly when $x$ is in $K$; but we could just as well have made it act like any other partial computable function under those circumstances. This observation lets us state a more general theorem, which says, roughly, that no nontrivial property of computable functions is decidable.

Keep in mind that $\varphi_0$, $\varphi_1$, $\varphi_2$, ... is our standard enumeration of the partial computable functions.

**Theorem 27.21** (Rice's Theorem)**.** *Let $C$ be any set of partial computable functions, and let $A = \{n : \varphi_n \in C\}$. If $A$ is computable, then either $C$ is $\emptyset$ or $C$ is the set of all the partial computable functions.*

An *index set* is a set $A$ with the property that if $n$ and $m$ are indices which "compute" the same function, then either both $n$ and $m$ are in $A$, or neither is. It is not hard to see that the set $A$ in the theorem has this property. Conversely, if $A$ is an index set and $C$ is the set of functions computed by these indices, then $A = \{n : \varphi_n \in C\}$.

With this terminology, Rice's theorem is equivalent to saying that no nontrivial index set is decidable. To understand what the theorem says, it is helpful to emphasize the distinction between *programs* (say, in your favorite programming language) and the functions they compute. There are certainly questions about programs (indices), which are syntactic objects, that are computable: does this program have more than 150 symbols? Does it have more than 22 lines? Does it have a "while" statement? Does the string "hello world" every appear in the argument to a "print" statement? Rice's theorem says that no nontrivial question about the program's *behavior* is computable. This includes questions like these: does the program halt on input 0? Does it ever halt? Does it ever output an even number?

*Proof of Rice's theorem.* Suppose $C$ is neither $\emptyset$ nor the set of all the partial computable functions, and let $A$ be the set of indices of functions in $C$. We will show that if $A$ were computable, we could solve the halting problem; so $A$ is not computable.

Without loss of generality, we can assume that the function $f$ which is nowhere defined is not in $C$ (otherwise, switch $C$ and its complement in the argument below). Let $g$ be any function in $C$. The idea is that if we could decide $A$, we could tell the difference between indices computing $f$, and indices computing $g$; and then we could use that capability to solve the halting problem.

Here's how. Using the universal computation predicate, we can define a function

$$h(x,y) \simeq \begin{cases} \text{undefined} & \text{if } \varphi_x(x) \uparrow \\ g(y) & \text{otherwise.} \end{cases}$$

To compute $h$, first we try to compute $\varphi_x(x)$; if that computation halts, we go on to compute $g(y)$; and if *that* computation halts, we return the output. More formally, we can write

$$h(x,y) \simeq P^2_0(g(y), \mathrm{Un}(x,x)).$$

where $P^2_0(z_0, z_1) = z_0$ is the 2-place projection function returning the 0-th argument, which is computable.

Then $h$ is a composition of partial computable functions, and the right side is defined and equal to $g(y)$ just when $\mathrm{Un}(x,x)$ and $g(y)$ are both defined.

Notice that for a fixed $x$, if $\varphi_x(x)$ is undefined, then $h(x,y)$ is undefined for every $y$; and if $\varphi_x(x)$ is defined, then $h(x,y) \simeq g(y)$. So, for any fixed value of $x$, either $h(x,y)$ acts just like $f$ or it acts just like $g$, and deciding whether or not $\varphi_x(x)$ is defined amounts to deciding which of these two cases holds. But this amounts to deciding whether or not $h_x(y) \simeq h(x,y)$ is in $C$ or not, and if $A$ were computable, we could do just that.

More formally, since $h$ is partial computable, it is equal to the function $\varphi_k$ for some index $k$. By the *s-m-n* theorem there is a primitive recursive function $s$ such that for each $x$, $\varphi_{s(k,x)}(y) = h_x(y)$. Now we have that for each $x$, if $\varphi_x(x) \downarrow$, then $\varphi_{s(k,x)}$ is the same function as $g$, and so $s(k,x)$ is in $A$. On the other hand, if $\varphi_x(x) \uparrow$, then $\varphi_{s(k,x)}$ is the same function as $f$, and so $s(k,x)$ is not in $A$. In other words we have that for every $x$, $x \in K$ if and only if $s(k,x) \in A$. If $A$ were computable, $K$ would be also, which is a contradiction. So $A$ is not computable. □

Rice's theorem is very powerful. The following immediate corollary shows some sample applications.

**Corollary 27.22.** *The following sets are undecidable.*

1. $\{x : 17 \text{ is in the range of } \varphi_x\}$
2. $\{x : \varphi_x \text{ is constant}\}$
3. $\{x : \varphi_x \text{ is total}\}$
4. $\{x : \text{whenever } y < y', \varphi_x(y) \downarrow, \text{ and if } \varphi_x(y') \downarrow, \text{ then } \varphi_x(y) < \varphi_x(y')\}$

*Proof.* These are all nontrivial index sets. □

## 27.20 The Fixed-Point Theorem

Let's consider the halting problem again. As temporary notation, let us write $\ulcorner \varphi_x(y) \urcorner$ for $\langle x, y \rangle$; think of this as representing a "name" for the value $\varphi_x(y)$. With this notation, we can reword one of our proofs that the halting problem is undecidable.

Question: is there a computable function $h$, with the following property? For every $x$ and $y$,

$$h(\ulcorner \varphi_x(y) \urcorner) = \begin{cases} 1 & \text{if } \varphi_x(y) \downarrow \\ 0 & \text{otherwise.} \end{cases}$$

Answer: No; otherwise, the partial function

$$g(x) \simeq \begin{cases} 0 & \text{if } h(\ulcorner \varphi_x(x) \urcorner) = 0 \\ \text{undefined} & \text{otherwise} \end{cases}$$

would be computable, and so have some index $e$. But then we have

$$\varphi_e(e) \simeq \begin{cases} 0 & \text{if } h(\ulcorner \varphi_e(e) \urcorner) = 0 \\ \text{undefined} & \text{otherwise,} \end{cases}$$

in which case $\varphi_e(e)$ is defined if and only if it isn't, a contradiction.

Now, take a look at the equation with $\varphi_e$. There is an instance of self-reference there, in a sense: we have arranged for the value of $\varphi_e(e)$ to depend on $\ulcorner \varphi_e(e) \urcorner$, in a certain way. The fixed-point theorem says that we *can* do this, in general—not just for the sake of proving contradictions.

Lemma 27.23 gives two equivalent ways of stating the fixed-point theorem. Logically speaking, the fact that the statements are equivalent follows from the fact that they are both true; but what we really mean is that each one follows straightforwardly from the other, so that they can be taken as alternative statements of the same theorem.

**Lemma 27.23.** *The following statements are equivalent:*

1. *For every partial computable function $g(x, y)$, there is an index $e$ such that for every $y$,*
$$\varphi_e(y) \simeq g(e, y).$$
2. *For every computable function $f(x)$, there is an index $e$ such that for every $y$,*
$$\varphi_e(y) \simeq \varphi_{f(e)}(y).$$

*Proof.* (1) $\Rightarrow$ (2): Given $f$, define $g$ by $g(x, y) \simeq \mathrm{Un}(f(x), y)$. Use (1) to get an index $e$ such that for every $y$,

$$\begin{aligned} \varphi_e(y) &= \mathrm{Un}(f(e), y) \\ &= \varphi_{f(e)}(y). \end{aligned}$$

(2) $\Rightarrow$ (1): Given $g$, use the *s-m-n* theorem to get $f$ such that for every $x$ and $y$, $\varphi_{f(x)}(y) \simeq g(x,y)$. Use (2) to get an index $e$ such that

$$\begin{aligned}\varphi_e(y) &= \varphi_{f(e)}(y)\\ &= g(e,y).\end{aligned}$$

This concludes the proof. □

Before showing that statement (1) is true (and hence (2) as well), consider how bizarre it is. Think of $e$ as being a computer program; statement (1) says that given any partial computable $g(x,y)$, you can find a computer program $e$ that computes $g_e(y) \simeq g(e,y)$. In other words, you can find a computer program that computes a function that references the program itself.

**Theorem 27.24.** *The two statements in Lemma 27.23 are true. Specifically, for every partial computable function $g(x,y)$, there is an index $e$ such that for every $y$,*

$$\varphi_e(y) \simeq g(e,y).$$

*Proof.* The ingredients are already implicit in the discussion of the halting problem above. Let $\mathrm{diag}(x)$ be a computable function which for each $x$ returns an index for the function $f_x(y) \simeq \varphi_x(x,y)$, i.e.

$$\varphi_{\mathrm{diag}(x)}(y) \simeq \varphi_x(x,y).$$

Think of diag as a function that transforms a program for a 2-ary function into a program for a 1-ary function, obtained by fixing the original program as its first argument. The function diag can be defined formally as follows: first define $s$ by

$$s(x,y) \simeq \mathrm{Un}^2(x,x,y),$$

where $\mathrm{Un}^2$ is a 3-ary function that is universal for partial computable 2-ary functions. Then, by the *s-m-n* theorem, we can find a primitive recursive function diag satisfying

$$\varphi_{\mathrm{diag}(x)}(y) \simeq s(x,y).$$

Now, define the function $l$ by

$$l(x,y) \simeq g(\mathrm{diag}(x),y).$$

and let $\ulcorner l \urcorner$ be an index for $l$. Finally, let $e = \mathrm{diag}(\ulcorner l \urcorner)$. Then for every $y$, we have

$$\begin{aligned}\varphi_e(y) &\simeq \varphi_{\mathrm{diag}(\ulcorner l \urcorner)}(y)\\ &\simeq \varphi_{\ulcorner l \urcorner}(\ulcorner l \urcorner, y)\\ &\simeq l(\ulcorner l \urcorner, y)\\ &\simeq g(\mathrm{diag}(\ulcorner l \urcorner), y)\\ &\simeq g(e,y),\end{aligned}$$

as required. □

What's going on? Suppose you are given the task of writing a computer program that prints itself out. Suppose further, however, that you are working with a programming language with a rich and bizarre library of string functions. In particular, suppose your programming language has a function diag which works as follows: given an input string $s$, diag locates each instance of the symbol 'x' occuring in $s$, and replaces it by a quoted version of the original string. For example, given the string

```
hello x world
```

as input, the function returns

```
hello 'hello x world' world
```

as output. In that case, it is easy to write the desired program; you can check that

```
print(diag('print(diag(x))'))
```

does the trick. For more common programming languages like C++ and Java, the same idea (with a more involved implementation) still works.

We are only a couple of steps away from the proof of the fixed-point theorem. Suppose a variant of the print function print$(x, y)$ accepts a string $x$ and another numeric argument $y$, and prints the string $x$ repeatedly, $y$ times. Then the "program"

```
getinput(y); print(diag('getinput(y); print(diag(x), y)'), y)
```

prints itself out $y$ times, on input $y$. Replacing the getinput—print—diag skeleton by an arbitrary funtion $g(x, y)$ yields

```
g(diag('g(diag(x), y)'), y)
```

which is a program that, on input $y$, runs $g$ on the program itself and $y$. Thinking of "quoting" with "using an index for," we have the proof above.

For now, it is o.k. if you want to think of the proof as formal trickery, or black magic. But you should be able to reconstruct the details of the argument given above. When we prove the incompleteness theorems (and the related "fixed-point theorem") we will discuss other ways of understanding why it works.

The same idea can be used to get a "fixed point" combinator. Suppose you have a lambda term $g$, and you want another term $k$ with the property that $k$ is $\beta$-equivalent to $gk$. Define terms

$$\text{diag}(x) = xx$$

and

$$l(x) = g(\text{diag}(x))$$

using our notational conventions; in other words, $l$ is the term $\lambda x.\, g(xx)$. Let $k$ be the term $ll$. Then we have

$$\begin{aligned} k &= (\lambda x.\, g(xx))(\lambda x.\, g(xx)) \\ &\triangleright g((\lambda x.\, g(xx))(\lambda x.\, g(xx))) \\ &= gk. \end{aligned}$$

If one takes

$$Y = \lambda g.\, ((\lambda x.\, g(xx))(\lambda x.\, g(xx)))$$

then $Yg$ and $g(Yg)$ reduce to a common term; so $Yg \equiv_\beta g(Yg)$. This is known as "Curry's combinator." If instead one takes

$$Y = (\lambda xg.\, g(xxg))(\lambda xg.\, g(xxg))$$

then in fact $Yg$ reduces to $g(Yg)$, which is a stronger statement. This latter version of $Y$ is known as "Turing's combinator."

## 27.21 Applying the Fixed-Point Theorem

The fixed-point theorem essentially lets us define partial computable functions in terms of their indices. For example, we can find an index $e$ such that for every $y$,

$$\varphi_e(y) = e + y.$$

As another example, one can use the proof of the fixed-point theorem to design a program in Java or C++ that prints itself out.

Remember that if for each $e$, we let $W_e$ be the domain of $\varphi_e$, then the sequence $W_0, W_1, W_2, \ldots$ enumerates the computably enumerable sets. Some of these sets are computable. One can ask if there is an algorithm which takes as input a value $x$, and, if $W_x$ happens to be computable, returns an index for its characteristic function. The answer is "no," there is no such algorithm:

**Theorem 27.25.** *There is no partial computable function $f$ with the following property: whenever $W_e$ is computable, then $f(e)$ is defined and $\varphi_{f(e)}$ is its characteristic function.*

*Proof.* Let $f$ be any computable function; we will construct an $e$ such that $W_e$ is computable, but $\varphi_{f(e)}$ is not its characteristic function. Using the fixed point theorem, we can find an index $e$ such that

$$\varphi_e(y) \simeq \begin{cases} 0 & \text{if } y = 0 \text{ and } \varphi_{f(e)}(0) \downarrow = 0 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

That is, $e$ is obtained by applying the fixed-point theorem to the function defined by

$$g(x, y) \simeq \begin{cases} 0 & \text{if } y = 0 \text{ and } \varphi_{f(x)}(0) \downarrow = 0 \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Informally, we can see that $g$ is partial computable, as follows: on input $x$ and $y$, the algorithm first checks to see if $y$ is equal to 0. If it is, the algorithm computes $f(x)$, and then uses the universal machine to compute $\varphi_{f(x)}(0)$. If this last computation halts and returns 0, the algorithm returns 0; otherwise, the algorithm doesn't halt.

But now notice that if $\varphi_{f(e)}(0)$ is defined and equal to 0, then $\varphi_e(y)$ is defined exactly when $y$ is equal to 0, so $W_e = \{0\}$. If $\varphi_{f(e)}(0)$ is not defined, or is defined but not equal to 0, then $W_e = \emptyset$. Either way, $\varphi_{f(e)}$ is not the characteristic function of $W_e$, since it gives the wrong answer on input 0. □

## 27.22 Defining Functions using Self-Reference

It is generally useful to be able to define functions in terms of themselves. For example, given computable functions $k$, $l$, and $m$, the fixed-point lemma tells us that there is a partial computable function $f$ satisfying the following equation for every $y$:

$$f(y) \simeq \begin{cases} k(y) & \text{if } l(y) = 0 \\ f(m(y)) & \text{otherwise.} \end{cases}$$

Again, more specifically, $f$ is obtained by letting

$$g(x, y) \simeq \begin{cases} k(y) & \text{if } l(y) = 0 \\ \varphi_x(m(y)) & \text{otherwise} \end{cases}$$

and then using the fixed-point lemma to find an index $e$ such that $\varphi_e(y) = g(e, y)$.

For a concrete example, the "greatest common divisor" function $\gcd(u, v)$ can be defined by

$$\gcd(u, v) \simeq \begin{cases} v & \text{if } 0 = 0 \\ \gcd(\text{mod}(v, u), u) & \text{otherwise} \end{cases}$$

where $\text{mod}(v, u)$ denotes the remainder of dividing $v$ by $u$. An appeal to the fixed-point lemma shows that gcd is partial computable. (In fact, this can be put in the format above, letting $y$ code the pair $\langle u, v \rangle$.) A subsequent induction on $u$ then shows that, in fact, gcd is total.

Of course, one can cook up self-referential definitions that are much fancier than the examples just discussed. Most programming languages support definitions of functions in terms of themselves, one way or another. Note that this is a little bit less dramatic than being able to define a function in terms of an *index* for an algorithm computing the functions, which is what, in full generality, the fixed-point theorem lets you do.

## 27.23 Minimization with Lambda Terms

When it comes to the lambda calculus, we've shown the following:

1. Every primitive recursive function is represented by a lambda term.
2. There is a lambda term $Y$ such that for any lambda term $G$, $YG \triangleright G(YG)$.

To show that every partial computable function is represented by some lambda term, we only need to show the following.

**Lemma 27.26.** *Suppose $f(x, y)$ is primitive recursive. Let $g$ be defined by*

$$g(x) \simeq \mu y\ f(x, y) = 0.$$

*Then $g$ is represented by a lambda term.*

*Proof.* The idea is roughly as follows. Given $x$, we will use the fixed-point lambda term $Y$ to define a function $h_x(n)$ which searches for a $y$ starting at $n$; then $g(x)$ is just $h_x(0)$. The function $h_x$ can be expressed as the solution of a fixed-point equation:

$$h_x(n) \simeq \begin{cases} n & \text{if } f(x, n) = 0 \\ h_x(n+1) & \text{otherwise.} \end{cases}$$

Here are the details. Since $f$ is primitive recursive, it is represented by some term $F$. Remember that we also have a lambda term $D$ such that $D(M, N, \overline{0}) \triangleright M$ and $D(M, N, \overline{1}) \triangleright N$. Fixing $x$ for the moment, to represent $h_x$ we want to find a term $H$ (depending on $x$) satisfying

$$H(\overline{n}) \equiv D(\overline{n}, H(S(\overline{n})), F(x, \overline{n})).$$

We can do this using the fixed-point term $Y$. First, let $U$ be the term

$$\lambda h.\, \lambda z.\, D(z, (h(Sz)), F(x, z)),$$

and then let $H$ be the term $YU$. Notice that the only free variable in $H$ is $x$. Let us show that $H$ satisfies the equation above.

By the definition of $Y$, we have

$$H = YU \equiv U(YU) = U(H).$$

In particular, for each natural number $n$, we have

$$\begin{aligned} H(\overline{n}) &\equiv U(H, \overline{n}) \\ &\triangleright D(\overline{n}, H(S(\overline{n})), F(x, \overline{n})), \end{aligned}$$

as required. Notice that if you substitute a numeral $\overline{m}$ for $x$ in the last line, the expression reduces to $\overline{n}$ if $F(\overline{m}, \overline{n})$ reduces to $\overline{0}$, and it reduces to $H(S(\overline{n}))$ if $F(\overline{m}, \overline{n})$ reduces to any other numeral.

To finish off the proof, let $G$ be $\lambda x.\, H(\overline{0})$. Then $G$ represents $g$; in other words, for every $m$, $G(\overline{m})$ reduces to reduces to $\overline{g(m)}$, if $g(m)$ is defined, and has no normal form otherwise. □

## Problems

**Problem 27.1.** Give a reduction of $K$ to $K_0$.

# Part VI

# Turing Machines

The material in this part is a basic and informal introduction to Turing machines. It needs more examples and exercises, and perhaps information on available Turing machine simulators. The proof of the unsolvability of the decision problem uses a successor function, hence all models are infinite. One could strengthen the result by using a successor relation instead. There probably are subtle oversights; use these as checks on students' attention (but also file issues!).

# Chapter 28

# Turing Machine Computations

## 28.1 Introduction

What does it mean for a function, say, from $\mathbb{N}$ to $\mathbb{N}$ to be *computable*? Among the first answers, and the most well known one, is that a function is computable if it can be computed by a Turing machine. This notion was set out by Alan Turing in 1936. Turing machines are an example of *a model of computation*—they are a mathematically precise way of defining the idea of a "computational procedure." What exactly that means is debated, but it is widely agreed that Turing machines are one way of specifying computational procedures. Even though the term "Turing machine" evokes the image of a physical machine with moving parts, strictly speaking a Turing machine is a purely mathematical construct, and as such it idealizes the idea of a computational procedure. For instance, we place no restriction on either the time or memory requirements of a Turing machine: Turing machines can compute something even if the computation would require more storage space or more steps than there are atoms in the universe.

It is perhaps best to think of a Turing machine as a program for a special kind of imaginary mechanism. This mechanism consists of a *tape* and a *read-write head*. In our version of Turing machines, the tape is infinite in one direction (to the right), and it is divided into *squares*, each of which may contain a symbol from a finite *alphabet*. Such alphabets can contain any number of different symbols, say, but we will mainly make do with three: $\triangleright$, 0, and 1. When the mechanism is started, the tape is empty (i.e., each square contains the symbol 0) except for the leftmost square, which contains $\triangleright$, and a finite number of squares which contain the *input*. At any time, the mechanism is in one of a finite number of *states*. At the outset, the head scans the leftmost square and in a specified *initial state*. At each step of the mechanism's run, the content of the square currently scanned together with the state the mechanism is in and the Turing machine program determine what happens next. The Turing machine program is given by a partial function which takes as input a state $q$

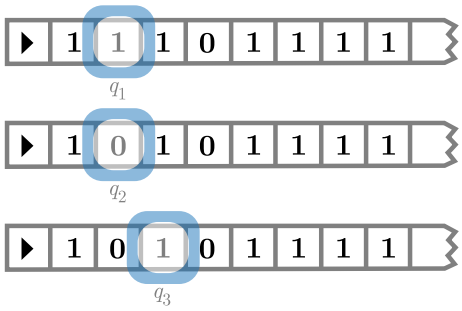



Figure 28.1: A Turing machine executing its program.

and a symbol $\sigma$ and outputs a triple $\langle q', \sigma', D \rangle$. Whenever the mechanism is in state $q$ and reads symbol $\sigma$, it replaces the symbol on the current square with $\sigma'$, the head moves left, right, or stays put according to whether $D$ is $L$, $R$, or $N$, and the mechanism goes into state $q'$.

For instance, consider the situation in section 28.1. The visible part of the tape of the Turing machine contains the end-of-tape symbol ▷ on the leftmost square, followed by three 1's, a 0, and four more 1's. The head is reading the third square from the left, which contains a 1, and is in state $q_1$—we say "the machine is reading a 1 in state $q_1$." If the program of the Turing machine returns, for input $\langle q_1, 1 \rangle$, the triple $\langle q_2, 0, N \rangle$, the machine would now replace the 1 on the third square with a 0, leave the read/write head where it is, and switch to state $q_2$. If then the program returns $\langle q_3, 0, R \rangle$ for input $\langle q_2, 0 \rangle$, the machine would now overwrite the 0 with another 0 (effectively, leaving the content of the tape under the read/write head unchanged), move one square to the right, and enter state $q_3$. And so on.

We say that the machine *halts* when it encounters some state, $q_n$, and symbol, $\sigma$ such that there is no instruction for $\langle q_n, \sigma \rangle$, i.e., the transition function for input $\langle q_n, \sigma \rangle$ is undefined. In other words, the machine has no instruction to carry out, and at that point, it ceases operation. Halting is sometimes represented by a specific halt state $h$. This will be demonstrated in more detail later on.

The beauty of Turing's paper, "On computable numbers," is that he presents not only a formal definition, but also an argument that the definition captures the intuitive notion of computability. From the definition, it should be clear that any function computable by a Turing machine is computable in the intuitive sense. Turing offers three types of argument that the converse is true, i.e., that any function that we would naturally regard as computable is computable by such a machine. They are (in Turing's words):

1. A direct appeal to intuition.

2. A proof of the equivalence of two definitions (in case the new definition has a greater intuitive appeal).

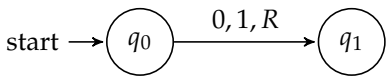
3. Giving examples of large classes of numbers which are computable.

Our goal is to try to define the notion of computability "in principle," i.e., without taking into account practical limitations of time and space. Of course, with the broadest definition of computability in place, one can then go on to consider computation with bounded resources; this forms the heart of the subject known as "computational complexity."

**Historical Remarks** Alan Turing invented Turing machines in 1936. While his interest at the time was the decidability of first-order logic, the paper has been described as a definitive paper on the foundations of computer design. In the paper, Turing focuses on computable real numbers, i.e., real numbers whose decimal expansions are computable; but he notes that it is not hard to adapt his notions to computable functions on the natural numbers, and so on. Notice that this was a full five years before the first working general purpose computer was built in 1941 (by the German Konrad Zuse in his parent's living room), seven years before Turing and his colleagues at Bletchley Park built the code-breaking Colossus (1943), nine years before the American ENIAC (1945), twelve years before the first British general purpose computer—the Manchester Small-Scale Experimental Machine—was built in Manchester (1948), and thirteen years before the Americans first tested the BINAC (1949). The Manchester SSEM has the distinction of being the first stored-program computer—previous machines had to be rewired by hand for each new task.

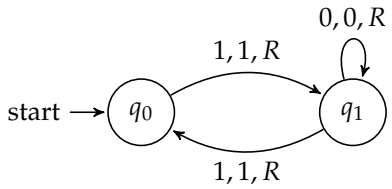
## 28.2 Representing Turing Machines

Turing machines can be represented visually by *state diagrams*. The diagrams are composed of state cells connected by arrows. Unsurprisingly, each state cell represents a state of the machine. Each arrow represents an instruction that can be carried out from that state, with the specifics of the instruction written above or below the appropriate arrow. Consider the following machine, which has only two internal states, $q_0$ and $q_1$, and one instruction:




Recall that the Turing machine has a read/write head and a tape with the input written on it. The instruction can be read as *if reading a blank in state $q_0$, write a stroke, move right, and move to state $q_1$*. This is equivalent to the transition function mapping $\langle q_0, 0\rangle$ to $\langle q_1, 1, R\rangle$.

**Example 28.1.** *Even Machine*: The following Turing machine halts if, and only if, there are an even number of strokes on the tape.




The state diagram corresponds to the following transition function:

$$\delta(q_0, 1) = \langle q_1, 1, R \rangle,$$
$$\delta(q_1, 1) = \langle q_0, 1, R \rangle,$$
$$\delta(q_1, 0) = \langle q_1, 0, R \rangle$$

The above machine halts only when the input is an even number of strokes. Otherwise, the machine (theoretically) continues to operate indefinitely. For any machine and input, it is possible to trace through the *configurations* of the machine in order to determine the output. We will give a formal definition of configurations later. For now, we can intuitively think of configurations as a series of diagrams showing the state of the machine at any point in time during operation. Configurations show the content of the tape, the state of the machine and the location of the read/write head.

Let us trace through the configurations of the even machine if it is started with an input of 4 1s. In this case, we expect that the machine will halt. We will then run the machine on an input of 3 1s, where the machine will run forever.

The machine starts in state $q_0$, scanning the leftmost 1. We can represent the initial state of the machine as follows:

$$\triangleright 1_0 1110 \ldots$$

The above configuration is straightforward. As can be seen, the machine starts in state one, scanning the leftmost 1. This is represented by a subscript of the state name on the first 1. The applicable instruction at this point is $\delta(q_0, 1) = \langle q_1, 1, R \rangle$, and so the machine moves right on the tape and changes to state $q_1$.

$$\triangleright 11_1 110 \ldots$$

Since the machine is now in state $q_1$ scanning a stroke, we have to "follow" the instruction $\delta(q_1, 1) = \langle q_0, 1, R \rangle$. This results in the configuration

$$\triangleright 111_0 10 \ldots$$

As the machine continues, the rules are applied again in the same order, resulting in the following two configurations:

$$\triangleright 1111_1 0 \ldots$$

$$\triangleright 11110_0 \ldots$$

The machine is now in state $q_0$ scanning a blank. Based on the transition diagram, we can easily see that there is no instruction to be carried out, and thus the machine has halted. This means that the input has been accepted.

Suppose next we start the machine with an input of three strokes. The first few configurations are similar, as the same instructions are carried out, with only a small difference of the tape input:

$$\triangleright 1_0 110 \ldots$$

$$\triangleright 11_1 10 \ldots$$

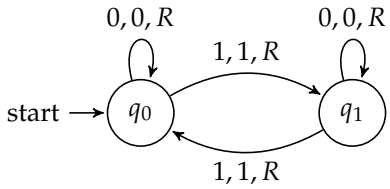
$$\triangleright 111_0 0 \ldots$$

$$\triangleright 1110_1 \ldots$$

The machine has now traversed past all the strokes, and is reading a blank in state $q_1$. As shown in the diagram, there is an instruction of the form $\delta(q_1, 0) = \langle q_1, 0, R \rangle$. Since the tape is infinitely blank to the right, the machine will continue to execute this instruction *forever*, staying in state $q_1$ and moving ever further to the right. The machine will never halt, and does not accept the input.

It is important to note that not all machines will halt. If halting means that the machine runs out of instructions to execute, then we can create a machine that never halts simply by ensuring that there is an outgoing arrow for each symbol at each state. The even machine can be modified to run infinitely by adding an instruction for scanning a blank at $q_0$.

**Example 28.2.**




Machine tables are another way of representing Turing machines. Machine tables have the tape alphabet displayed on the $x$-axis, and the set of machine states across the $y$-axis. Inside the table, at the intersection of each state and symbol, is written the rest of the instruction—the new state, new symbol, and direction of movement. Machine tables make it easy to determine in what

state, and for what symbol, the machine halts. Whenever there is a gap in the table is a possible point for the machine to halt. Unlike state diagrams and instruction sets, where the points at which the machine halts are not always immediately obvious, any halting points are quickly identified by finding the gaps in the machine table.

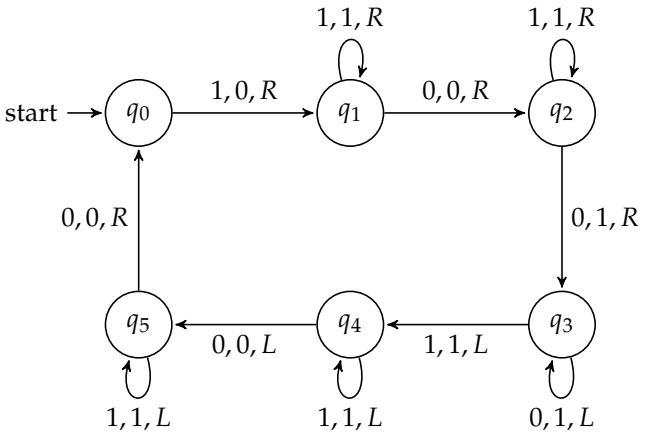
**Example 28.3.** The machine table for the even machine is:

| | 0 | 1 |
|---|---|---|
| $q_0$ | | $1, q_1, R$ |
| $q_1$ | $0, q_1, 0$ | $1, q_0, R$ |

As we can see, the machine halts when scanning a blank in state $q_0$.

So far we have only considered machines that read and accept input. However, Turing machines have the capacity to both read and write. An example of such a machine (although there are many, many examples) is a *doubler*. A doubler, when started with a block of $n$ strokes on the tape, outputs a block of $2n$ strokes.

**Example 28.4.** Before building a doubler machine, it is important to come up with a *strategy* for solving the problem. Since the machine (as we have formulated it) cannot remember how many strokes it has read, we need to come up with a way to keep track of all the strokes on the tape. One such way is to separate the output from the input with a blank. The machine can then erase the first stroke from the input, traverse over the rest of the input, leave a blank, and write two new strokes. The machine will then go back and find the second stroke in the input, and double that one as well. For each one stroke of input, it will write two strokes of output. By erasing the input as the machine goes, we can guarantee that no stroke is missed or doubled twice. When the entire input is erased, there will be $2n$ strokes left on the tape.



## 28.3 Turing Machines

The formal definition of what constitutes a Turing machine looks abstract, but is actually simple: it merely packs into one mathematical structure all the information needed to specify the workings of a Turing machine. This includes (1) which states the machine can be in, (2) which symbols are allowed to be on the tape, (3) which state the machine should start in, and (4) what the instruction set of the machine is.

**Definition 28.5** (Turing machine)**.** A *Turing machine* $T = \langle Q, \Sigma, q_0, \delta \rangle$ consists of

1. a finite set of *states* $Q$,
2. a finite *alphabet* $\Sigma$ which includes $\triangleright$ and 0,
3. an *initial state* $q_0 \in Q$,
4. a finite *instruction set* $\delta \colon Q \times \Sigma \nrightarrow Q \times \Sigma \times \{L, R, N\}$.

The partial function $\delta$ is also called the *transition function* of $T$.

We assume that the tape is infinite in one direction only. For this reason it is useful to designate a special symbol $\triangleright$ as a marker for the left end of the tape. This makes it easier for Turing machine programs to tell when they're "in danger" of running off the tape.

**Example 28.6.** *Even Machine:* The even machine is formally the quadruple $\langle Q, \Sigma, q_0, \delta \rangle$ where

$$\begin{aligned} Q &= \{q_0, q_1\} \\ \Sigma &= \{\triangleright, 0, 1\}, \\ \delta(q_0, 1) &= \langle q_1, 1, R \rangle, \\ \delta(q_1, 1) &= \langle q_0, 1, R \rangle, \\ \delta(q_1, 0) &= \langle q_1, 0, R \rangle. \end{aligned}$$

## 28.4 Configurations and Computations

Recall tracing through the configurations of the even machine earlier. The imaginary mechanism consisting of tape, read/write head, and Turing machine program is really just in intuitive way of visualizing what a Turing machine computation is. Formally, we can define the computation of a Turing machine on a given input as a sequence of *configurations*—and a configuration in turn is a sequence of symbols (corresponding to the contents of the tape at a given point in the computation), a number indicating the position of the read/write head, and a state. Using these, we can define what the Turing machine $M$ computes on a given input.

**Definition 28.7** (Configuration)**.** A *configuration* of Turing machine $M = \langle Q, \Sigma, q_0, \delta \rangle$ is a triple $\langle C, n, q \rangle$ where

1. $C \in \Sigma^*$ is a finite sequence of symbols from $\Sigma$,
2. $n \in \mathbb{N}$ is a number $< \text{len}(C)$, and
3. $q \in Q$

Intuitively, the sequence $C$ is the content of the tape (symbols of all squares from the leftmost square to the last non-blank or previously visited square), $n$ is the number of the square the read/write head is scanning (beginning with 0 being the number of the leftmost square), and $q$ is the current state of the machine.

The potential input for a Turing machine is a sequence of symbols, usually a sequence that encodes a number in some form. The initial configuration of the Turing machine is that configuration in which we start the Turing machine to work on that input: the tape contains the tape end marker immediately followed by the input written on the squares to the right, the read/write head is scanning the leftmost square of the input (i.e., the square to the right of the left end marker), and the mechanism is in the designated start state $q_0$.

**Definition 28.8** (Initial configuration)**.** The *initial configuration* of $M$ for input $I \in \Sigma^*$ is

$$\langle \triangleright \frown I, 1, q_0 \rangle$$

The $\frown$ symbol is for *concatenation*—we want to ensure that there are no blanks between the left end marker and the beginning of the input.

**Definition 28.9.** We say that a configuration $\langle C, n, q \rangle$ *yields* $\langle C', n', q' \rangle$ *in one step* (according to $M$), iff

1. the $n$-th symbol of $C$ is $\sigma$,
2. the instruction set of $M$ specifies $\delta(q, \sigma) = \langle q', \sigma', D \rangle$,
3. the $n$-th symbol of $C'$ is $\sigma'$, and
4. a) $D = L$ and $n' = n - 1$ if $n > 0$, otherwise $n' = 0$, or
   b) $D = R$ and $n' = n + 1$, or
   c) $D = N$ and $n' = n$,
5. if $n' > \text{len}(C)$, then $\text{len}(C') = \text{len}(C) + 1$ and the $n'$-th symbol of $C'$ is 0.
6. for all $i$ such that $i < \text{len}(C')$ and $i \neq n$, $C'(i) = C(i)$,

**Definition 28.10.** A *run of $M$ on input $I$* is a sequence $C_i$ of configurations of $M$, where $C_0$ is the initial configuration of $M$ for input $I$, and each $C_i$ yields $C_{i+1}$ in one step.

We say that *$M$ halts on input $I$ after $k$ steps* if $C_k = \langle C, n, q\rangle$, the $n$th symbol of $C$ is $\sigma$, and $\delta(q, \sigma)$ is undefined. In that case, the *output* of $M$ for input $I$ is $O$, where $O$ is a string of symbols not beginning or ending in 0 such that $C = \triangleright \frown 0^i \frown O \frown 0^j$ for some $i, j \in \mathbb{N}$.

According to this definition, the output $O$ of $M$ always begins and ends in a symbol other than 0, or, if at time $k$ the entire tape is filled with 0 (except for the leftmost $\triangleright$), $O$ is the empty string.

## 28.5 Unary Representation of Numbers

Turing machines work on sequences of symbols written on their tape. Depending on the alphabet a Turing machine uses, these sequences of symbols can represent various inputs and outputs. Of particular interest, of course, are Turing machines which compute *arithmetical* functions, i.e., functions of natural numbers. A simple way to represent positive integers is by coding them as sequences of a single symbol 1. If $n \in \mathbb{N}$, let $1^n$ be the empty sequence if $n = 0$, and otherwise the sequence consisting of exactly $n$ 1's.

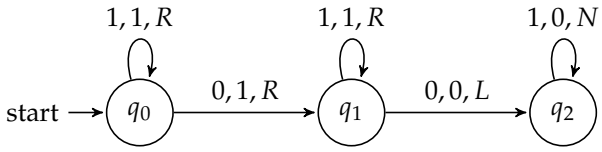
**Definition 28.11** (Computation)**.** A Turing machine $M$ *computes* the function $f : \mathbb{N}^n \to \mathbb{N}$ iff $M$ halts on input

$$1^{k_1}01^{k_2}0 \ldots 01^{k_n}$$

with output $1^{f(k_1,\ldots,k_n)}$.

**Example 28.12.** *Addition:* Build a machine that, when given an input of two non-empty strings of 1's of length $n$ and $m$, computes the function $f(n, m) = n + m$.

We want to come up with a machine that starts with two blocks of strokes on the tape and halts with one block of strokes. We first need a method to carry out. The input strokes are separated by a blank, so one method would be to write a stroke on the square containing the blank, and erase the first (or last) stroke. This would result in a block of $n + m$ 1's. Alternatively, we could proceed in a similar way to the doubler machine, by erasing a stroke from the first block, and adding one to the second block of strokes until the first block has been removed completely. We will proceed with the former example.

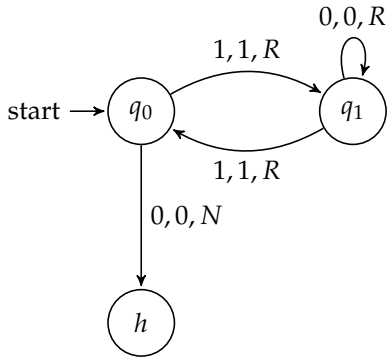


## 28.6 Halting States

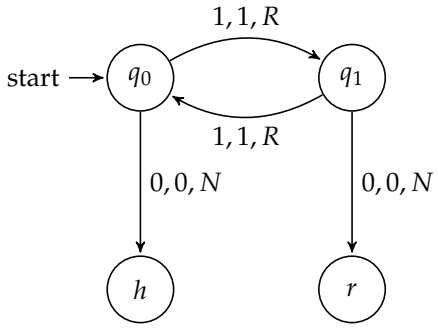
Although we have defined our machines to halt only when there is no instruction to carry out, common representations of Turing machines have a dedicated *halting state*, $h$, such that $h \in Q$.

The idea behind a halting state is simple: when the machine has finished operation (it is ready to accept input, or has finished writing the output), it goes into a state $h$ where it halts. Some machines have two halting states, one that accepts input and one that rejects input.

**Example 28.13.** *Halting States.* To elucidate this concept, let us begin with an alteration of the even machine. Instead of having the machine halt in state $q_0$ if the input is even, we can add an instruction to send the machine into a halt state.




Let us further expand the example. When the machine determines that the input is odd, it never halts. We can alter the machine to include a *reject* state by replacing the looping instruction with an instruction to go to a reject state $r$.




Adding a dedicated halting state can be advantageous in cases like this, where it makes explicit when the machine accepts/rejects certain inputs. However, it is important to note that no computing power is gained by adding a dedicated halting state. Similarly, a less formal notion of halting has its own

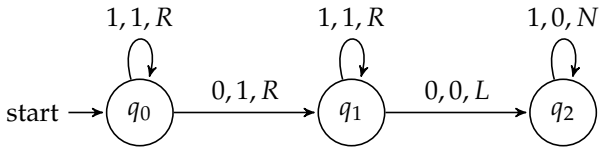
advantages. The definition of halting used so far in this chapter makes the proof of the *Halting Problem* intuitive and easy to demonstrate. For this reason, we continue with our original definition.

## 28.7 Combining Turing Machines

The examples of Turing machines we have seen so far have been fairly simple in nature. But in fact, any problem that can be solved with any modern programming language can als o be solved with Turing machines. To build more complex Turing machines, it is important to convince ourselves that we can combine them, so we can build machines to solve more complex problems by breaking the procedure into simpler parts. If we can find a natural way to break a complex problem down into constituent parts, we can tackle the problem in several stages, creating several simple Turing machines and combining then into one machine that can solve the problem. This point is especially important when tackling the Halting Problem in the next section.

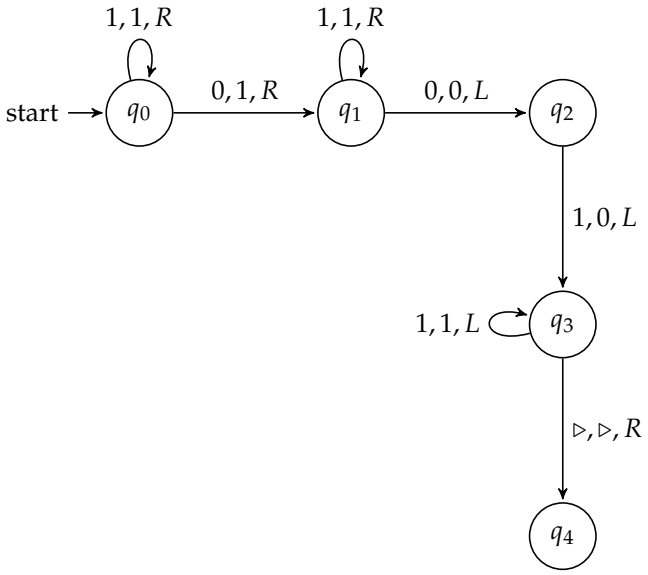
**Example 28.14.** *Combining Machines:* Design a machine that computes the function $f(m, n) = 2(m + n)$.

In order to build this machine, we can combine two machines we are already familiar with: the addition machine, and the doubler. We begin by drawing a state diagram for the addition machine.



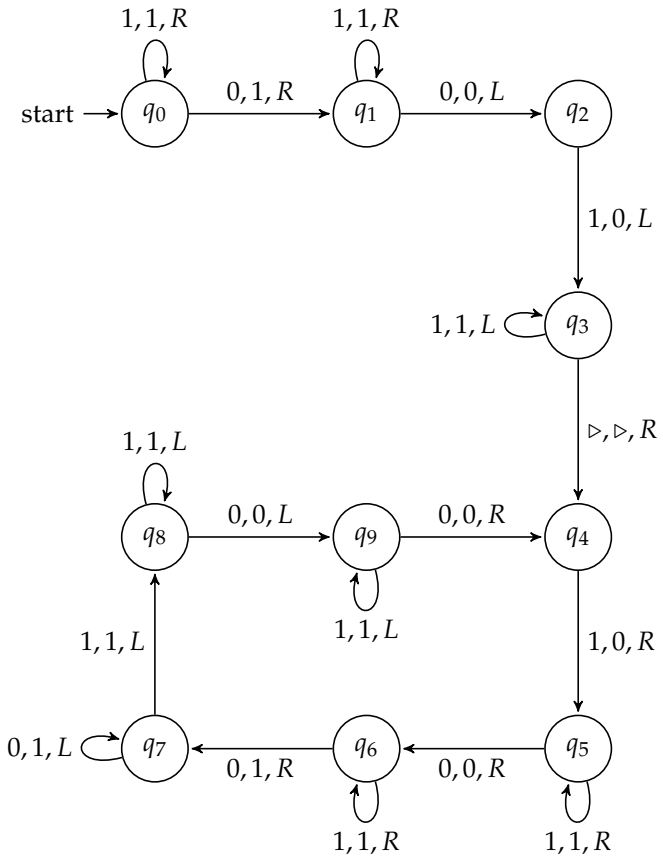

Instead of halting at state $q_2$, we want to continue operation in order to double the output. Recall that the doubler machine erases the first stroke in the input and writes two strokes in a separate output. Let's add an instruction to make sure the tape head is reading the first stroke of the output of the addition

machine.




It is now easy to double the input—all we have to do is connect the doubler machine onto state $q_4$. This requires renaming the states of the doubler machine so that they start at $q_4$ instead of $q_0$—this way we don't end up with two

starting states. The final diagram should look like:




## 28.8 Variants of Turing Machines

There are in fact many possible ways to define Turing machines, of which ours is only one. In some ways, our definition is more liberal than others. We allow arbitrary finite alphabets, a more restricted definition might allow only two tape symbols, 1 and 0. We allow the machine to write a symbol to the tape and move at the same time, other definitions allow either writing or moving. We allow the possibility of writing without moving the tape head, other definitions leave out the $N$ "instruction." In other ways, our definition is more restrictive. We assumed that the tape is infinite in one direction only, other definitions allow the tape to be infinite both to the left and the right. In fact, one can even even allow any number of separate tapes, or even an infinite grid of squares. We represent the instruction set of the Turing machine by a transition function; other definitions use a transition relation where the machine has more than one possible instruction in any given situation.

This last relaxation of the definition is particularly interesting. In our definition, when the machine is in state $q$ reading symbol $\sigma$, $\delta(q, \sigma)$ determines what the new symbol, state, and tape head position is. But if we allow the instruction set to be a relation between current state-symbol pairs $\langle q, \sigma \rangle$ and new state-symbol-direction triples $\langle q', \sigma', D \rangle$, the action of the Turing machine may not be uniquely determined—the instruction relation may contain both $\langle q, \sigma, q', \sigma', D \rangle$ and $\langle q, \sigma, q'', \sigma'', D' \rangle$. In this case we have a *non-deterministic* Turing machine. These play an important role in computational complexity theory.

There are also different conventions for when a Turing machine halts: we say it halts when the transition function is undefined, other definitions require the machine to be in a special designated halting state. Since the tapes of our turing machines are infinite in one direction only, there ae cases where a Turing machine can't properly carry out an instruction: if it reads the leftmost square and is supposed to move left. According to our definition, it just stays put instead, but we could have defined it so that it halts when that happens. There are also different ways of representing numbers (and hence the input-output function computed by a Turing machine): we use unary representation, but you can also use binary representation (this requires two symbols in addition to 0).

Now here is an interesting fact: none of these variations matters as to which functions are Turing computable. *If a function is Turing computable according to one definition, it is Turing computable according to all of them.*

## 28.9 The Church-Turing Thesis

Turing machines are supposed to be a precise replacement for the concept of an effective procedure. Turing took it that anyone who grasped the concept of an effective procedure and the concept of a Turing machine would have the intuition that anything that could be done via an effective procedure could be done by Turing machine. This claim is given support by the fact that all the other proposed precise replacements for the concept of an effective procedure turn out to be extensionally equivalent to the concept of a Turing machine—that is, they can compute exactly the same set of functions. This claim is called the *Church-Turing thesis*.

**Definition 28.15** (Church-Turing thesis)**.** The *Church-Turing Thesis* states that anything computable via an effective procedure is Turing computable.

The Church-Turing thesis is appealed to in two ways. The first kind of use of the Church-Turing thesis is an excuse for laziness. Suppose we have a description of an effective procedure to compute something, say, in "pseudo-code." Then we can invoke the Church-Turing thesis to justify the claim that

the same function is computed by some Turing machine, eve if we have not in fact constructed it.

The other use of the Church-Turing thesis is more philosophically interesting. It can be shown that there are functions whch cannot be computed by a Turing machines. From this, using the Church-Turing thesis, one can conclude that it cannot be effectively computed, using any procedure whatsoever. For if there were such a procedure, by the Church-Turing thesis, it would follow that there would be a Turing machine. So if we can prove that there is no Turing machine that computes it, there also can't be an effective procedure. In particular, the Church-Turing thesis is invoked to claim that the so-called halting problem not only cannot be solved by Turing machines, it cannot be effectively solved at all.

## Problems

**Problem 28.1.** Choose an arbitary input and trace through the configurations of the doubler machine in Example 28.4.

**Problem 28.2.** The double machine in Example 28.4 writes its output to the right of the input. Come up with a new method for solving the doubler problem which generates its output immediately to the right of the end-of-tape marker. Build a machine that executes your method. Check that your machine works by tracing through the configurations.

**Problem 28.3.** Design a Turing-machine with alphabet $\{0, A, B\}$ that accepts any string of $A$s and $B$s where the number of $A$s is the same as the number of $B$s *and* all the $A$s precede all the $B$s, and rejects any string where the number of $A$s is not equal to the number of $B$s or the $A$s do not precede all the $B$s. (E.g., the machine should accept $AABB$, and $AAABBB$, but reject both $AAB$ and $AABBAABB$.)

**Problem 28.4.** Design a Turing-machine with alphabet $\{0, A, B\}$ that takes as input any string $\alpha$ of $A$s and $B$s and duplicates them to produce an output of the form $\alpha\alpha$. (E.g. input $ABBA$ should result in output $ABBAABBA$).

**Problem 28.5.** *Alphabetical?:* Design a Turing-machine with alphabet $\{0, A, B\}$ that when given as input a finite sequence of As and Bs checks to see if all the As appear left of all the Bs or not. The machine should leave the input string on the tape, and output either halt if the string is "alphabetical", or loop forever if the string is not.

**Problem 28.6.** *Alphabetizer:* Design a Turing-machine with alphabet $\{0, A, B\}$ that takes as input a finite sequence of $A$s and $B$s rearranges them so that all the $A$s are to the left of all the $B$s. (e.g., the sequence $BABAA$ should become the sequence $AAABB$, and the sequence $ABBABB$ should become the sequence $AABBBB$).

**Problem 28.7.** Trace through the configurations of the machine for input $\langle 3, 5\rangle$.

**Problem 28.8.** *Subtraction:* Design a Turing machine that when given an input of two non-empty strings of strokes of length $n$ and $m$, where $n > m$, computes the function $f(n, m) = n - m$.

**Problem 28.9.** *Equality:* Design a Turing machine to compute the following function:

$$\text{equality}(x, y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y \end{cases}$$

where $x$ and $y$ are integers greater than 0.

**Problem 28.10.** Design a Turing machine to compute the function $\min(x, y)$ where $x$ and $y$ are positive integers represented on the tape by strings of 1's separated by a 0. You may use additional symbols in the alphabet of the machine.

The function min selects the smallest value from its arguments, so $\min(3, 5) = 3$, $\min(20, 16) = 16$, and $\min(4, 4) = 4$, and so on.

# Chapter 29

# Undecidability

## 29.1 Introduction

It might seem obvious that not every function, even every arithmetical function, can be computable. There are just too many, whose behavior is too complicated. Functions defined from the decay of radioactive particles, for instance, or other chaotic or random behavior. Suppose we start counting 1-second intervals from a given time, and define the function $f(n)$ as the number of particles in the universe that decay in the $n$-th 1-second interval after that initial moment. This seems like a candidate for a function we cannot ever hope to compute.

But it is one thing to not be able to imagine how one would compute such functions, and quite another to actually prove that they are uncomputable. In fact, even functions that seem hopelessly complicated may, in an abstract sense, be computable. For instance, suppose the universe is finite in time—some day, in the very distant future the universe will contract into a single point, as some cosmological theories predict. Then there is only a finite (but incredibly large) number of seconds from that initial moment for which $f(n)$ is defined. And any function which is defined for only finitely many inputs is computable: we could list the outputs in one big table, or code it in one very big Turing machine state transition diagram.

We are often interested in special cases of functions whose values give the answers to yes/no questions. For instance, the question "is $n$ a prime number?" is associated with the function

$$\text{isprime}(n) = \begin{cases} 1 & \text{if } n \text{ is prime} \\ 0 & \text{otherwise.} \end{cases}$$

We say that a yes/no question can be *effectively decided*, if the associated 1/0-valued function is effectively computable.

To prove mathematically that there are functions which cannot be effectively computed, or problems that cannot effectively decided, it is essential to

fix a specific model of computation, and show about it that there are functions it cannot compute or problems it cannot decide. We can show, for instance, that not every function can be computed by Turing machines, and not every problem can be decided by Turing machines. We can then appeal to the Church-Turing thesis to conclude that not only are Turing machines not powerful enough to compute every function, but no effective procedure can.

The key to proving such negative results is the fact that we can assign numbers to Turing machines themselves. The easiest way to do this is to enumerate them, perhaps by fixing a specific way to write down Turing machines and their programs, and then listing them in a systematic fashion. Once we see that this can be done, then the existence of Turing-uncomputable functions follows by simple cardinality considerations: the set of functions from $\mathbb{N}$ to $\mathbb{N}$ (in fact, even just from $\mathbb{N}$ to $\{0, 1\}$) are non-enumerable, but since we can enumerate all the Turing machines, the set of Turing-computable functions is only denumerable.

We can also define *specific* functions and problems which we can prove to be uncomputable and undecidable, respectively. One such problem is the so-called *Halting Problem.* Turing machines can be finitely described by listing their instructions. Such a description of a Turing machine, i.e., a Turing machine program, can of course be used as input to another Turing machine. So we can consider Turing machines that decide questions about other Turing machines. One particularly interesting question is this: "Does the given Turing machine eventually halt when started on input $n$?" It would be nice if there were a Turing machine that could decide this question: think of it as a quality-control Turing machine which ensures that Turing machines don't get caught in infinite loops and such. The interestign fact, which Turing proved, is that there cannot be such a Turing machine. There cannot be a single Turing machine which, when started on input consisting of a description of a Turing machine $M$ and some number $n$, will always halt with either output 1 or 0 according to whether $M$ machine would have halted when started on input $n$ or not.

Once we have examples of specific undecidable problems we can use them to show that other problems are undecidable, too. For instance, one celebrated undecidable problem is the question, "Is the first-order formula $\varphi$ valid?". There is no Turing machine which, given as input a first-order formula $\varphi$, is guaranteed to halt with output 1 or 0 according to whether $\varphi$ is valid or not. Historically, the question of finding a procedure to effectively solve this problem was called simply "the" decision problem; and so we say that the decision problem is unsolvable. Turing and Church proved this result independently at around the same time, so it is also called the Church-Turing Theorem.

## 29.2 Enumerating Turing Machines

We can show that the set of all Turing-machines is enumerable. This follows from the fact that each Turing machine can be finitely described. The set of states and the tape vocabulary are finite sets. The transition function is a partial function from $Q \times \Sigma$ to $Q \times \Sigma \times \{L, R, N\}$, and so likewise can be specified by listing its values for the finitely many argument pairs for which it is defined. Of course, strictly speaking, the states and vocabulary can be anything; but the *behavior* of the Turing machine is independent of which objects serve as states and vocabulary. So we may assume, for instance, that the states and vocabulary symbols are natural numbers, or that the states and vocabulary are all strings of letters and digits.

Suppose we fix a denumerable vocabulary for specifying Turing machines: $\sigma_0 = \triangleright$, $\sigma_1 = 0$, $\sigma_2 = 1$, $\sigma_3$, ..., $R$, $L$, $N$, $q_0$, $q_1$, .... Then any Turing machine can be specified by some finite string of symbols from this alphabet (though not every finite string of symbols specifies a Turing machine). For instance, suppose we have a Turing machine $M = \langle Q, \Sigma, q, \delta \rangle$ where

$$Q = \{q'_0, \ldots, q'_n\} \subseteq \{q_0, q_1, \ldots\} \text{ and}$$
$$\Sigma = \{\triangleright, \sigma'_1, \sigma'_2, \ldots, \sigma'_m\} \subseteq \{\sigma_0, \sigma_1, \ldots\}.$$

We could specify it by the string

$$q'_0 q'_1 \ldots q'_n \triangleright \sigma'_1 \ldots \sigma'_m \triangleright q \triangleright S(\sigma'_0, q'_0) \triangleright \ldots \triangleright S(\sigma'_m, q'_n)$$

where $S(\sigma'_i, q'_j)$ is the string $\sigma'_i q'_j \delta(\sigma'_i, q'_j)$ if $\delta(\sigma'_i, q'_j)$ is defined, and $\sigma'_i q'_j$ otherwise.

**Theorem 29.1.** *There are functions from $\mathbb{N}$ to $\mathbb{N}$ which are not Turing computable.*

*Proof.* We know that the set of finite strings of symbols from a denumerable alphabet is enumerable. This gives us that the set of descriptions of Turing machines, as a subset of the finite strings from the enumerable vocabulary $\{q_0, q_1, \ldots, \triangleright, \sigma_1, \sigma_2, \ldots\}$, is itself enumerable. Since every Turing computable function is computed by some (in fact, many) Turing machines, this means that the set of all Turing computable functions from $\mathbb{N}$ to $\mathbb{N}$ is also enumerable.

On the other hand, the set of all functions from $\mathbb{N}$ to $\mathbb{N}$ is not enumerable. This follows immediately from the fact that not even the set of all functions of one argument from $\mathbb{N}$ to the set $\{0, 1\}$ is enumerable. If all functions were computable by some Turing machine we could enumerate the set of all functions. So there are some functions that are not Turing-computable. □

## 29.3 The Halting Problem

Assume we have fixed some finite descriptions of Turing machines. Using these, we can enumerate Turing machines via their descriptions, say, ordered

by the lexicographic ordering. Each Turing machine thus receives an *index*: its place in the enumeration $M_1$, $M_2$, $M_3$, ... of Turing machine descriptions.

We know that there must be non-Turing-computable functions: the set of Turing machine descriptions—and hence the set of Turing machines—is enumerable, but the set of all functions from $\mathbb{N}$ to $\mathbb{N}$ is not. But we can find specific examples of non-computable function as well. One such function is the halting function.

**Definition 29.2** (Halting function)**.** The *halting function* $h$ is defined as

$$h(e,n) = \begin{cases} 0 & \text{if machine } M_e \text{ does not halt for input } n \\ 1 & \text{if machine } M_e \text{ halts for input } n \end{cases}$$

**Definition 29.3** (Halting problem)**.** The *Halting Problem* is the problem of determining (for any $m$, $w$) whether the Turing machine $M_e$ halts for an input of $n$ strokes.

We show that $h$ is not Turing-computable by showing that a related function, $s$, is not Turing-computable. This proof relies on the fact that anything that can be computed by a Turing machine can be computed using just two symbols: 0 and 1, and the fact that two Turing machines can be hooked together to create a single machine.

**Definition 29.4.** The function $s$ is defined as

$$s(e) = \begin{cases} 0 & \text{if machine } M_e \text{ does not halt for input } e \\ 1 & \text{if machine } M_e \text{ halts for input } e \end{cases}$$

**Lemma 29.5.** *The function s is not Turing computable.*

*Proof.* We suppose, for contradiction, that the function $s$ is Turing-computable. Then there would be a Turing machine $S$ that computes $s$. We may assume, without loss of generality, that when $S$ halts, it does so while scanning the first square. This machine can be "hooked up" to another machine $J$, which halts if it is started on a blank tape (i.e., if it reads 0 in the initial state while scanning the square to the right of the end-of-tape symbol), and otherwise wanders off to the right, never halting. $S \frown J$, the machine created by hooking $S$ to $J$, is a Turing machine, so it is $M_e$ for some $e$ (i.e., it appears somewhere in the enumeration). Start $M_e$ on an input of $e$ 1s. There are two possibilities: either $M_e$ halts or it does not halt.

1. Suppose $M_e$ halts for an input of $e$ 1s. Then $s(e) = 1$. So $S$, when started on $e$, halts with a single 1 as output on the tape. Then $J$ starts with a 1 on the tape. In that case $J$ does not halt. But $M_e$ is the machine $S \frown J$, so it should do exactly what $S$ followed by $J$ would do. So $M_e$ cannot halt for an input of $e$ 1's.

2. Now suppose $M_e$ does not halt for an input of $e$ 1s. Then $s(e) = 0$, and $S$, when started on input $e$, halts with a blank tape. $J$, when started on a blank tape, immediately halts. Again, $M_e$ does what $S$ followed by $J$ would do, so $M_e$ must halt for an input of $e$ 1's.

This shows there cannot be a Turing machine $S$: $s$ is not Turing computable. □

**Theorem 29.6** (Unsolvability of the Halting Problem)**.** *The halting problem is unsolvable, i.e., the function h is not Turing computable.*

*Proof.* Suppose $h$ were Turing computable, say, by a Turing machine $H$. We could use $H$ to build a Turing machine that computes $s$: First, make a copy of the input (separated by a blank). Then move back to the beginning, and run $H$. We can clearly make a machine that does the former, and if $H$ existed, we would be able to "hook it up" to such a modified doubling machine to get a new machine which would determine if $M_e$ halts on input $e$, i.e., computes $s$. But we've already shown that no such machine can exist. Hence, $h$ is also not Turing computable. □

## 29.4 The Decision Problem

We say that first-order logic is *decidable* iff there is an effective method for determining whether or not a given sentence is valid. As it turns out, there is no such method: the problem of deciding validity of first-order sentences is unsolvable.

In order to establish this important negative result, we prove that the decision problem cannot be solved by a Turing machine. That is, we show that there is no Turing machine which, whenever it is started on a tape that contains a first-order sentence, eventually halts and outputs either 1 or 0 depending on whether the sentence is valid or not. By the Church-Turing thesis, every function which is computable is Turing computable. So if if this "validity function" were effectively computable at all, it would be Turing computable. If it isn't Turing computable, then, it also cannot be effectively computable.

Our strategy for proving that the decision problem is unsolvable is to reduce the halting problem to it. This means the following: We have proved that the function $h(e, w)$ that halts with output 1 if the Turing-machine described by $e$ halts on input $w$ and outputs 0 otherwise, is not Turing-computable. We will show that if there were a Turing machine that decides validity of first-order sentences, then there is also Turing machine that computes $h$. Since $h$ cannot be computed by a Turing machine, there cannot be a Turing machine that decides validity either.

The first step in this strategy is to show that for every input $w$ and a Turing machine $M$, we can effectively describe a sentence $\tau(M, w)$ representing the

instruction set of $M$ and the input $w$ and a sentence $\alpha(M, w)$ expressing "$M$ eventually halts" such that:

$$\vDash \tau(M, w) \to \alpha(M, w) \text{ iff } M \text{ halts for input } w.$$

The bulk of our proof will consist in describing these sentences $\tau(M, w)$ and $\alpha(M, w)$ and verifying that $\tau(M, w) \to \alpha(M, w)$ is valid iff $M$ halts on input $w$.

## 29.5 Representing Turing Machines

In order to represent Turing machines and their behavior by a sentence of first-order logic, we have to define a suitable language. The language consists of two parts: predicate symbols for describing configurations of the machine, and expressions for numbering execution steps ("moments") and positions on the tape.

We introduce two kinds of predicate symbols, both of them 2-place: For each state $q$, a predicate symbol $Q_q$, and for each tape symbol $\sigma$, a predicate symbol $S_\sigma$. The former allow us to describe the state of $M$ and the position of its tape head, the latter allow us to describe the contents of the tape.

In order to express the positions of the tape head and the number of steps executed, we need a way to express numbers. This is done using a constant symbol $\mathsf{o}$, and a 1-place function $\prime$, the successor function. By convention it is written *after* its argument (and we leave out the parentheses). So $\mathsf{o}$ names the leftmost position on the tape as well as the time before the first execution step (the initial configuration), $\mathsf{o}'$ names the square to the right of the leftmost square, and the time after the first execution step, and so on. We also introduce a predicate symbol $<$ to express both the ordering of tape positions (when it means "to the left of") and execution steps (then it means "before").

Once we have the language in place, we list the "axioms" of $\tau(M, w)$, i.e., the sentences which, taken together, describe the behavior of $M$ when run on input $w$. There will be sentences which lay down conditions on $\mathsf{o}$, $\prime$, and $<$, sentences that describes the input configuration, and sentences that describe what the configuration of $M$ is after it executes a particular instruction.

**Definition 29.7.** Given a Turing machine $M = \langle Q, \Sigma, q_0, \delta \rangle$, the language $\mathcal{L}_M$ consists of:

1. A two-place predicate symbol $Q_q(x, y)$ for every state $q \in Q$. Intuitively, $Q_q(\overline{m}, \overline{n})$ expresses "after $n$ steps, $M$ is in state $q$ scanning the $m$th square."

2. A two-place predicate symbol $S_\sigma(x, y)$ for every symbol $\sigma \in \Sigma$. Intuitively, $S_\sigma(\overline{m}, \overline{n})$ expresses "after $n$ steps, the $m$th square contains symbol $\sigma$."

3. A constant symbol $\mathsf{o}$

4. A one-place function symbol $'$

5. A two-place predicate symbol $<$

For each number $n$ there is a canonical term $\overline{n}$, the *numeral* for $n$, which represents it in $\mathcal{L}_M$. $\overline{0}$ is $\mathsf{o}$, $\overline{1}$ is $\mathsf{o}'$, $\overline{2}$ is $\mathsf{o}''$, and so on. More formally:

$$\overline{0} = \mathsf{o}$$
$$\overline{n+1} = \overline{n}'$$

The sentences describing the operation of the Turing machine $M$ on input $w = \sigma_{i_1} \dots \sigma_{i_k}$ are the following:

1. Axioms describing numbers:

   a) A sentence that says that the successor function is injective:

   $$\forall x \, \forall y \, (x' = y' \to x = y)$$

   b) A sentence that says that every number is less than its successor:

   $$\forall x \, x < x'$$

   c) A sentence that ensures that $<$ is transitive:

   $$\forall x \, \forall y \, \forall z \, ((x < y \wedge y < z) \to x < z)$$

   d) A sentence that connects $<$ and $=$:

   $$\forall x \, \forall y \, (x < y \to x \neq y)$$

2. Axioms describing the input configuration:

   a) After after 0 steps—before the machine starts—$M$ is in the inital state $q_0$, scanning square 1:

   $$Q_{q_0}(\overline{1}, \overline{0})$$

   b) The first $k+1$ squares contain the symbols $\triangleright, \sigma_{i_1}, \dots, \sigma_{i_k}$:

   $$S_{\triangleright}(\overline{0}, \overline{0}) \wedge S_{\sigma_{i_1}}(\overline{1}, \overline{0}) \wedge \dots \wedge S_{\sigma_{i_k}}(\overline{n}, \overline{0})$$

   c) Otherwise, the tape is empty:

   $$\forall x \, (\overline{k} < x \to S_0(x, \overline{0}))$$

3. Axioms describing the transition from one configuration to the next:

   For the following, let $\varphi(x, y)$ be the conjunction of all sentences of the form

   $$\forall z\,(((z < x \vee x < z) \wedge S_\sigma(z, y)) \rightarrow S_\sigma(z, y'))$$

   where $\sigma \in \Sigma$. We use $\varphi(\overline{m}, \overline{n})$ to express "other than at square $m$, the tape after $n + 1$ steps is the same as after $n$ steps."

   a) For every instruction $\delta(q_i, \sigma) = \langle q_j, \sigma', R\rangle$, the sentence:

   $$\begin{aligned}&\forall x\,\forall y\,((Q_{q_i}(x, y) \wedge S_\sigma(x, y)) \rightarrow \\ &\qquad (Q_{q_j}(x', y') \wedge S_{\sigma'}(x, y') \wedge \varphi(x, y)))\end{aligned}$$

   This says that if, after $y$ steps, the machine is in state $q_i$ scanning square $x$ which contains symbol $\sigma$, then after $y + 1$ steps it is scanning square $x + 1$, is in state $q_j$, square $x$ now contains $\sigma'$, and every square other than $x$ contains the same symbol as it did after $y$ steps.

   b) For every instruction $\delta(q_i, \sigma) = \langle q_j, \sigma', L\rangle$, the sentence:

   $$\begin{aligned}&\forall x\,\forall y\,((Q_{q_i}(x', y) \wedge S_\sigma(x', y)) \rightarrow \\ &\qquad (Q_{q_j}(x, y') \wedge S_{\sigma'}(x', y') \wedge \varphi(x, y))) \wedge \\ &\forall y\,((Q_{q_i}(\mathsf{o}, y) \wedge S_\sigma(\mathsf{o}, y)) \rightarrow \\ &\qquad (Q_{q_j}(\mathsf{o}, y') \wedge S_{\sigma'}(\mathsf{o}, y') \wedge \varphi(\mathsf{o}, y)))\end{aligned}$$

   Take a moment to think about how this works: now we don't start with "if scanning square $x \ldots$" but: "if scanning square $x + 1 \ldots$" A move to the left means that in the next step the machine is scanning square $x$. But the square that is written on is $x + 1$. We do it this way since we don't have subtraction or a predecessor function.

   Note that numbers of the form $x + 1$ are 1, 2, ..., i.e., this doesn't cover the case where the machine is scanning square 0 and is supposed to move left (which of course it can't—it just stays put). That special case is covered by the second conjunction: it says that if, after $y$ steps, the machine is scanning square 0 in state $q_i$ and square 0 contains symbol $\sigma$, then after $y + 1$ steps it's still scanning square 0, is now in state $q_j$, the symbol on square 0 is $\sigma'$, and the squares other than square 0 contain the same symbols they contained ofter $y$ steps.

   c) For every instruction $\delta(q_i, \sigma) = \langle q_j, \sigma', N\rangle$, the sentence:

   $$\begin{aligned}&\forall x\,\forall y\,((Q_{q_i}(x, y) \wedge S_\sigma(x, y)) \rightarrow \\ &\qquad (Q_{q_j}(x, y') \wedge S_{\sigma'}(x, y') \wedge \varphi(x, y)))\end{aligned}$$

Let $\tau(M, w)$ be the conjunction of all the above sentences for Turing machine $M$ and input $w$

In order to express that $M$ eventually halts, we have to find a sentence that says "after some number of steps, the transition function will be undefined." Let $X$ be the set of all pairs $\langle q, \sigma \rangle$ such that $\delta(q, \sigma)$ is undefined. Let $\alpha(M, w)$ then be the sentence

$$\exists x \, \exists y \, ( \bigvee_{\langle q,\sigma \rangle \in X} (Q_q(x, y) \wedge S_\sigma(x, y)))$$

If we use a Turing machine with a designated halting state $h$, it is even easier: then the sentence $\alpha(M, w)$

$$\exists x \, \exists y \, Q_h(x, y)$$

expresses that the machine eventually halts.

**Proposition 29.8.** *If $m < k$, then $\tau(M, w) \vDash \overline{m} < \overline{k}$*

*Proof.* Exercise. □

## 29.6 Verifying the Representation

In order to verify that our representation works, we have to prove two things. First, we have to show that if $M$ halts on input $w$, then $\tau(M, w) \to \alpha(M, w)$ is valid. Then, we have to show the converse, i.e., that if $\tau(M, w) \to \alpha(M, w)$ is valid, then $M$ does in fact eventually halt when run on input $w$.

The strategy for proving these is very different. For the first result, we have to show that a sentence of first-order logic (namely, $\tau(M, w) \to \alpha(M, w)$) is valid. The easiest way to do this is to give a derivation. Our proof is supposed to work for all $M$ and $w$, though, so there isn't really a single sentence for which we have to give a derivation, but infinitely many. So the best we can do is to prove by induction that, whatever $M$ and $w$ look like, and however many steps it takes $M$ to halt on input $w$, there will be a derivation of $\tau(M, w) \to \alpha(M, w)$.

Naturally, our induction will proceed on the number of steps $M$ takes before it reaches a halting configuration. In our inductive proof, we'll establish that for each step $n$ of the run of $M$ on input $w$, $\tau(M, w) \vDash \chi(M, w, n)$, where $\chi(M, w, n)$ correctly describes the configuration of $M$ run on $w$ after $n$ steps. Now if $M$ halts on input $w$ after, say, $n$ steps, $\chi(M, w, n)$ will describe a halting configuration. We'll also show that $\chi(M, w, n) \vDash \alpha(M, w)$, whenever $\chi(M, w, n)$ describes a halting configuration. So, if $M$ halts on input $w$, then for some $n$, $M$ will be in a halting configuration after $n$ steps. Hence, $\tau(M, w) \vDash \chi(M, w, n)$ where $\chi(M, w, n)$ describes a halting configuration, and since in that case $\chi(M, w, n) \vDash \alpha(M, w)$, we get that $T(M, w) \vDash \alpha(M, w)$, i.e., that $\vDash \tau(M, w) \to \alpha(M, w)$.

The strategy for the converse is very different. Here we assume that $\models \tau(M,w) \to \alpha(M,w)$ and have to prove that $M$ halts on input $w$. From the hypothesis we get that $\tau(M,w) \models \alpha(M,w)$, i.e., $\alpha(M,w)$ is true in every structure in which $\tau(M,w)$ is true. So we'll describe a structure $\mathfrak{M}$ in which $\tau(M,w)$ is true: its domain will be $\mathbb{N}$, and the interpretation of all the $Q_q$ and $S_\sigma$ will be given by the configurations of $M$ during a run on input $w$. So, e.g., $\mathfrak{M} \models Q_q(\overline{m},\overline{n})$ iff $T$, when run on input $w$ for $n$ steps, is in state $q$ and scanning square $m$. Now since $\tau(M,w) \models \alpha(M,w)$ by hypothesis, and since $\mathfrak{M} \models \tau(M,w)$ by construction, $\mathfrak{M} \models \alpha(M,w)$. But $\mathfrak{M} \models \alpha(M,w)$ iff there is some $n \in |\mathfrak{M}| = \mathbb{N}$ so that $M$, run on input $w$, is in a halting configuration after $n$ steps.

**Definition 29.9.** Let $\chi(M,w,n)$ be the sentence

$$Q_q(\overline{m},\overline{n}) \wedge S_{\sigma_0}(\overline{0},\overline{n}) \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}) \wedge \forall x\,(\overline{k} < x \to S_0(x,\overline{n}))$$

where $q$ is the state of $M$ at time $n$, $M$ is scanning square $m$ at time $n$, square $i$ contains symbol $\sigma_i$ at time $n$ for $0 \leq i \leq k$ and $k$ is the right-most non-blank square of the tape at time 0, or the right-most square the tape head has visited after $n$ steps, whichever is greater.

**Lemma 29.10.** *If $M$ run on input $w$ is in a halting configuration after $n$ steps, then $\chi(M,w,n) \models \alpha(M,w)$.*

*Proof.* Suppose that $M$ halts for input $w$ after $n$ steps. There is some state $q$, square $m$, and symbol $\sigma$ such that:

1. After $n$ steps, $M$ is in state $q$ scanning square $m$ on which $\sigma$ appears.
2. The transition function $\delta(q,\sigma)$ is undefined.

$\chi(M,w,n)$ is the description of this configuration and will include the clauses $Q_q(\overline{m},\overline{n})$ and $S_\sigma(\overline{m},\overline{n})$. These clauses together imply $\alpha(M,w)$:

$$\exists x\,\exists y\,(\bigvee_{\langle q,\sigma\rangle \in X} (Q_q(x,y) \wedge S_\sigma(x,y)))$$

since $Q_{q'}(\overline{m},\overline{n}) \wedge S_{\sigma'}(\overline{m},\overline{n}) \models \bigvee_{\langle q,\sigma\rangle \in X}(Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n}))$, as $\langle q',\sigma'\rangle \in X$. □

So if $M$ halts for input $w$, then there is some $n$ such that $\chi(M,w,n) \models \alpha(M,w)$. We will now show that for any time $n$, $\tau(M,w) \models \chi(M,w,n)$.

**Lemma 29.11.** *For each $n$, if $M$ has not halted after $n$ steps, $\tau(M,w) \models \chi(M,w,n)$.*

*Proof.* Induction basis: If $n = 0$, then the conjuncts of $\chi(M,w,0)$ are also conjuncts of $\tau(M,w)$, so entailed by it.

Inductive hypothesis: If $M$ has not halted before the $n$th step, then $\tau(M,w) \vDash \chi(M,w,n)$. We have to show that (unless $\chi(M,w,n)$ describes a halting configuration), $\tau(M,w) \vDash \chi(M,w,n+1)$.

Suppose $n > 0$ and after $n$ steps, $M$ started on $w$ is in state $q$ scanning square $m$. Since $M$ does not halt after $n$ steps, there must be an instruction of one of the following three forms in the program of $M$:

1. $\delta(q,\sigma) = \langle q', \sigma', R\rangle$
2. $\delta(q,\sigma) = \langle q', \sigma', L\rangle$
3. $\delta(q,\sigma) = \langle q', \sigma', N\rangle$

We will consider each of these three cases in turn.

1. Suppose there is an instruction of the form (1). By Definition 29.7, (3a), this means that

$$\begin{aligned}&\forall x\, \forall y\, ((Q_q(x,y) \wedge S_\sigma(x,y)) \rightarrow \\ &\qquad (Q_{q'}(x',y') \wedge S_{\sigma'}(x,y') \wedge \varphi(x,y)))\end{aligned}$$

   is a conjunct of $\tau(M,w)$. This entails the following sentence (universal instantiation, $\overline{m}$ for $x$ and $\overline{n}$ for $y$):

$$\begin{aligned}&(Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n})) \rightarrow \\ &\qquad (Q_{q'}(\overline{m}',\overline{n}') \wedge S_{\sigma'}(\overline{m},\overline{n}') \wedge \varphi(\overline{m},\overline{n})).\end{aligned}$$

   By induction hypothesis, $\tau(M,w) \vDash \chi(M,w,n)$, i.e.,

$$Q_q(\overline{m},\overline{n}) \wedge S_{\sigma_0}(\overline{0},\overline{n}) \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}) \wedge \forall x\, (\overline{k} < x \rightarrow S_0(x,\overline{n}))$$

   Since after $n$ steps, tape square $m$ contains $\sigma$, the corresponding conjunct is $S_\sigma(\overline{m},\overline{n})$, so this entails:

$$Q_q(\overline{m},\overline{n}) \wedge S_\sigma(\overline{m},\overline{n}))$$

   We now get

$$\begin{aligned}&Q_{q'}(\overline{m}',\overline{n}') \wedge S_{\sigma'}(\overline{m},\overline{n}') \wedge \\ &\qquad S_{\sigma_0}(\overline{0},\overline{n}') \wedge \cdots \wedge S_{\sigma_k}(\overline{k},\overline{n}') \wedge \\ &\qquad \forall x\, (\overline{k} < x \rightarrow S_0(x,\overline{n}'))\end{aligned}$$

   as follows: The first line comes directly from the consequent of the preceding conditional, by modus ponens. Each conjunct in the middle

line—which excludes $S_{\sigma_m}(\overline{m}, \overline{n}')$—follows from the corresponding conjunct in $\chi(M, w, n)$ together with $\varphi(\overline{m}, \overline{n})$.

If $m < k$, $\tau(M, w) \vdash \overline{m} < \overline{k}$ (Proposition 29.8) and by transitivity of $<$, we have $\forall x\,(\overline{k} < x \to \overline{m} < x)$. If $m = k$, then $\forall x\,(\overline{k} < x \to \overline{m} < x)$ by logic alone. The last line then follows from the corresponding conjunct in $\chi(M, w, n)$, $\forall x\,(\overline{k} < x \to \overline{m} < x)$, and $\varphi(\overline{m}, \overline{n})$. If $m < k$, this already is $\chi(M, w, n+1)$.

Now suppose $m = k$. In that case, after $n+1$ steps, the tape head has also visited square $k+1$, which now is the right-most square visited. So $\chi(M, w, n+1)$ has a new conjunct, $S_0(\overline{k}', \overline{n}')$, and the last conjuct is $\forall x\,(\overline{k}' < x \to S_0(x, \overline{n}'))$. We have to verify that these two sentences are also implied.

We already have $\forall x\,(\overline{k} < x \to S_0(x, \overline{n}'))$. In particular, this gives us $\overline{k} < \overline{k'} \to S_0(\overline{k}', \overline{n}')$. From the axiom $\forall x\, x < x'$ we get $\overline{k} < \overline{k'}$. By modus ponens, $S_0(\overline{k}', \overline{n}')$ follows.

Also, since $\tau(M, w) \vdash \overline{k} < \overline{k}'$, the axiom for transitivity of $<$ gives us $\forall x\,(\overline{k}' < x \to S_0(x, \overline{n}'))$. (We leave the verification of this as an exercise.)

2. Suppose there is an instruction of the form (2). Then, by Definition 29.7, (3b),

$$\begin{aligned}
&\forall x\,\forall y\,((Q_q(x', y) \wedge S_\sigma(x', y)) \to \\
&\qquad (Q_{q'}(x, y') \wedge S_{\sigma'}(x', y') \wedge \varphi(x, y))) \wedge \\
&\forall y\,((Q_{q_i}(\mathsf{o}, y) \wedge S_\sigma(\mathsf{o}, y)) \to \\
&\qquad (Q_{q_j}(\mathsf{o}, y') \wedge S_{\sigma'}(\mathsf{o}, y') \wedge \varphi(\mathsf{o}, y)))
\end{aligned}$$

is a conjunct of $\tau(M, w)$. If $m > 0$, then let $l = m - 1$ (i.e., $m = l + 1$). The first conjunct of the above sentence entails the following:

$$\begin{aligned}
&(Q_q(\overline{l}', \overline{n}) \wedge S_\sigma(\overline{l}', \overline{n})) \to \\
&\qquad (Q_{q'}(\overline{l}, \overline{n}') \wedge S_{\sigma'}(\overline{l}', \overline{n}') \wedge \varphi(\overline{l}, \overline{n}))
\end{aligned}$$

Otherwise, let $l = m = 0$ and consider the following sentence entailed by the second conjunct:

$$\begin{aligned}
&((Q_{q_i}(\mathsf{o}, \overline{n}) \wedge S_\sigma(\mathsf{o}, \overline{n})) \to \\
&\qquad (Q_{q_j}(\mathsf{o}, \overline{n}') \wedge S_{\sigma'}(\mathsf{o}, \overline{n}') \wedge \varphi(\mathsf{o}, \overline{n})))
\end{aligned}$$

Either sentence implies

$$\begin{aligned} &Q_{q'}(\bar{l}, \overline{n}') \wedge S_{\sigma'}(\overline{m}, \overline{n}') \wedge \\ &\quad S_{\sigma_0}(\bar{0}, \overline{n}') \wedge \cdots \wedge S_{\sigma_k}(\bar{k}, \overline{n}') \wedge \\ &\quad \forall x\, (\bar{k} < x \to S_0(x, \overline{n}')) \end{aligned}$$

as before. (Note that in the first case, $\bar{l}' = \overline{m}$ and in the second case $\bar{l} = 0$.) But this just is $\chi(M, w, n+1)$.

3. Case (3) is left as an exercise.

We have shown that for any $n$, $\tau(M, w) \vDash \chi(M, w, n)$. □

**Lemma 29.12.** *If $M$ halts on input $w$, then $\tau(M, w) \to \alpha(M, w)$ is valid.*

*Proof.* By Lemma 29.11, we know that, for any time $n$, the description $\chi(M, w, n)$ of the configuration of $M$ at time $n$ is entailed by $\tau(M, w)$. Suppose $M$ halts after $k$ steps. It will be scanning square $m$, say. Then $\chi(M, w, k)$ describes a halting configuration of $M$, i.e., it contains as conjuncts both $Q_q(\overline{m}, \bar{k})$ and $S_\sigma(\overline{m}, \bar{k})$ with $\delta(q, \sigma)$ undefined. By Lemma 29.10 Thus, $\chi(M, w, k) \vDash \alpha(M, w)$. But since $(M, w) \vDash \chi(M, w, k)$, we have $\tau(M, w) \vDash \alpha(M, w)$ and therefore $\tau(M, w) \to \alpha(M, w)$ is valid. □

To complete the verification of our claim, we also have to establish the reverse direction: if $\tau(M, w) \to \alpha(M, w)$ is valid, then $M$ does in fact halt when started on input $m$.

**Lemma 29.13.** *If $\vDash \tau(M, w) \to \alpha(M, w)$, then $M$ halts on input $w$.*

*Proof.* Consider the $\mathcal{L}_M$-structure $\mathfrak{M}$ with domain $\mathbb{N}$ which interprets $0$ as $0$, $'$ as the successor function, and $<$ as the less-than relation, and the predicates $Q_q$ and $S_\sigma$ as follows:

$$Q_q^{\mathfrak{M}} = \{\langle m, n\rangle : \begin{array}{l}\text{started on } w, \text{ after } n \text{ steps,} \\ M \text{ is in state } q \text{ scanning square } m\end{array}\}$$

$$S_\sigma^{\mathfrak{M}} = \{\langle m, n\rangle : \begin{array}{l}\text{started on } w, \text{ after } n \text{ steps,} \\ \text{square } m \text{ of } M \text{ contains symbol } \sigma\end{array}\}$$

In other words, we construct the structure $\mathfrak{M}$ so that it describes what $M$ started on input $w$ actually does, step by step. Clearly, $\mathfrak{M} \vDash \tau(M, w)$. If $\vDash \tau(M, w) \to \alpha(M, w)$, then also $\mathfrak{M} \vDash \alpha(M, w)$, i.e.,

$$\mathfrak{M} \vDash \exists x\, \exists y\, (\bigvee_{\langle q, \sigma\rangle \in X} (Q_q(x, y) \wedge S_\sigma(x, y))).$$

As $|\mathfrak{M}| = \mathbb{N}$, there must be $m, n \in \mathbb{N}$ so that $\mathfrak{M} \vDash Q_q(\overline{m}, \overline{n}) \wedge S_\sigma(\overline{m}, \overline{n})$ for some $q$ and $\sigma$ such that $\delta(q, \sigma)$ is undefined. By the definition of $\mathfrak{M}$, this means that $M$ started on input $w$ after $n$ steps is in state $q$ and reading symbol $\sigma$, and the transition function is undefined, i.e., $M$ has halted. □

## 29.7 The Decision Problem is Unsolvable

**Theorem 29.14.** *The decision problem is unsolvable.*

*Proof.* Suppose the decision problem were solvable, i.e., suppose there were a Turing machine $D$ of the following sort. Whenever $D$ is started on a tape that contains a sentence $\psi$ of first-order logic as input, $D$ eventually halts, and outputs 1 iff $\psi$ is valid and 0 otherwise. Then we could solve the halting problem as follows. We construct a Turing machine $E$ that, given as input the number $e$ of Turing machine $M_e$ and input $w$, computes the corresponding sentence $\tau(M_e, w) \to \alpha(M_e, w)$ and halts, scanning the leftmost square on the tape. The machine $E \frown D$ would then, given input $e$ and $w$, first compute $\tau(M_e, w) \to \alpha(M_e, w)$ and then run the decision problem machine $D$ on that input. $D$ halts with output 1 iff $\tau(M_e, w) \to \alpha(M_e, w)$ is valid and outputs 0 otherwise. By Lemma 29.13 and Lemma 29.12, $\tau(M_e, w) \to \alpha(M_e, w)$ is valid iff $M_e$ halts on input $w$. Thus, $E \frown D$, given input $e$ and $w$ halts with output 1 iff $M_e$ halts on input $w$ and halts with output 0 otherwise. In other words, $E \frown D$ would solve the halting problem. But we know, by Theorem 29.6, that no such Turing machine can exist. □

## Problems

**Problem 29.1.** The Three Halting (3-Halt) problem is the problem of giving a decision procedure to determine whether or not an arbitrarily chosen Turing Machine halts for an input of three strokes on an otherwise blank tape. Prove that the 3-Halt problem is unsolvable.

**Problem 29.2.** Show that if the halting problem is solvable for Turing machine and input pairs $M_e$ and $n$ where $e \neq n$, then it is also solvable for the cases where $e = n$.

**Problem 29.3.** We proved that the halting problem is unsolvable if the input is a number $e$, which identifies a Turing machine $M_e$ via an enumaration of all Turing machines. What if we allow the description of Turing machines from section 29.2 directly as input? (This would require a larger alphabet of course.) Can there be a Turing machine which decides the halting problem but takes as input descriptions of Turing machines rather than indices? Explain why or why not.

**Problem 29.4.** Prove Proposition 29.8. (Hint: use induction on $k - m$).

**Problem 29.5.** Complete case (3) of the proof of Lemma 29.11.

**Problem 29.6.** Give a derivation of $S_{\sigma_i}(\bar{i}, \overline{n}')$ from $S_{\sigma_i}(\bar{i}, \overline{n})$ and $\varphi(m, n)$ (assuming $i \neq m$, i.e., either $i < m$ or $m < i$).

**Problem 29.7.** Give a derivation of $\forall x\,(\overline{k}' < x \to S_0(x, \overline{n}'))$ from $\forall x\,(\overline{k} < x \to S_0(x, \overline{n}'))$, $\forall x\, x < x'$, and $\forall x\,\forall y\,\forall z\,((x < y \wedge y < z) \to x < z)$.)

# Part VII

# Incompleteness

Material in this part covers the incompleteness theorems. It depends on material in the parts on first-order logic (esp., the proof system), the material on recursive functions (in the computability part). It is based on Jeremy Avigad's notes with revisions by Richard Zach.

# Chapter 30

# Introduction to Incompleteness

## 30.1 Historical Background

In this section, we will briefly discuss historical developments that will help put the incompleteness theorems in context. In particular, we will give a very sketchy overview of the history of mathematical logic; and then say a few words about the history of the foundations of mathematics.

The phrase "mathematical logic" is ambiguous. One can interpret the word "mathematical" as describing the subject matter, as in, "the logic of mathematics," denoting the principles of mathematical reasoning; or as describing the methods, as in "the mathematics of logic," denoting a mathematical study of the principles of reasoning. The account that follows involves mathematical logic in both senses, often at the same time.

The study of logic began, essentially, with Aristotle, who lived approximately 384–322 BCE. His *Categories*, *Prior analytics*, and *Posterior analytics* include systematic studies of the principles of scientific reasoning, including a thorough and systematic study of the syllogism.

Aristotle's logic dominated scholastic philosophy through the middle ages; indeed, as late as eighteenth century Kant maintained that Aristotle's logic was perfect and in no need of revision. But the theory of the syllogism is far too limited to model anything but the most superficial aspects of mathematical reasoning. A century earlier, Leibniz, a contemporary of Newton's, imagined a complete "calculus" for logical reasoning, and made some rudimentary steps towards designing such a calculus, essentially describing a version of propositional logic.

The nineteenth century was a watershed for logic. In 1854 George Boole wrote *The Laws of Thought*, with a thorough algebraic study of propositional logic that is not far from modern presentations. In 1879 Gottlob Frege published his *Begriffsschrift* (Concept writing) which extends propositional logic with quantifiers and relations, and thus includes first-order logic. In fact, Frege's logical systems included higher-order logic as well, and more. In his

*Basic Laws of Arithmetic*, Frege set out to show that all of arithmetic could be derived in his Begriffsschrift from purely logical assumption. Unfortunately, these assumptions turned out to be inconsistent, as Russell showed in 1902. But setting aside the inconsistent axiom, Frege more or less invented modern logic singlehandedly, a startling achievement. Quantificational logic was also developed independently by algebraically-minded thinkers after Boole, including Peirce and Schröder.

Let us now turn to developments in the foundations of mathematics. Of course, since logic plays an important role in mathematics, there is a good deal of interaction with the developments I just described. For example, Frege developed his logic with the explicit purpose of showing that all of mathematics could be based solely on his logical framework; in particular, he wished to show that mathematics consists of a priori *analytic* truths instead of, as Kant had maintained, a priori *synthetic* ones.

Many take the birth of mathematics proper to have occurred with the Greeks. Euclid's *Elements*, written around 300 B.C., is already a mature representative of Greek mathematics, with its emphasis on rigor and precision. The definitions and proofs in Euclid's *Elements* survive more or less in tact in high school geometry textbooks today (to the extent that geometry is still taught in high schools). This model of mathematical reasoning has been held to be a paradigm for rigorous argumentation not only in mathematics but in branches of philosophy as well. (Spinoza even presented moral and religious arguments in the Euclidean style, which is strange to see!)

Calculus was invented by Newton and Leibniz in the seventeenth century. (A fierce priority dispute raged for centuries, but most scholars today hold that the two developments were for the most part independent.) Calculus involves reasoning about, for example, infinite sums of infinitely small quantities; these features fueled criticism by Bishop Berkeley, who argued that belief in God was no less rational than the mathematics of his time. The methods of calculus were widely used in the eighteenth century, for example by Leonhard Euler, who used calculations involving infinite sums with dramatic results.

In the nineteenth century, mathematicians tried to address Berkeley's criticisms by putting calculus on a firmer foundation. Efforts by Cauchy, Weierstrass, Bolzano, and others led to our contemporary definitions of limits, continuity, differentiation, and integration in terms of "epsilons and deltas," in other words, devoid of any reference to infinitesimals. Later in the century, mathematicians tried to push further, and explain all aspects of calculus, including the real numbers themselves, in terms of the natural numbers. (Kronecker: "God created the whole numbers, all else is the work of man.") In 1872, Dedekind wrote "Continuity and the irrational numbers," where he showed how to "construct" the real numbers as sets of rational numbers (which, as you know, can be viewed as pairs of natural numbers); in 1888 he wrote "Was sind und was sollen die Zahlen" (roughly, "What are the natural num-

bers, and what should they be?") which aimed to explain the natural numbers in purely "logical" terms. In 1887 Kronecker wrote "Über den Zahlbegriff" ("On the concept of number") where he spoke of representing all mathematical object in terms of the integers; in 1889 Giuseppe Peano gave formal, symbolic axioms for the natural numbers.

The end of the nineteenth century also brought a new boldness in dealing with the infinite. Before then, infinitary objects and structures (like the set of natural numbers) were treated gingerly; "infinitely many" was understood as "as many as you want," and "approaches in the limit" was understood as "gets as close as you want." But Georg Cantor showed that it was possible to take the infinite at face value. Work by Cantor, Dedekind, and others help to introduce the general set-theoretic understanding of mathematics that is now widely accepted.

This brings us to twentieth century developments in logic and foundations. In 1902 Russell discovered the paradox in Frege's logical system. In 1904 Zermelo proved Cantor's well-ordering principle, using the so-called "axiom of choice"; the legitimacy of this axiom prompted a good deal of debate. Between 1910 and 1913 the three volumes of Russell and Whitehead's *Principia Mathematica* appeared, extending the Fregean program of establishing mathematics on logical grounds. Unfortunately, Russell and Whitehead were forced to adopt two principles that seemed hard to justify as purely logical: an axiom of infinity and an axiom of "reducibility." In the 1900's Poincaré criticized the use of "impredicative definitions" in mathematics, and in the 1910's Brouwer began proposing to refound all of mathematics in an "intuitionistic" basis, which avoided the use of the law of the excluded middle ($\varphi \vee \neg\varphi$).

Strange days indeed! The program of reducing all of mathematics to logic is now referred to as "logicism," and is commonly viewed as having failed, due to the difficulties mentioned above. The program of developing mathematics in terms of intuitionistic mental constructions is called "intuitionism," and is viewed as posing overly severe restrictions on everyday mathematics. Around the turn of the century, David Hilbert, one of the most influential mathematicians of all time, was a strong supporter of the new, abstract methods introduced by Cantor and Dedekind: "no one will drive us from the paradise that Cantor has created for us." At the same time, he was sensitive to foundational criticisms of these new methods (oddly enough, now called "classical"). He proposed a way of having one's cake and eating it too:

1. Represent classical methods with formal axioms and rules; represent mathematical questions as formulas in an axiomatic system.

2. Use safe, "finitary" methods to prove that these formal deductive systems are consistent.

Hilbert's work went a long way toward accomplishing the first goal. In 1899, he had done this for geometry in his celebrated book *Foundations of ge-*

*ometry*. In subsequent years, he and a number of his students and collaborators worked on other areas of mathematics to do what Hilbert had done for geometry. Hilbert himself gave axiom systems for arithmetic and analysis. Zermelo gave an axiomatization of set theory, which was expanded on by Fraenkel, Skolem, von Neumann, and others. By the mid-1920s, there were two approaches that laid claim to the title of an axiomatization of "all" of mathematics, the *Principia mathematica* of Russell and Whitehead, and what came to be known as Zermelo-Fraenkel set theory.

In 1921, Hilbert set out on a research project to establish the goal of proving these systems to be consistent. He was aided in this project by several of his students, in particular Bernays, Ackermann, and later Gentzen. The basic idea for accomplishing this goal was to cast the question of the possibility of a derivation of an inconsistency in mathmatics as a combinatorial problem about possible sequences of symbols, namely possible sequences of sentences which meet the criterion of being a correct derivation of, say, $\varphi \wedge \neg\varphi$ from the axioms of an axiom system for arithmetic, analysis, or set theory. A proof of the impossibility of such a sequence of symbols would—since it is itself a mathematical proof—be formalizable in these axiomatic systems. In other words, there would be some sentence $\mathsf{Con}$ which states that, say, arithmetic is consistent. Moreover, this sentence should be provable in the systems in question, especially if its proof requires only very restricted, "finitary" means.

The second aim, that the axiom systems developed would settle every mathematical question, can be made precise in two ways. In one way, we can formulate it as follows: For any sentence $\varphi$ in the language of an axiom system for mathematics, either $\varphi$ or $\neg\varphi$ is provable from the axioms. If this were true, then there would be no sentences which can neither be proved nor refuted on the basis of the axioms, no questions which the axioms do not settle. An axiom system with this property is called *complete*. Of course, for any given sentence it might still be a difficult task to determine which of the two alternatives holds. But in principle there should be a method to do so. In fact, for the axiom and derivation systems considered by Hilbert, completeness would imply that such a method exists—although Hilbert did not realize this. The second way to interpret the question would be this stronger requirement: that there be a mechanical, computational method which would determine, for a given sentence $\varphi$, whether it is derivable from the axioms or not.

In 1931, Gödel proved the two "incompleteness theorems," which showed that this program could not succeed. There is no axiom system for mathematics which is complete, specifically, the sentence that expresses the consistency of the axioms is a sentence which can neither be proved nor refuted.

This struck a lethal blow to Hilbert's original program. However, as is so often the case in mathematics, it also opened up exciting new avenues for research. If there is no one, all-encompassing formal system of mathematics, it makes sense to develop more circumscribesd systems and investigate what

can be proved in them. It also makes sense to develop less restricted methods of proof for establishing the consistency of these systems, and to find ways to measure how hard it is to prove their consistency. Since Gödel showed that (almost) every formal system has questions it cannot settle, it makes sense to look for "interesting" questions a given formal system cannot settle, and to figure out how strong a formal system has to be to settle them. To the present day, logicians have been pursuing these questions in a new mathematical discipline, the theory of proofs.

## 30.2 Definitions

In order to carry out Hilbert's project of formalizing mathematics and showing that such a formalization is consistent and complete, the first order of business would be that of picking a language, logical framework, and a system of axioms. For our purposes, let us suppose that mathematics can be formalized in a first-order language, i.e., that there is some set of constant symbols, function symbols, and predicate symbols which, together with the connectives and quatifiers of first-order logic, allow us to express the claims of mathematics. Most people agree that such a language exists: the language of set theory, in which $\in$ is the only non-logical symbol. That such a simple language is so expressive is of course a very implausible claim at first sight, and it took a lot of work to establish that practically of all mathematics can be expressed in this very austere vocabulary. To keep things simple, for now, let's restrict our discussion to arithmetic, so the part of mathematics that just deals with the natural numbers $\mathbb{N}$. The natural language in which to express facts of arithmetic is $\mathcal{L}_A$. $\mathcal{L}_A$ contains a single two-place predicate symbol $<$, a single constant symbol $\mathsf{0}$, one one-place function symbol $\prime$, and two two-place function symbols $+$ and $\times$.

**Definition 30.1.** A set of sentences $\Gamma$ is a *theory* if it is closed under entailment, i.e., if $\Gamma = \{\varphi : \Gamma \vDash \varphi\}$.

There are two easy ways to specify theories. One is as the set of sentences true in some structure. For instance, consider the structure for $\mathcal{L}_A$ in which the domain is $\mathbb{N}$ and all non-logical symbols are interpreted as you would expect.

**Definition 30.2.** The *standard model of arithmetic* is the structure $\mathfrak{N}$ defined as follows:

1. $|\mathfrak{N}| = \mathbb{N}$
2. $\mathsf{0}^{\mathfrak{N}} = 0$
3. $\prime^{\mathfrak{N}}(n) = n + 1$ for all $n \in \mathbb{N}$

4. $+^{\mathfrak{N}}(n, m) = n + m$ for all $n, m \in \mathbb{N}$
5. $\times^{\mathfrak{N}}(n, m) = n \cdot m$ for all $n, m \in \mathbb{N}$
6. $<^{\mathfrak{N}} = \{\langle n, m\rangle : n \in \mathbb{N}, m \in \mathbb{N}, n < m\}$

Note the difference between $\times$ and $\cdot$: $\times$ is a symbol in the language of arithmetic. Of course, we've chosen it to remind us of multiplication, but $\times$ is not the multiplication operation but a two-place function symbol (officially, $f^2_1$. By contrast, $\cdot$ *is* the ordinary multiplication function. When you see something like $n \cdot m$, we mean the product of the numbers $n$ and $m$; when you see something like $x \times y$ we are talking about a term in the language of arithmetic. In the standard model, the function symbol times is interpreted as the function $\cdot$ on the natural numbers. For addition, we use $+$ as both the function symbol of the language of arithmetic, and the addition function on the natural numbers. Here you have to use the context to determine what is meant.

**Definition 30.3.** The theory of *true arithmetic* is the set of sentences satisfied in the standard model of arithmetic, i.e.,

$$\mathbf{TA} = \{\varphi : \mathfrak{N} \vDash \varphi\}.$$

**TA** is a theory, for whenever $\mathbf{TA} \vDash \varphi$, $\varphi$ is satisfied in every structure which satisfies **TA**. Since $\mathfrak{M} \vDash \mathbf{TA}$, $\mathfrak{M} \vDash \varphi$, and so $\varphi \in \mathbf{TA}$.

The other way to specify a theory $\Gamma$ is as the set of sentences entailed by some set of sentences $\Gamma_0$. In that case, $\Gamma$ is the "closure" of $\Gamma_0$ under entailment. Specifying a theory this way is only interesting if $\Gamma_0$ is explicitly specified, e.g., if the elements of $\Gamma_0$ are listed. At the very least, $\Gamma_0$ has to be decidable, i.e., there has to be a computable test for when a sentence counts as an element of $\Gamma_0$ or not. We call the sentences in $\Gamma_0$ *axioms* for $\Gamma$, and $\Gamma$ *axiomatized* by $\Gamma_0$.

**Definition 30.4.** A theory $\Gamma$ is *axiomatized* by $\Gamma_0$ iff

$$\Gamma = \{\varphi : \Gamma_0 \vDash \varphi\}$$

**Definition 30.5.** The theory **Q** axiomatized by the following sentences is known as "Robinson's **Q**" and is a very simple theory of arithmetic.

$$\forall x\, \forall y\, (x' = y' \to x = y) \tag{$Q_1$}$$
$$\forall x\, \mathsf{o} \neq x' \tag{$Q_2$}$$
$$\forall x\, (x \neq \mathsf{o} \to \exists y\, x = y') \tag{$Q_3$}$$
$$\forall x\, (x + \mathsf{o}) = x \tag{$Q_4$}$$
$$\forall x\, \forall y\, (x + y') = (x + y)' \tag{$Q_5$}$$
$$\forall x\, (x \times \mathsf{o}) = \mathsf{o} \tag{$Q_6$}$$
$$\forall x\, \forall y\, (x \times y') = ((x \times y) + x) \tag{$Q_7$}$$
$$\forall x\, \forall y\, (x < y \leftrightarrow \exists z\, (x + z' = y)) \tag{$Q_8$}$$

The set of sentences $\{Q_1, \ldots, Q_8\}$ are the axioms of **Q**, so **Q** consists of all sentences entailed by them:

$$\mathbf{Q} = \{\varphi : \{Q_1, \ldots, Q_8\} \vDash \varphi\}.$$

**Definition 30.6.** Suppose $\varphi(x)$ is a formula in $\mathcal{L}_A$ with free variables $x$ and $y_1$, $\ldots$, $y_n$. Then any sentence of the form

$$\forall y_1 \ldots \forall y_n \left((\varphi(0) \wedge \forall x \left(\varphi(x) \to \varphi(x')\right)\right) \to \forall x\, \varphi(x)\right)$$

is an instance of the *induction schema*.

*Peano arithmetic* **PA** is the theory axiomatized by the axioms of **Q** together with all instances of the induction schema.

Every instance of the induction schema is true in $\mathfrak{N}$. This is easiest to see if the formula $\varphi$ only has one free variable $x$. Then $\varphi(x)$ defines a subset $X_A$ of $\mathbb{N}$ in $\mathfrak{N}$. $X_A$ is the set of all $n \in \mathbb{N}$ such that $\mathfrak{N}, s \vDash \varphi(x)$ when $s(x) = n$. The corresponding instance of the induction schema is

$$\left((\varphi(0) \wedge \forall x \left(\varphi(x) \to \varphi(x')\right)\right) \to \forall x\, \varphi(x)\right)$$

If its antecedent is true in $\mathfrak{N}$, then $0 \in X_A$ and, whenever $n \in X_A$, so is $n + 1$. Since $0 \in X_A$, we get $1 \in X_A$. With $1 \in X_A$ we get $2 \in X_A$. And so on. So for every $n \in \mathbb{N}$, $n \in X_A$. But this means that $\forall x\, \varphi(x)$ is satisfied in $\mathfrak{N}$.

Both **Q** and **PA** are axiomatized theories. The big question is, how strong are they? For instance, can **PA** prove all the truths about $\mathbb{N}$ that can be expressed in $\mathcal{L}_A$? Specifically, do the axioms of **PA** settle all the questions that can be formulated in $\mathcal{L}_A$?

Another way to put this is to ask: Is **PA** $=$ **TA**? **TA** obviously does prove (i.e., it includes) all the truths about $\mathbb{N}$, and it settles all the questions that can be formulated in $\mathcal{L}_A$, since if $\varphi$ is a sentence in $\mathcal{L}_A$, then either $\mathfrak{N} \vDash \varphi$ or $\mathfrak{N} \vDash \neg\varphi$, and so either **TA** $\vDash \varphi$ or **TA** $\vDash \neg\varphi$. Call such a theory complete.

**Definition 30.7.** A theory $\Gamma$ is *complete* iff for every sentence $\varphi$ in its language, either $\Gamma \vDash \varphi$ or $\Gamma \vDash \neg\varphi$.

By the Completeness Theorem, $\Gamma \vDash \varphi$ iff $\Gamma \vdash \varphi$, so $\Gamma$ is complete iff for every sentence $\varphi$ in its language, either $\Gamma \vdash \varphi$ or $\Gamma \vdash \neg\varphi$.

Another question we are led to ask is this: Is there a computational procedure we can use to test if a sentence is in **TA**, in **PA**, or even just in **Q**? We can make this more precise by defining when a set (e.g., a set of sentences) is decidable.

**Definition 30.8.** A set $X$ is *decidable* iff there is a computational procedure which on input $x$ returns 1 if $x \in X$ and 0 otherwise.

So our question becomes: Is **TA** (**PA**, **Q**) decidable?

The answer to all these questions will be: no. None of these theories are decidable. However, this phenomenon is not specific to these particular theories. In fact, *any* theory that satisfies certain conditions is subject to the same results. One of these conditions, which **Q** and **PA** satisfy, is that they are axiomatized by a decidable set of axioms.

**Definition 30.9.** A theory is *axiomatizable* if it is axiomatized by a decidable set of axioms.

**Example 30.10.** Any theory axiomatized by a finite set of sentences is axiomatizable, since any finite set is decidable. Thus, **Q**, for instance, is axiomatizable.

Schematically axiomatized theories like **PA** are also axiomatizable. For to test if $\psi$ is among the axioms of **PA**, i.e., to compute the function $\chi_X$ where $\chi_X(\psi) = 1$ if $\psi$ is an axiom of **PA** and $= 0$ otherwise, we can do the following: First, check if $\psi$ is one of the axioms of **Q**. If it is, the answer is "yes" and the value of $\chi_X(\psi) = 1$. If not, test if it is an instance of the induction schema. This can be done systematically; in this case, perhaps it's easiest to see that it can be done as follows: Any instance of the induction schema begins with a number of universal quantifiers, and then a sub-formula that is a conditional. The consequent of that conditional is $\forall x\, \varphi(x, y_1, \ldots, y_n)$ where $x$ and $y_1, \ldots, y_n$ are all the free variables of $\varphi$ and the initial quantifiers of $\psi$ bind the variables $y_1$, $\ldots, y_n$. Once we have extracted this $\varphi$ and checked that its free variables match the variables bound by the universal qauntifiers at the front and $\forall x$, we go on to check that the antecedent of the conditional matches

$$\varphi(\mathsf{0}, y_1, \ldots, y_n) \wedge \forall x\, (\varphi(x, y_1, \ldots, y_n) \to \varphi(x', y_1, \ldots, y_n))$$

Again, if it does, $\psi$ is an instance of the induction schema, and if it doesn't, $\psi$ isn't.

In answering this question—and the more general question of which theories are complete or decidable—it will be useful to consider also the following definition. Recall that a set $X$ is enumerable iff it is empty or if there is a surjective function $f : \mathbb{N} \to X$. Such a function is called an enumeration of $X$.

**Definition 30.11.** A set $X$ is called *computably enumerable* (c.e. for short) iff it is empty or it has a computable enumeration.

In addition to axiomatizability, another condition on theories to which the incompleteness theorems apply will be that they are strong enough to prove basic facts about computable functions and decidable relations. By "basic facts," we mean sentences which express what the values of computable functions are for each of their arguments. And by "strong enough" we mean that the theories in question count these sentences among its theorems. For instance, consider a prototypical computable function: addition. The value of

$+$ for arguments 2 and 3 is 5, i.e., $2 + 3 = 5$. A sentence in the language of arithmetic that expresses that the value of $+$ for arguments 2 and 3 is 5 is: $(\overline{2} + \overline{3}) = \overline{5}$. And, e.g., **Q** proves this sentence. More generally, we would like there to be, for each computable function $f(x_1, x_2)$ a formula $\varphi_f(x_1, x_2, y)$ in $\mathcal{L}_A$ such that $\mathbf{Q} \vdash \varphi_f(\overline{n_1}, \overline{n_2}, \overline{m})$ whenever $f(n_1, n_2) = m$. In this way, **Q** proves that the value of $f$ for arguments $n_1$, $n_2$ is $m$. In fact, we require that it proves a bit more, namely that no other number is the value of $f$ for arguments $n_1$, $n_2$. And the same goes for decidable relations. This is made precise in the following two definitions.

**Definition 30.12.** A formula $\varphi(x_1, \ldots, x_k, y)$ *represents* the function $f\colon \mathbb{N}^k \to \mathbb{N}$ in $\Gamma$ iff whenever $f(n_1, \ldots, n_k) = m$, then

1. $\Gamma \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k}, \overline{m})$, and
2. $\Gamma \vdash \forall y(\varphi(\overline{n_1}, \ldots, \overline{n_k}, y) \to y = \overline{m})$.

**Definition 30.13.** A formula $\varphi(x_1, \ldots, x_k)$ *represents* the relation $R \subseteq \mathbb{N}^k$ iff,

1. whenever $R(n_1, \ldots, n_k)$, $\Gamma \vdash \varphi(\overline{n_1}, \ldots, \overline{n_k})$, and
2. whenever not $R(n_1, \ldots, n_k)$, $\Gamma \vdash \neg\varphi(\overline{n_1}, \ldots, \overline{n_k})$.

A theory is "strong enough" for the incompleteness theorems to apply if it represents all computable functions and all decidable relations. **Q** and its extensions satisfy this condition, but it will take us a while to establish this—it's a non-trivial fact about the kinds of things **Q** can prove, and it's hard to show because **Q** has only a few axioms from which we'll have to prove all these facts. However, **Q** is a very weak theory. So although it's hard to prove that **Q** represents all computable functions, most interesting theories are stronger than **Q**, i.e., prove more than **Q** does. And if **Q** proves something, any stronger theory does; since **Q** represents all computable functions, every stronger theory does. This means that many interesting theories meet this condition of the incompleteness theorems. So our hard work will pay off, since it shows that the incompletess theorems apply to a wide range of theories. Certainly, any theory aiming to formalize "all of mathematics" must prove everything that **Q** proves, since it should at the very least be able to capture the results of elementary computations. So any theory that is a candidate for a theory of "all of mathematics" will be one to which the incompleteness theorems apply.

## 30.3 Overview of Incompleteness Results

Hilbert expected that mathematics could be formalized in an axiomatizable theory which it would be possible to prove complete and decidable. Moreover, he aimed to prove the consistency of this theory with very weak, "fini-

tary," means, which would defend classical mathematics agianst the challenges of intuitionism. Gödel's incompleteness theorems showed that these goals cannot be achieved.

Gödel's first incompleteness theorem showed that a version of Russell and Whitehead's *Principia Mathematica* is not complete. But the proof was actually very general and applies to a wide variety of theories. This means that it wasn't just that *Principia Mathematica* did not manage to completely capture mathematics, but that *no* acceptable theory does. It took a while to isolate the features of theories that suffice for the incompleteness theorems to apply, and to generalize Gödel's proof to apply make it depend only on these features. But we are now in a position to state a very general version of the first incompleteness theorem for theories in the language $\mathcal{L}_A$ of arithmetic.

**Theorem 30.14.** *If $\Gamma$ is a consistent and axiomatizable theory in $\mathcal{L}_A$ which represents all computable functions and decidable relations, then $\Gamma$ is not complete.*

To say that $\Gamma$ is not complete is to say that for at least one sentence $\varphi$, $\Gamma \nvdash \varphi$ and $\Gamma \nvdash \neg\varphi$. Such a sentence is called *independent* (of $\Gamma$). We can in fact relatively quickly prove that there must be independent sentences. But the power of Gödel's proof of the theorem lies in the fact that it exhibits a *specific example* of such an independent sentence. The intriguing construction produces a sentence $G_\Gamma$, called a *Gödel sentence* for $\Gamma$, which is unprovable because in $\Gamma$, $G_\Gamma$ is equivalent to the claim that $G_\Gamma$ is unprovable in $\Gamma$. It does so *constructively*, i.e., given an axiomatization of $\Gamma$ and a description of the proof system, the proof gives a method for actually writing down $G_\Gamma$.

The construction in Gödel's proof requires that we find a way to express in $\mathcal{L}_A$ the properties of and operations on terms and formulas of $\mathcal{L}_A$ itself. These include properties such as "$\varphi$ is a sentence," "$\delta$ is a derivation of $\varphi$," and operations such as $\varphi[t/x]$. This way must (a) express these properties and relations via a "coding" of symbols and sequences thereof (which is what terms, formulas, derivations, etc. are) as natural numbers (which is what $\mathcal{L}_A$ can talk about). It must (b) do this in such a way that $\Gamma$ will prove the relevant facts, so we must show that these properties are coded by decidable properties of natural numbers and the operations correspond to computable functions on natural numbers. This is called "arithmetization of syntax."

Before we investigate how syntax can be arithmetized, however, we will consider the condition that $\Gamma$ is "strong enough," i.e., represents all computable functions and decidable relations. This requires that we give a precise definition of "computable." This can be done in a number of ways, e.g., via the model of Turing machines, or as those functions computable by programs in some general-purpose programming language. Since our aim is to represent these functions and relations in a theory in the language $\mathcal{L}_A$, however, it is best to pick a simple definition of computability of just numerical functions. This is the notion of *recursive function*. So we will first discuss the recursive

functions. We will then show that **Q** already represents all recursive functions and relations. This will allow us to apply the incompleteness theorem to specific theories such as **Q** and **PA**, since we will have established that these are examples of theories that are "strong enough."

The end result of the arithmetization of syntax is a formula $\mathsf{Prov}_\Gamma(x)$ which, via the coding of formulas as numbers, expresses provability from the axioms of $\Gamma$. Specifically, if $\varphi$ is coded by the number $n$, and $\Gamma \vdash \varphi$, then $\Gamma \vdash \mathrm{Prov}_\Gamma(\overline{n})$. This "provability predicate" for $\Gamma$ allows us also to express, in a certain sense, the consistency of $\Gamma$ as a sentence of $\mathcal{L}_A$: let the "consistency statemetn" for $\Gamma$ be the sentence $\neg\mathrm{Prov}_\Gamma(\overline{n})$, where we take $n$ to be the code of a contradiction, e.g., of $\bot$. The second incompleteness theorem states that consistent axiomatizable theories also do not prove their own consistency statements. The conditions required for this theorem to apply are a bit more stringent than just that the theory represents all computable functions and decidable relations, but we will show that **PA** satisifes them.

## 30.4 Undecidability and Incompleteness

Gödel's proof of the incompleteness theorems require arithmetization of syntax. But even without that we can obtain some nice results just on the assumtion that a theory represents all decidable relations. The proof is a diagonal argument similar to the proof of the undecidability of the halting problem.

**Theorem 30.15.** *If $\Gamma$ is a consistent theory that represents every decidable relation, then $\Gamma$ is not decidable.*

*Proof.* Suppose $\Gamma$ were decidable. We show that if $\Gamma$ represents every decidable relation, it must be inconsistent.

Decidable properties (one-place relations) are represented by formulas with one free variable. Let $\varphi_0(x)$, $\varphi_1(x)$, $\ldots$, be a computable enumeration of all such formulas. Now consider the following set $D \subseteq \mathbb{N}$:

$$D = \{n : \Gamma \vdash \neg\varphi_n(\overline{n})\}$$

The set $D$ is decidable, since we can test if $n \in D$ by first computing $\varphi_n(x)$, and from this $\neg\varphi_n(\overline{n})$. Obviously, substituting the term $\overline{n}$ for every free occurrence of $x$ in $\varphi_n(x)$ and prefixing $\varphi(\overline{n})$ by $\neg$ is a mechanical matter. By assumption, $\Gamma$ is decidable, so we can test if $\neg\varphi(\overline{n}) \in \Gamma$. If it is, $n \in D$, and if it isn't, $n \notin D$. So $D$ is likewise decidable.

Since $\Gamma$ represents all decidable properties, it represents $D$. And the formulas which represent $D$ in $\Gamma$ are all among $\varphi_0(x)$, $\varphi_1(x)$, $\ldots$. So let $d$ be a number such that $\varphi_d(x)$ represents $D$ in $\Gamma$. If $d \notin D$, then, since $\varphi_d(x)$ represents $D$, $\Gamma \vdash \neg\varphi_d(\overline{d})$. But that means that $d$ meets the defining condition of $D$, and so $d \in D$. This contradicts $d \notin D$. So by indirect proof, $d \in D$.

Since $d \in D$, by the definition of $D$, $\Gamma \vdash \neg\varphi_d(\overline{d})$. On the other hand, since $\varphi_d(x)$ represents $D$ in $\Gamma$, $\Gamma \vdash \varphi_d(\overline{d})$. Hence, $\Gamma$ is inconsistent. □

The preceding theorem shows that no theory that represents all decidable relations can be decidable. We will show that **Q** does represent all decidable relations; this means that all theories that include **Q**, such as **PA** and **TA**, also do, and hence also are not decidable.

We can also use this result to obtain a weak version of the first incompleteness theorem. Any theory that is axiomatizable and complete is decidable. Consistent theories that are axiomatizable and represent all decidable properties then cannot be complete.

**Theorem 30.16.** *If $\Gamma$ is axiomatizable and complete it is decidable.*

*Proof.* Any inconsistent theory is decidable, since inconsistent theories contain all sentences, so the answer to the question "is $\varphi \in \Gamma$" is always "yes," i.e., can be decided.

So suppose $\Gamma$ is consistent, and furthermore is axiomatizable, and complete. Since $\Gamma$ is axiomatizable, it is computably enumerable. For we can enumerate all the correct derivations from the axioms of $\Gamma$ by a computable function. From a correct derivation we can compute the sentence it derives, and so together there is a computable function that enumerates all theorems of $\Gamma$. A sentence is a theorem of $\Gamma$ iff $\neg\varphi$ is not a theorem, since $\Gamma$ is consistent and complete. We can therefore decide if $\varphi \in \Gamma$ as follows. Enumerate all theorems of $\Gamma$. When $\varphi$ appears on this list, we know that $\Gamma \vdash \varphi$. When $\neg\varphi$ appears on this list, we know that $\Gamma \nvdash \varphi$. Since $\Gamma$ is complete, one of these cases eventually obtains, so the procedure eventually produces and answer. □

**Corollary 30.17.** *If $\Gamma$ is consistent, axiomatizable, and represents every decidable property, it is not complete.*

*Proof.* If $\Gamma$ were complete, it would be decidable by the previous theorem (since it is axiomatizable and consistent). But since $\Gamma$ represents every decidable property, it is not decidable, by the first theorem. □

Once we have established that, e.g., **Q**, represents all decidable properties, the corollary tells us that **Q** must be incomplete. However, its proof does not provide an example of an independent sentence; it merely shows that such a sentence must exist. For this, we have to arithmetize syntax and follow Gödel's original proof idea. And of course, we still have to show the first claim, namely that **Q** does, in fact, represent all decidable properties.

It should be noted that not every *interesting* theory is incomplete or undecidable. There are many theories that are sufficiently strong to describe interesting mathematical facts that do not satisify the conditions of Gödel's result. For instance, $\mathbf{Pres} = \{\varphi \in \mathcal{L}_{A^+} : \mathbb{N} \vDash \varphi\}$, the set of sentences of the language

of arithmetic without $\times$ true in the standard model, is both complete and decidable. This theory is called Presburger arithmetic, and proves all the truths about natural numbers that can be formulated just with $0$, $'$, and $+$.

## Problems

**Problem 30.1.** Show that $\mathbf{TA} = \{\varphi : \mathfrak{N} \vDash \varphi\}$ is not axiomatizable. You may assume that **TA** represents all decidable properties.

# Chapter 31

# Arithmetization of Syntax

Note that arithmetization for signed tableaux is not yet available.

## 31.1 Introduction

In order to connect computability and logic, we need a way to talk about the objects of logic (symbols, terms, formulas, derivations), operations on them, and their properties and relations, in a way amenable to computational treatment. We can do this directly, by considering computable functions and relations on symbols, sequences of symbols, and other objects built from them. Since the objects of logical syntax are all finite and built from an enumerable sets of symbols, this is possible for some models of computation. But other models of computation—such as the recursive functions—-are restricted to numbers, their relations and functions. Moreover, ultimately we also want to be able to deal with syntax within certain theories, specifically, in theories formulated in the language of arithmetic. In these cases it is necessary to *arithmetize* syntax, i.e., to represent syntactic objects, operations on them, and their relations, as numbers, arithmetical functions, and arithmetical relations, respectively. The idea, which goes back to Leibniz, is to assign numbers to syntactic objects.

It is relatively straightforward to assign numbers to symbols as their "codes." Some symbols pose a bit of a challenge, since, e.g., there are infinitely many variables, and even infinitely many function symbols of each arity $n$. But of course it's possible to assign numbers to symbols systematically in such a way that, say, $v_2$ and $v_3$ are assigned different codes. Sequences of symbols (such as terms and formulas) are a bigger challenge. But if can deal with sequences of numbers purely arithmetically (e.g., by the powers-of-primes coding of sequences), we can extend the coding of individual symbols to coding of sequences of symbols, and then further to sequences or other arrangements of

formulas, such as derivations. This extended coding is called "Gödel numbering." Every term, formula, and derivation is assigned a Gödel number.

By coding sequences of symbols as sequences of their codes, and by chosing a system of coding sequences that can be dealt with using computable functions, we can then also deal with Gödel numbers using computable functions. In practice, all the relevant functions will be primitive recursive. For instance, computing the length of a sequence and computing the $i$-th element of a sequence from the code of the sequence are both primitive recursive. If the number coding the sequence is, e.g., the Gödel number of a formula $\varphi$, we immediately see that the length of a formula and the (code of the) $i$-th symbol in a formula can also be computed from the Gödel number of $\varphi$. It is a bit harder to prove that, e.g., the property of being the Gödel number of a correctly formed term, of being the Gödel number of a corret derivation is primitive recursive. It is nevertheless possible, because the sequences of interest (terms, formulas, derivations) are inductively defined.

As an example, consider the operation of substitution. If $\varphi$ is a formula, $x$ a variable, and $t$ a term, then $\varphi[t/x]$ is the result of replacing every free occurrence of $x$ in $\varphi$ by $t$. Now suppose we have assigned Gödel numbers to $\varphi$, $x$, $t$—say, $k$, $l$, and $m$, respectively. The same scheme assigns a Gödel number to $\varphi[t/x]$, say, $n$. This mapping—of $k$, $l$, and $m$ to $n$—is the arithmetical analog of the substitution operation. When the substitution operation maps $\varphi$, $x$, $t$ to $\varphi[t/x]$, the arithmetized substitution functions maps the Gödel numbers $k$, $l$, $m$ to the Gödel number $n$. We will see that this function is primitive recursive.

Arithmetization of syntax is not just of abstract interest, although it was originally a non-trivial insight that languages like the language of arithmetic, which do not come with mechanisms for "talking about" languages can, after all, formalize complex properties of expressions. It is then just a small step to ask what a theory in this language, such as Peano arithmetic, can *prove* about its own language (including, e.g., whether sentences are provable or true). This leads us to the famous limitative theorems of Gödel (about unprovability) and Tarski (the undefinability of truth). But the trick of arithmetizing syntax is also important in order to prove some important results in computability theory, e.g., about the computational prower of theories or the relationship between different models of computability. The arithmetization of syntax serves as a model for arithmetizing other objects and properties. For instance, it is similarly possible to arithmetize configurations and computations (say, of Turing machines). This makes it possible to simulate computations in one model (e.g., Turing machines) in another (e.g., recursive functions).

## 31.2 Coding Symbols

The basic language $\mathcal{L}$ of first order logic makes use of the symbols

$$\bot \quad \neg \quad \vee \quad \wedge \quad \to \quad \forall \quad \exists \quad = \quad ( \quad ) \quad ,$$

together with enumerable sets of variables and constant symbols, and enumerable sets of function symbols and predicate symbols of arbitrary arity. We can assign *codes* to each of these symbols in such a way that every symbol is assigned a unique number as its code, and no two different symbols are assigned the same number. We know that this is possible since the set of all symbols is enumerable and so there is a bijection between it and the set of natural numbers. But we want to make sure that we can recover the symbol (as well as some information about it, e.g., the arity of a function symbol) from its code in a computable way. There are many possible ways of doing this, of course. Here is one such way, which uses primitive recursive functions. (Recall that $\langle n_0, \ldots, n_k \rangle$ is the number coding the sequence of numbers $n_0, \ldots, n_k$.)

**Definition 31.1.** If $s$ is a symbol of $\mathcal{L}$, let the *symbol code* $\mathrm{c}_s$ be defined as follows:

1. If $s$ is among the logical symbols, $\mathrm{c}_s$ is given by the following table:

| $\bot$ | $\neg$ | $\vee$ | $\wedge$ | $\to$ | $\forall$ |
|---|---|---|---|---|---|
| $\langle 0,0 \rangle$ | $\langle 0,1 \rangle$ | $\langle 0,2 \rangle$ | $\langle 0,3 \rangle$ | $\langle 0,4 \rangle$ | $\langle 0,5 \rangle$ |
| $\exists$ | $=$ | $($ | $)$ | , | |
| $\langle 0,6 \rangle$ | $\langle 0,7 \rangle$ | $\langle 0,8 \rangle$ | $\langle 0,9 \rangle$ | $\langle 0,10 \rangle$ | |

2. If $s$ is the $i$-th variable $v_i$, then $\mathrm{c}_s = \langle 1, i \rangle$.
3. If $s$ is the $i$-th constant symbol $c_i^n$, then $\mathrm{c}_s = \langle 2, i \rangle$.
4. If $s$ is the $i$-th $n$-ary function symbol $f_i^n$, then $\mathrm{c}_s = \langle 3, n, i \rangle$.
5. If $s$ is the $i$-th $n$-ary predicate symbol $P_i^n$, then $\mathrm{c}_s = \langle 4, n, i \rangle$.

**Proposition 31.2.** *The following relations are primitive recursive:*

1. $\mathrm{Fn}(x, n)$ *iff* $x$ *is the code of* $f_i^n$ *for some* $i$*, i.e.,* $x$ *is the code of an* $n$*-ary function symbol.*
2. $\mathrm{Pred}(x, n)$ *iff* $x$ *is the code of* $P_i^n$ *for some* $i$ *or* $x$ *is the code of* $=$ *and* $n = 2$*, i.e.,* $x$ *is the code of an* $n$*-ary predicate symbol.*

**Definition 31.3.** If $s_0, \ldots, s_{n-1}$ is a sequence of symbols, its *Gödel number* is $\langle \mathrm{c}_{s_0}, \ldots, \mathrm{c}_{s_{n-1}} \rangle$.

Note that *codes* and *Gödel numbers* are different things. For instance, the variable $v_5$ has a code $\mathrm{c}_{v_5} = \langle 1, 5 \rangle = 2^2 \cdot 3^6$. But the variable $v_5$ considered as a term is also a sequence of symbols (of length 1). The *Gödel number* $\ulcorner v_5 \urcorner$ of the *term* $v_5$ is $\langle \mathrm{c}_{v_5} \rangle = 2^{\mathrm{c}_{v_5}+1} = 2^{2^2 \cdot 3^6 + 1}$.

**Example 31.4.** Recall that if $k_0, \ldots, k_{n-1}$ is a sequence of numbers, then the code of the sequence $\langle k_0, \ldots, k_{n-1}\rangle$ in the power-of-primes coding is

$$2^{k_0+1} \cdot 3^{k_1+1} \cdot \ldots \cdot p_{n-1}^{k_{n-1}},$$

where $p_i$ is the $i$-th prime (starting with $p_0 = 2$). So for instance, the formula $v_0 = 0$, or, more explicitly, $=(v_0, c_0)$, has the Gödel number

$$\langle \mathrm{c}_=, \mathrm{c}_(, \mathrm{c}_{v_0}, \mathrm{c}_,, \mathrm{c}_{c_0}, \mathrm{c}_)\rangle.$$

Here, $\mathrm{c}_=$ is $\langle 0, 7\rangle = 2^{0+1} \cdot 3^{7=1}$, $\mathrm{c}_{v_0}$ is $\langle 1, 0\rangle = 2^{1+1} \cdot 3^{0+1}$, etc. So ${}^{\#}{=}\,(v_0, c_0)^{\#}$ is

$$\begin{aligned} &2^{\mathrm{c}_=+1} \cdot 3^{\mathrm{c}_(+1} \cdot 5^{\mathrm{c}_{v_0}+1} \cdot 7^{\mathrm{c}_,+1} \cdot 11^{\mathrm{c}_{c_0}+1} \cdot 13^{\mathrm{c}_)+1} = \\ &\qquad 2^{2^1\cdot 3^8+1} \cdot 3^{2^1\cdot 3^9+1} \cdot 5^{2^2\cdot 3^1+1} \cdot 7^{2^1\cdot 3^{11}+1} \cdot 11^{2^3\cdot 3^1+1} \cdot 13^{2^1\cdot 3^{10}+1} = \\ &\qquad\qquad 2^{13\,123} \cdot 3^{39\,367} \cdot 5^{13} \cdot 7^{354\,295} \cdot 11^{25} \cdot 13^{118\,099}. \end{aligned}$$

## 31.3 Coding Terms

A term is simply a certain kind of sequence of symbols: it is built up inductively from constants and variables according to the formation rules for terms. Since sequences of symbols can be coded as numbers—using a coding scheme for the symbols plus a way to code sequences of numbers—assigning Gödel numbers to terms is not difficult. The challenge is rather to show that the property a number has if it is the Gödel number of a correctly formed term is computable, or in fact primitive recursive.

Variables and constant symbols are the simplest terms, and testing whether $x$ is the Gödel number of such a term is easy: $\mathrm{Var}(x)$ holds if $x$ is ${}^{\#}v_i{}^{\#}$ for some $i$. In other words, $x$ is a sequence of length 1 and its single element $(x)_0$ is the code of some variable $v_i$, i.e., $x$ is $\langle\langle 1, i\rangle\rangle$ for some $i$. Similarly, $\mathrm{Const}(x)$ holds if $x$ is ${}^{\#}c_i{}^{\#}$ for some $i$. Both of these relations are primitive recursive, since if such an $i$ exists, it must be $< x$:

$$\begin{aligned} \mathrm{Var}(x) &\Leftrightarrow (\exists i < x)\ x = \langle\langle 1, i\rangle\rangle \\ \mathrm{Const}(x) &\Leftrightarrow (\exists i < x)\ x = \langle\langle 2, i\rangle\rangle \end{aligned}$$

**Proposition 31.5.** *The relations* $\mathrm{Term}(x)$ *and* $\mathrm{ClTerm}(x)$ *which hold iff $x$ is the Gödel number of a term or a closed term, respectively, are primitive recursive.*

*Proof.* A sequence of symbols $s$ is a term iff there is a sequence $s_0, \ldots, s_{k-1} = s$ of terms which records how the term $s$ was formed from constant symbols and variables according to the formation rules for terms. To express that such a putative formation sequence follows the formation rules it has to be the case that, for each $i < k$, either

1. $s_i$ is a variable $v_j$, or
2. $s_i$ is a constant symbol $c_j$, or
3. $s_i$ is built from $n$ terms $t_1, \ldots, t_n$ occurring prior to place $i$ using an $n$-place function symbol $f^n_j$.

To show that the corresponding relation on Gödel numbers is primitive recursive, we have to express this condition primitive recursively, i.e., using primitive recursive functions, relations, and bounded quantification.

Suppose $y$ is the number that codes the sequence $s_0, \ldots, s_{k-1}$, i.e., $y = \langle {}^\ulcorner s_0 {}^\urcorner, \ldots, {}^\ulcorner s_k {}^\urcorner \rangle$. It codes a formation sequence for the term with Gödel number $x$ iff for all $i < k$:

1. $\mathrm{Var}((y)_i)$, or
2. $\mathrm{Const}((y)_i)$, or
3. there is an $n$ and a number $z = \langle z_1, \ldots, z_n \rangle$ such that each $z_l$ is equal to some $(y)_{i'}$ for $i' < i$ and

$$(y)_i = {}^\ulcorner f^n_j {}^\frown ({}^\frown \mathrm{flatten}(z) {}^\frown {}^\urcorner) {}^\urcorner,$$

and moreover $(y)_{k-1} = x$. (The function $\mathrm{flatten}(z)$ turns the sequence $\langle {}^\ulcorner t_1 {}^\urcorner, \ldots, {}^\ulcorner t_n {}^\urcorner \rangle$ into ${}^\ulcorner t_1, \ldots, t_n {}^\urcorner$ and is primitive recursive.)

The indices $j$, $n$, the Gödel numbers $z_l$ of the terms $t_l$, and the code $z$ of the sequence $\langle z_1, \ldots, z_n \rangle$, in (3) are all less than $y$. We can replace $k$ above with $\mathrm{len}(y)$. Hence we can express "$y$ is the code of a formation sequence of the term with Gödel number $x$" in a way that shows that this relation is primitive recursive.

We now just have to convince ourselves that there is a primitive recursive bound on $y$. But if $x$ is the Gödel number of a term, it must have a formation sequence with at most $\mathrm{len}(x)$ terms (since every term in the formation sequence of $s$ must start at some place in $s$, and no two subterms can start at the same place). The Gödel number of each subterm of $s$ is of course $\leq x$. Hence, there always is a formation sequence with code $\leq x^{\mathrm{len}(x)}$.

For ClTerm, simply leave out the clause for variables. □

**Proposition 31.6.** *The function* $\mathrm{num}(n) = {}^\ulcorner \overline{n} {}^\urcorner$ *is primitive recursive.*

*Proof.* We define $\mathrm{num}(n)$ by primitive recursion:

$$\begin{aligned} \mathrm{num}(0) &= {}^\ulcorner \mathsf{0} {}^\urcorner \\ \mathrm{num}(n+1) &= {}^\ulcorner \mathsf{'} ({}^\frown \mathrm{num}(n) {}^\frown {}^\urcorner) {}^\urcorner. \end{aligned}$$

□

## 31.4 Coding Formulas

**Proposition 31.7.** *The relation* $\mathrm{Atom}(x)$ *which holds iff* $x$ *is the Gödel number of an atomic formula, is primitive recursive.*

*Proof.* The number $x$ is the Gödel number of an atomic formula iff one of the following holds:

1. There are $n$, $j < x$, and $z < x$ such that for each $i < n$, $\mathrm{Term}((z)_i)$ and $x =$
$$^{\#}P^n_j(^{\#} \frown \mathrm{flatten}(z) \frown {}^{\#})^{\#}.$$
2. There are $z_1, z_2 < x$ such that $\mathrm{Term}(z_1)$, $\mathrm{Term}(z_2)$, and $x =$
$$^{\#}{=}(^{\#} \frown z_1 \frown {}^{\#},^{\#} \frown z_2 \frown {}^{\#})^{\#}.$$
3. $x = {}^{\#}\bot^{\#}$.

□

**Proposition 31.8.** *The relation* $\mathrm{Frm}(x)$ *which holds iff* $x$ *is the Gödel number of a formula is primitive recursive.*

*Proof.* A sequence of symbols $s$ is a formula iff there is formation sequence $s_0$, $\ldots$, $s_{k-1} = s$ of formula which records how $s$ was formed from atomic formulas according to the formation rules. The code for each $s_i$ (and indeed of the code of the sequence $\langle s_0, \ldots, s_{k-1}\rangle$ is less than the code $x$ of $s$. □

**Proposition 31.9.** *The relation* $\mathrm{FreeOcc}(x, z, i)$*, which holds iff the* $i$*-th symbol of the formula with Gödel number* $x$ *is a free occurrence of the variable with Gödel number* $z$*, is primitive recursive.*

*Proof.* Exercise. □

**Proposition 31.10.** *The property* $\mathrm{Sent}(x)$ *which holds iff* $x$ *is the Gödel number of a sentence is primitive recursive.*

*Proof.* A sentence is a formula without free occurrences of variables. So $\mathrm{Sent}(x)$ holds iff
$$(\forall i < \mathrm{len}(x))\ (\forall z < x)\ ((\exists j < z)\ z = {}^{\#}v_j{}^{\#} \to \neg\mathrm{FreeOcc}(x, z, i)).$$

□

## 31.5 Substitution

Recall that substitution is the operation of replacing all free occurrences of a variable $u$ in a formula $\varphi$ by a term $t$, written $\varphi[t/u]$. This operation, when carried out on Gödel numbers of variables, formulas, and terms, is primitive recursive.

**Proposition 31.11.** *There is a primitive recursive function* $\mathrm{Subst}(x, y, z)$ *with the property that*

$$\mathrm{Subst}({}^{\#}\varphi^{\#}, {}^{\#}t^{\#}, {}^{\#}u^{\#}) = {}^{\#}\varphi[t/u]^{\#}$$

*Proof.* We can then define a function hSubst by primitive recursion as follows:

$$\begin{aligned}
\mathrm{hSubst}(x, y, z, 0) &= \Lambda \\
\mathrm{hSubst}(x, y, z, i+1) &= \\
&\begin{cases} \mathrm{hSubst}(x, y, z, i) \frown y & \text{if } \mathrm{FreeOcc}(x, z, i) \\ \mathrm{append}(\mathrm{hSubst}(x, y, z, i), (x)_i) & \text{otherwise.} \end{cases}
\end{aligned}$$

$\mathrm{Subst}(x, y, z)$ can now be defined as $\mathrm{hSubst}(x, y, z, \mathrm{len}(x))$. □

**Proposition 31.12.** *The relation* $\mathrm{FreeFor}(x, y, z)$*, which holds iff the term with Gödel number* $y$ *is free for the variable with Gödel number* $z$ *in the formula with Gödel number* $x$*, is primitive recursive.*

*Proof.* Exercise. □

## 31.6 Derivations in LK

> The definition of Deriv is not actually primitive recursive and needs to be rewritten.

In order to arithmetize derivations, we must represent derivations as numbers. Since derivations are trees of sequents where each inference carries also a label, a recursive representation is the most obvious approach: we represent a derivation as a tuple, the components of which are the end-sequent, the label, and the representations of the sub-derivations leading to the premises of the last inference.

**Definition 31.13.** If $\Gamma$ is a finite sequence of sentences, $\Gamma = \langle \varphi_1, \ldots, \varphi_n \rangle$, then ${}^{\#}\Gamma^{\#} = \langle {}^{\#}\varphi_1{}^{\#}, \ldots, {}^{\#}\varphi_n{}^{\#} \rangle$.

If $\Gamma \Rightarrow \Delta$ is a sequent, then a Gödel number of $\Gamma \Rightarrow \Delta$ is

$${}^{\#}\Gamma \Rightarrow \Delta^{\#} = \langle {}^{\#}\Gamma^{\#}, {}^{\#}\Delta^{\#} \rangle$$

If $\pi$ is a derivation in **LK**, then ${}^{\#}\pi^{\#}$ is

1. $\langle 0, {}^{\#}\Gamma \Rightarrow \Delta^{\#} \rangle$ if $\pi$ consists only of the initial sequent $\Gamma \Rightarrow \Delta$.

2. $\langle 1, {}^{\#}\Gamma \Rightarrow \Delta^{\#}, k, {}^{\#}\pi'^{\#} \rangle$ if $\pi$ ends in an inference with one premise, $k$ is given by the following table according to which rule was used in the last inference, and $\pi'$ is the immediate subproof ending in the premise of the last inference.

   | Rule: | WL | WR | CL | CR | XL | XR |
   |---|---|---|---|---|---|---|
   | $k$: | 1 | 2 | 3 | 4 | 5 | 6 |

   | Rule: | $\neg$L | $\neg$R | $\wedge$L | $\vee$R | $\rightarrow$R |
   |---|---|---|---|---|---|
   | $k$: | 7 | 8 | 9 | 10 | 11 |

   | Rule: | $\forall$L | $\forall$R | $\exists$L | $\exists$R | = |
   |---|---|---|---|---|---|
   | $k$: | 12 | 13 | 14 | 15 | 16 |

3. $\langle 2, {}^{\#}\Gamma \Rightarrow \Delta^{\#}, k, {}^{\#}\pi'^{\#}, {}^{\#}\pi''^{\#} \rangle$ if $\pi$ ends in an inference with two premises, $k$ is given by the following table according to which rule was used in the last inference, and $\pi'$, $\pi''$ are the immediate subproof ending in the left and right premise of the last inference, respectively.

   | Rule: | Cut | $\wedge$R | $\vee$L | $\rightarrow$L |
   |---|---|---|---|---|
   | $k$: | 1 | 2 | 3 | 4 |

Having settled on a representation of derivations, we must also show that we can manipulate such derivations primitive recursively, and express their essential properties and relations so. Some operations are simple: e.g., given a Gödel number $d$ of a derivation, $(s)_1$ gives us the Gödel number of its end-sequent. Some are much harder. We'll at least sketch how to do this. The goal is to show that the relation "$\pi$ is a derivation of $\varphi$ from $\Gamma$" is a primitive recursive relation of the Gödel numbers of $\pi$ and $\varphi$.

**Proposition 31.14.** *The following relations are primitive recursive:*

1. *$\Gamma \Rightarrow \Delta$ is an initial sequent.*

2. *$\Gamma \Rightarrow \Delta$ follows from $\Gamma' \Rightarrow \Delta'$ (and $\Gamma'' \Rightarrow \Delta''$) by a rule of* **LK**.

3. *$\pi$ is a correct* **LK***-derivation.*

*Proof.* We have to show that the corresponding relations between Gödel numbers of formulas, sequences of Gödel numbers of formulas (which code sequences of formulas), and Gödel numbers of sequents, are primitive recursive.

1. $\Gamma \Rightarrow \Delta$ is an initial sequent if either there is a sentence $\varphi$ such that $\Gamma \Rightarrow \Delta$ is $\varphi \Rightarrow \varphi$, or there is a term $t$ such that $\Gamma \Rightarrow \Delta$ is $\emptyset \Rightarrow t = t$. In terms of

Gödel numbers, $\mathrm{InitSeq}(s)$ holds iff

$$(\exists x < s)\ (\mathrm{Sent}(x) \wedge s = \langle\langle x\rangle, \langle x\rangle\rangle) \vee$$
$$(\exists t < s)\ (\mathrm{Term}(t) \wedge s = \langle 0, \langle {}^{\ulcorner}{=}{}^{\urcorner}({}^{\#} \frown t \frown {}^{\ulcorner},{}^{\urcorner}, {}^{\#} \frown t \frown {}^{\ulcorner})^{\urcorner}\rangle\rangle).$$

2. Here we have to show that for each rule of inference $R$ the relation $\mathrm{FollowsBy}_R(s, s')$ which holds if $s$ and $s'$ are the Gödel numbers of conclusion and premise of a correct application of $R$ is primitive recursive. If $R$ has two premises, $\mathrm{FollowsBy}_R$ of course has three arguments.

   For instance, $\Gamma \Rightarrow \Delta$ follows correctly from $\Gamma' \Rightarrow \Delta'$ by $\exists\mathrm{R}$ iff $\Gamma = \Gamma'$ and there is a sequence of formulas $\Delta''$, a formula $\varphi$, a variable $x$ and a closed term $t$ such that $\Delta' = \Delta'', \varphi[t/x]$ and $\Delta = \Delta'', \exists x\, \varphi$. We just have to translate this into Gödel numbers. If $s = {}^{\ulcorner}\Gamma \Rightarrow \Delta^{\urcorner}$ then $(s)_0 = {}^{\ulcorner}\Gamma^{\urcorner}$ and $(s)_1 = {}^{\ulcorner}\Delta^{\urcorner}$. So, $\mathrm{FollowsBy}_{\exists\mathrm{R}}(s, s')$ holds iff

   $$(s)_0 = (s')_0 \wedge$$
   $$(\exists d < s)\ (\exists f < s)\ (\exists x < s)\ (\exists t < s')\ (\mathrm{Frm}(f) \wedge \mathrm{Var}(y) \wedge \mathrm{Term}(t) \wedge$$
   $$\quad (s')_1 = d \frown \langle \mathrm{Subst}(f, t, x)\rangle \wedge$$
   $$\quad (s)_1 = d \frown \langle \#(\exists) \frown y \frown f\rangle$$

   The individual lines express, respectively, "$\Gamma = \Gamma$," "there is a sequence ($\Delta''$) with Gödel number $d$, a formula ($\varphi$) with Gödel number $f$, a variable with Gödel number $x$, and a term with Gödel number $t$," "$\Delta' = \Delta'', \varphi[t/x]$," and "$\Delta = \Delta'', \exists x\, \varphi$". (Remember that ${}^{\ulcorner}\Delta^{\urcorner}$ is the number of a sequence of Gödel numbers of formulas in $\Delta$.)

3. We first define a helper relation $\mathrm{hDeriv}(s, n)$ which holds if $s$ codes a correct derivation to at least $n$ inferences up from the end sequent. If $n = 0$ we let the relation be satisfied by default. Otherwise, $\mathrm{hDeriv}(s, n+1)$ iff either $s$ consists just of an initial sequent, or it ends in a correct inference

and the codes of the immediate subderivations satisfy $\mathrm{hDeriv}(s, n)$.

$$
\begin{aligned}
\mathrm{hDeriv}(s, 0) &\Leftrightarrow \mathrm{true} \\
\mathrm{hDeriv}(s, n+1) &\Leftrightarrow \\
&\quad ((s)_0 = 0 \wedge \mathrm{InitialSeq}((s)_1)) \vee \\
&\quad ((s)_0 = 1 \wedge \\
&\qquad ((s)_2 = 1 \wedge \mathrm{FollowsBy}_{\mathrm{CL}}((s)_1, ((s)_3)_1)) \vee \\
&\qquad\quad \vdots \\
&\qquad ((s)_2 = 16 \wedge \mathrm{FollowsBy}_{=}((s)_1, ((s)_3)_1)) \wedge \\
&\qquad \mathrm{hDeriv}((s)_3, n)) \vee \\
&\quad ((s)_0 = 2 \wedge \\
&\qquad ((s)_2 = 1 \wedge \mathrm{FollowsBy}_{\mathrm{Cut}}((s)_1, ((s)_3)_1), ((s)_4)_1)) \vee \\
&\qquad\quad \vdots \\
&\qquad ((s)_2 = 4 \wedge \mathrm{FollowsBy}_{\to \mathrm{L}}((s)_1, ((s)_3)_1), ((s)_4)_1)) \wedge \\
&\qquad \mathrm{hDeriv}((s)_3, n) \wedge \mathrm{hDeriv}((s)_4, n))
\end{aligned}
$$

This is a primitive recursive definition. If the number $n$ is large enough, e.g., larger than the maximum number of inferences between an initial sequent and the end sequent in $s$, it holds of $s$ iff $s$ is the Gödel number of a correct derivation. The number $s$ itself is larger than that maximum number of inferences. So we can now define $\mathrm{Deriv}(s)$ by $\mathrm{hDeriv}(s, s)$.

□

**Proposition 31.15.** *Suppose $\Gamma$ is a primitive recursive set of sentences. Then the relation $\mathrm{Prf}_\Gamma(x, y)$ expressing "$x$ is the code of a derivation $\pi$ of $\Gamma_0 \Rightarrow \varphi$ for some finite $\Gamma_0 \subseteq \Gamma$ and $x$ is the Gödel number of $\varphi$" is primitive recursive.*

*Proof.* Suppose "$y \in \Gamma$" is given by the primitive recursive predicate $R_\Gamma(y)$. We have to show that $\mathrm{Prf}_\Gamma(x, y)$ which holds iff $y$ is the Gödel number of a sentence $\varphi$ and $x$ is the code of an **LK**-derivation with end sequent $\Gamma_0 \Rightarrow \varphi$ is primitive recursive.

By the previous proposition, the property $\mathrm{Deriv}(x)$ which holds iff $x$ is the code of a correct derivation $\pi$ in **LK** is primitive recursive. If $x$ is such a code, then $(x)_1$ is the code of the end sequent of $\pi$, and so $((x)_1)_0$ is the code of the left side of the end sequent and $((x)_1)_1$ the right side. So we can express "the right side of the end sequent of $\pi$ is $\varphi$" as $\mathrm{len}(((x)_1)_1) = 1 \wedge (((x)_1)_1)_0 = x$. The left side of the end sequent of $\pi$ is of course automatically finite, we just have to express that every sentence in it is in $\Gamma$. Thus we can define $\mathrm{Prf}_\Gamma(x, y)$

by

$$\begin{aligned}\mathrm{Prf}_\Gamma(x,y) \Leftrightarrow\ & \mathrm{Sent}(y) \wedge \mathrm{Deriv}(x) \wedge \\ & (\forall i < \mathrm{len}(((x)_1)_0))\ R_\Gamma((((x)_1)_0)_i) \wedge \\ & \mathrm{len}(((x)_1)_1) = 1 \wedge (((x)_1)_1)_0 = x\end{aligned}$$

□

## 31.7 Derivations in Natural Deduction

In order to arithmetize derivations, we must represent derivations as numbers. Since derivations are trees of formulas where each inference carries one or two labels, a recursive representation is the most obvious approach: we represent a derivation as a tuple, the components of which are the number of immediate sub-derivations leading to the premises of the last inference, the representations of these sub-derivations, and the end-formula, the discharge label of the last inference, and a number indicating the type of the last inference.

**Definition 31.16.** If $\delta$ is a derivation in natural deduction, then ${}^\natural\delta^\#$ is defined inductively as follows:

1. If $\delta$ consists only of the assumption $\varphi$, then ${}^\natural\delta^\#$ is $\langle 0, {}^\natural\varphi^\#, n\rangle$. The number $n$ is 0 if it is an undischarged assumption, and the numerical label otherwise.

2. If $\delta$ ends in an inference with one, two, or three premises, then ${}^\natural\delta^\#$ is

$$\begin{aligned}&\langle 1, {}^\natural\delta_1{}^\#, {}^\natural\varphi^\#, n, k\rangle, \\ &\langle 2, {}^\natural\delta_1{}^\#, {}^\natural\delta_2{}^\#, {}^\natural\varphi^\#, n, k\rangle, \text{ or} \\ &\langle 3, {}^\natural\delta_1{}^\#, {}^\natural\delta_2{}^\#, {}^\natural\delta_3{}^\#, {}^\natural\varphi^\#, n, k\rangle,\end{aligned}$$

   respectively. Here $\delta_1, \delta_2, \delta_3$ are the sub-derivations ending in the premise(s) of the last inference in $\delta$, $\varphi$ is the conclusion of the last inference in $\delta$, $n$ is the discharge label of the last inference (0 if the inference does not discharge any assumptions), and $k$ is given by the following table according to which rule was used in the last inference.

| Rule: | $\wedge$Intro | $\wedge$Elim | $\vee$Intro | $\vee$Elim |
|---|---|---|---|---|
| $k$: | 1 | 2 | 3 | 4 |

| Rule: | $\to$Intro | $\to$Elim | $\neg$Intro | $\neg$Elim |
|---|---|---|---|---|
| $k$: | 5 | 6 | 7 | 8 |

| Rule: | $\bot_I$ | $\bot_C$ | $\forall$Intro | $\forall$Elim |
|---|---|---|---|---|
| $k$: | 9 | 10 | 11 | 12 |

| Rule: | $\exists$Intro | $\exists$Elim | $=$Intro | $=$Elim |
|---|---|---|---|---|
| $k$: | 13 | 14 | 15 | 16 |

**Example 31.17.** Consider the very simple derivation

$$\dfrac{\dfrac{[\varphi \wedge \psi]^1}{\varphi}\,\wedge\text{Elim}}{(\varphi \wedge \psi) \to \varphi}\,{}^{1}\,{\to}\text{Intro}$$

The Gödel number of the assumption would be $d_0 = \langle 0, {}^{\#}\varphi \wedge \psi^{\#}, 1\rangle$. The Gödel number of the derivation ending in the conclusion of $\wedge$Elim would be $d_1 = \langle 1, d_0, {}^{\#}\varphi^{\#}, 0, 2\rangle$ (1 since $\wedge$Elim has one premise, the Gödel number of conclusion $\varphi$, 0 because no assumption is discharged, and 2 is the number coding $\wedge$Elim). The Gödel number of the entire derivation then is $\langle 1, d_1, {}^{\#}((\varphi \wedge \psi) \to \varphi)^{\#}, 1, 5\rangle$, i.e.,

$$\langle 1, \langle 1, \langle 0, {}^{\#}(\varphi \wedge \psi)^{\#}, 1\rangle, {}^{\#}\varphi^{\#}, 0, 2\rangle, {}^{\#}((\varphi \wedge \psi) \to \varphi)^{\#}, 1, 5\rangle.$$

Having settled on a representation of derivations, we must also show that we can manipulate Gödel numbers of such derivations primitive recursively, and express their essential properties and relations. Some operations are simple: e.g., given a Gödel number $d$ of a derivation, $\mathrm{EndFmla}(d) = (d)_{(d)_0+1}$ gives us the Gödel number of its end-formula, $\mathrm{DischargeLabel}(d) = (d)_{(d)_0+2}$ gives us the discharge label and $\mathrm{LastRule}(d) = (d)_{(d)_0+3}$ the number indicating the type of the last inference. Some are much harder. We'll at least sketch how to do this. The goal is to show that the relation "$\delta$ is a derivation of $\varphi$ from $\Gamma$" is a primitive recursive relation of the Gödel numbers of $\delta$ and $\varphi$.

**Proposition 31.18.** *The following relations are primitive recursive:*

1. *$\varphi$ occurs as an assumption in $\delta$ with label $n$.*

2. *All assumptions in $\delta$ with label $n$ are of the form $\varphi$ (i.e., we can discharge the assumption $\varphi$ using label $n$ in $\delta$).*

*Proof.* We have to show that the corresponding relations between Gödel numbers of formulas and Gödel numbers of derivations are primitive recursive.

1. We want to show that $\mathrm{Assum}(x, d, n)$, which holds if $x$ is the Gödel number of an assumption of the derivation with Gödel number $d$ labelled $n$, is primitive recursive. This is the case if the derivation with Gödel number $\langle 0, x, n\rangle$ is a sub-derivation of $d$. Note that the way we code derivations is a special case of the coding of trees introduced in section 25.12, so the primitive recursive function $\mathrm{SubtreeSeq}(d)$ gives a sequence of Gödel numbers of all sub-derivations of $d$ (of length a most $d$). So we can define

$$\mathrm{Assum}(x, d, n) \Leftrightarrow (\exists i < d)\ (\mathrm{SubtreeSeq}(d))_i = \langle 0, x, n\rangle.$$

2. We want to show that $\text{Discharge}(x, d, n)$, which holds if all assumptions with label $n$ in the derivation with Gödel number $d$ all are the formula with Gödel number $x$. But this relation holds iff $(\forall y < d)\ (\text{Assum}(y, d, n) \to y = x)$.

□

**Proposition 31.19.** *The relation* $\text{Correct}(d)$ *which holds if the last inference in the derivation* $\delta$ *with Gödel number* $d$ *is correct, is primitive recursive.*

*Proof.* Here we have to show that for each rule of inference $R$ the relation $\text{FollowsBy}_R(d)$ is primitive recursive, where $\text{FollowsBy}_R(d)$ holds iff $d$ is the Gödel number of derivation $\delta$, and the end-formula of $\delta$ follows by a correct application of $R$ from the immediate sub-derivations of $\delta$.

A simple case is that of the $\wedge$Intro rule. If $\delta$ ends in a correct $\wedge$Intro inference, it looks like this:

$$\dfrac{\begin{array}{cc}\vdots & \vdots \\ A & B\end{array}}{A \wedge B}\ \wedge\text{Intro}$$

Then the Gödel number $d$ of $\delta$ is $\langle 2, d_1, d_2, {}^{\ulcorner}(A \wedge B)^{\urcorner}, 0, k\rangle$ where $\text{EndFmla}(d_1) = {}^{\ulcorner}A^{\urcorner}$, $\text{EndFmla}(d_2) = {}^{\ulcorner}B^{\urcorner}$, $n = 0$, and $k = 1$. So we can define $\text{FollowsBy}_{\wedge\text{Intro}}(d)$ as

$$\begin{aligned}&(d)_0 = 2 \wedge \text{DischargeLabel}(d) = 0 \wedge \text{LastRule}(d) = 1 \wedge \\ &\quad \text{EndFmla}(d) = {}^{\ulcorner}({}^{\urcorner} \frown \text{EndFmla}((d)_1) \frown {}^{\ulcorner}\wedge{}^{\urcorner} \frown \text{EndFmla}((d)_2) \frown {}^{\ulcorner})^{\urcorner}.\end{aligned}$$

Another simple example if the $=$Intro rule. Here the premise is an empty derivation, i.e., $(d)_1 = 0$, and no discharge label, i.e., $n = 0$. However, $\varphi$ must be of the form $t = t$, for a closed term $t$. Here, a primitive recursive definition is

$$\begin{aligned}&(d)_0 = 1 \wedge (d)_1 = 0 \wedge \text{DischargeLabel}(d) = 0 \wedge \\ &\qquad (\exists t < d)\ (\text{ClTerm}(t) \wedge \text{EndFmla}(d) = {}^{\ulcorner}={}^{\urcorner}({}^{\urcorner} \frown t \frown {}^{\ulcorner},{}^{\urcorner} \frown t \frown {}^{\ulcorner})^{\urcorner})\end{aligned}$$

For a more complicated example, $\text{FollowsBy}_{\to\text{Intro}}(d)$ holds iff the end-formula of $\delta$ is of the form $(\varphi \to \psi)$, where the end-formula of $\delta_1$ is $\psi$, and any assumption in $\delta$ labelled $n$ is of the form $\varphi$. We can express this primitive recursively by

$$\begin{aligned}&(d)_0 = 1 \wedge \\ &\qquad (\exists a < d)\ (\text{Discharge}(a, (d)_1, \text{DischargeLabel}(d)) \wedge \\ &\qquad\quad \text{EndFmla}(d) = ({}^{\ulcorner}({}^{\urcorner} \frown a \frown {}^{\ulcorner}\to{}^{\urcorner} \frown \text{EndFmla}((d)_1) \frown {}^{\ulcorner})^{\urcorner}))\end{aligned}$$

(Think of $a$ as the Gödel number of $\varphi$).

For another example, consider $\exists$Intro. Here, the last inference in $\delta$ is correct iff there is a formula $\varphi$, a closed term $t$ and a variable $x$ such that $\varphi[t/x]$ is the end-formula of the derivation $\delta_1$ and $\exists x\, \varphi$ is the conclusion of the last inference. So, $\text{FollowsBy}_{\exists\text{Intro}}(d)$ holds iff

$$\begin{aligned}(d)_0 = 1 \wedge \text{DischargeLabel}(d) = 0 \wedge \\ (\exists a < d)\ (\exists x < d)\ (\exists t < d)\ (\text{ClTerm}(t) \wedge \text{Var}(x) \wedge \\ \text{Subst}(a, t, x) = \text{EndFmla}((d)_1) \wedge \text{EndFmla}(d) = ({}^{\#}\exists^{\#} \frown x \frown a))\end{aligned}$$

We then define $\text{Correct}(d)$ as

$$\begin{aligned}\text{Sent}(\text{EndFmla}(d)) \wedge (\text{LastRule}(d) = 1 \wedge \text{FollowsBy}_{\wedge\text{Intro}}(d)) \vee \cdots \vee \\ (\text{LastRule}(d) = 16 \wedge \text{FollowsBy}_{=\text{Elim}}(d)) \vee \\ (\exists n < d)\ (\exists x < d)\ (d = \langle 0, x, n\rangle)\end{aligned}$$

The first line ensures that the end-formula of $d$ is a sentence. The last line covers the case where $d$ is just an assumption. □

**Proposition 31.20.** *The relation* $\text{Deriv}(d)$ *which holds if $d$ is the Gödel number of a correct derivation $\delta$, is primitive recursive.*

*Proof.* A derivation $\delta$ is correct if every one of its inferences is a correct application of a rule, i.e., if every one of its sub-derivations ends in a correct inference. So, $\text{Deriv}(d)$ iff

$$(\forall i < \text{len}(\text{SubtreeSeq}(d)))\ \text{Correct}((\text{SubtreeSeq}(d))_i)$$

□

**Proposition 31.21.** *The relation* $\text{OpenAssum}(z, d)$ *that holds if $z$ is the Gödel number of an undischarged assumption $\varphi$ of the derivation $\delta$ with Gödel number $d$, is primitive recursive.*

*Proof.* An occurrence of an assumption is discharged if it occurs with label $n$ in a sub-derivation of $\delta$ that ends in a rule with discharge label $n$. So $\varphi$ is an undischarged assumption of $\delta$ if at least one of its occurrences is not discharged in $\delta$. We must be careful: $\delta$ may contain both discharged and undischarged occurrences of $\varphi$.

Consider a sequence $\delta_0, \ldots, \delta_k$ where $\delta_0 = d$, $\delta_k$ is the assumption $[\varphi]^n$ (for some $n$), and $\delta_i$ is an immediate sub-derivation of $\delta_{i+1}$. If such a sequence exists in which no $\delta_i$ ends in an inference with discharge label $n$, then $\varphi$ is an undischarged assumption of $\delta$.

The primitive recursive function $\text{SubtreeSeq}(d)$ provides us with a sequence of Gödel numbers of all sub-derivations of $\delta$. Any sequence of Gödel numbers

of sub-derivations of $\delta$ is a subsequence of it. Being a subsequence of is a primitive recursive relation: $\text{Subseq}(s, s')$ holds iff $(\forall i < \text{len}(s))\ \exists j < \text{len}(s')\ (s)_i = (s)_j$. Being an immediate sub-derivation is as well: $\text{Subderiv}(d, d')$ iff $(\exists j < (d')_0)\ d = (d')_j$. So we can define $\text{OpenAssum}(z, d)$ by

$$\begin{aligned}(\exists s < \text{SubtreeSeq}(d))\ (\text{Subseq}(s, \text{SubtreeSeq}(d)) \wedge (s)_0 = d \wedge \\ (\exists n < d)\ ((s)_{\text{len}(s) \dot{-} 1} = \langle 0, z, n\rangle \wedge \\ (\forall i < (\text{len}(s) \dot{-} 1))\ (\text{Subderiv}((s)_i, (s)_{i+1})] \wedge \\ \text{DischargeLabel}((s)_{i+1}) \neq n))).\end{aligned}$$

□

**Proposition 31.22.** *Suppose $\Gamma$ is a primitive recursive set of sentences. Then the relation $\text{Prf}_\Gamma(x, y)$ expressing "$x$ is the code of a derivation $\delta$ of $\varphi$ from undischarged assumptions in $\Gamma$ and $y$ is the Gödel number of $\varphi$" is primitive recursive.*

*Proof.* Suppose "$y \in \Gamma$" is given by the primitive recursive predicate $R_\Gamma(y)$. We have to show that $\text{Prf}_\Gamma(x, y)$ which holds iff $y$ is the Gödel number of a sentence $\varphi$ and $x$ is the code of a natural deduction derivation with end formula $\varphi$ and all undischarged assumptions in $\Gamma$ is primitive recursive.

By Proposition 31.20, the property $\text{Deriv}(x)$ which holds iff $x$ is the Gödel number of a correct derivation $\delta$ in natural deduction is primitive recursive. Thus we can define $\text{Prf}_\Gamma(x, y)$ by

$$\begin{aligned}\text{Deriv}(x) \wedge \text{EndFmla}(x) = y \wedge \\ (\forall z < x)\ (\text{OpenAssum}(z, x) \to R_\Gamma(z))\end{aligned}$$

□

## 31.8 Axiomatic Derivations

In order to arithmetize axiomatic derivations, we must represent derivations as numbers. Since derivations are simply sequences of formulas, the obvious approach is to code every derivation as the code of the sequence of codes of formulas in it.

**Definition 31.23.** If $\delta$ is an axiomatic derivation consisting of formulas $\varphi_1$, $\ldots$, $\varphi_n$, then ${}^\ulcorner\delta^\urcorner$ is

$$\langle {}^\ulcorner\varphi_1{}^\urcorner, \ldots, {}^\ulcorner\varphi_n{}^\urcorner\rangle.$$

**Example 31.24.** Consider the very simple derivation

1. $\psi \to (\psi \vee \varphi)$
2. $(\psi \to (\psi \vee \varphi)) \to (\varphi \to (\psi \to (\psi \vee \varphi)))$
3. $\varphi \to (\psi \to (\psi \vee \varphi))$

The Gödel number of this derivation would simply be

$$\langle {}^{\ulcorner}\psi \to (\psi \vee \varphi)^{\urcorner}, {}^{\ulcorner}(\psi \to (\psi \vee \varphi)) \to (\varphi \to (\psi \to (\psi \vee \varphi)))^{\urcorner}, {}^{\ulcorner}\varphi \to (\psi \to (\psi \vee \varphi))^{\urcorner} \rangle.$$

Having settled on a representation of derivations, we must also show that we can manipulate such derivations primitive recursively, and express their essential properties and relations so. Some operations are simple: e.g., given a Gödel number $d$ of a derivation, $(d)_{\mathrm{len}(d)-1}$ gives us the Gödel number of its end-formula. Some are much harder. We'll at least sketch how to do this. The goal is to show that the relation "$\delta$ is a derivation of $\varphi$ from $\Gamma$" is primitive recursive on the Gödel numbers of $\delta$ and $\varphi$.

**Proposition 31.25.** *The following relations are primitive recursive:*

1. *$\varphi$ is an axiom.*
2. *The ith line in $\delta$ is justified by modus ponens*
3. *The ith line in $\delta$ is justified by* QR.
4. *$\delta$ is a correct derivation.*

*Proof.* We have to show that the corresponding relations between Gödel numbers of formulas and Gödel numbers of derivations are primitive recursive.

1. We have a given list of axiom schemas, and $\varphi$ is an axiom if it is of the form given by one of these schemas. Since the list of schemas is finite, it suffices to show that we can test primitive recursively, for each axiom schema, if $\varphi$ is of that form. For instance, consider the axiom schema

   $$\psi \to (\chi \to \psi).$$

   $\varphi$ is an instance of this axiom schema if there are formulas $\psi$ and $\chi$ such that we obtain $\varphi$ when we concatenate ( with $\psi$ with $\to$ with ( with $\chi$ with $\to$ with $\psi$ and with )). We can test the corresponding property of the Gödel number $n$ of $\varphi$, since concatenation of sequences is primitive recursive, and the Gödel numbers of $\psi$ and $C$ must be smaller than the Gödel number of $\varphi$, since when the relation holds, both $\psi$ and $\chi$ are sub-formulas of $\varphi$. Hence, we can define

   $$\begin{aligned}\mathrm{IsAx}_{\psi\to(\chi\to\psi)}(n) \Leftrightarrow (\exists b < n)\ (\exists c < n)\ (\mathrm{Sent}(b) \wedge \mathrm{Sent}(c) \wedge \\ n = {}^{\ulcorner}({}^{\urcorner} \frown b \frown {}^{\ulcorner}\to{}^{\urcorner} \frown {}^{\ulcorner}({}^{\urcorner} \frown c \frown {}^{\ulcorner}\to{}^{\urcorner} \frown b \frown {}^{\ulcorner})^{\urcorner}).\end{aligned}$$

   If we have such a definition for each axiom schema, their disjunction defines the property $\mathrm{IsAx}(n)$, "$n$ is the Gödel number of an axiom."

2. The $i$th line in $\delta$ is justified by modus ponens iff there are lines $j$ and $k < i$ where the sentence on line $j$ is some formula $\varphi$, the sentence on line $k$ is $\varphi \to \psi$, and the sentence on line $i$ is $\psi$.

$$\mathrm{MP}(d,i) \Leftrightarrow (\exists j < i)\ (\exists k < i)\ (d)_k = {}^{\#}({}^{\#} \frown (d)_j \frown {}^{\#}{\to}^{\#} \frown (d)_i \frown {}^{\#})^{\#}$$

   Since bounded quantification, concatenation, and $=$ are primitive recursive, this defines a primitive recursive relation.

3. A line in $\delta$ is justified by QR if it is of the form $\psi \to \forall x\, \varphi(x)$, a preceding line is $\psi \to \varphi(c)$ for some constant symbol $c$, and $c$ does on occur in $\psi$. This is the case iff

   a) there is a sentence $\psi$ and
   b) a formula $\varphi(x)$ with a single variable $x$ free so that
   c) line $i$ contains $\psi \to \forall x\, \varphi(x)$
   d) some line $j < i$ contains $\psi \to \varphi[c/x]$ for a constant $c$
   e) which does not occur in $\psi$.

   All of these can be tested primitive recursively, since the Gödel numbers of $\psi$, $\varphi(x)$, and $x$ are less than the Gödel number of the formula on line $i$, and that of $a$ less than the Gödel number of the formula on line $j$:

$$\begin{aligned}\mathrm{QR}_1(d,i) \Leftrightarrow (\exists b < (d)_i)\ (\exists x < (d)_i)\ (\exists a < (d)_i)\ (\exists c < (d)_j)\ (&\\ \mathrm{Var}(x) \wedge \mathrm{Const}(c) \wedge&\\ (d)_i = {}^{\#}({}^{\#} \frown b \frown {}^{\#}{\to}^{\#} \frown {}^{\#}{\forall}^{\#} \frown x \frown a \frown {}^{\#})^{\#} \wedge&\\ (d)_j = {}^{\#}({}^{\#} \frown b \frown {}^{\#}{\to}^{\#} \frown \mathrm{Subst}(a,c,x) \frown {}^{\#})^{\#} \wedge&\\ \mathrm{Sent}(b) \wedge \mathrm{Sent}(\mathrm{Subst}(a,c,x)) \wedge (\forall k < \mathrm{len}(b))\ (b)_k \neq (c)_0)&\end{aligned}$$

   Here we assume that $c$ and $x$ are the Gödel numbers of the variable and constant considered as terms (i.e., not their symbol codes). We test that $x$ is the only free variable of $\varphi(x)$ by testing if $\varphi(x)[c/x]$ is a sentence, and ensure that $c$ does not occur in $\psi$ by requiring that every symbol of $\psi$ is different from $c$.

   We leave the other version of QR as an exercise.

4. $d$ is the Gödel number of a correct derivation iff every line in it is an axiom, or justified by modus ponens or QR. Hence:

$$\mathrm{Deriv}(d) \Leftrightarrow (\forall i < \mathrm{len}(d))\ (\mathrm{IsAx}((d)_i) \vee \mathrm{MP}(d,i) \vee \mathrm{QR}(d,i))$$

□

**Proposition 31.26.** *Suppose $\Gamma$ is a primitive recursive set of sentences. Then the relation $\mathrm{Prf}_\Gamma(x, y)$ expressing "$x$ is the code of a derivation $\delta$ of $\varphi$ from $\Gamma$ and $y$ is the Gödel number of $\varphi$" is primitive recursive.*

*Proof.* Suppose "$y \in \Gamma$" is given by the primitive recursive predicate $R_\Gamma(y)$. We have to show that the relation $\mathrm{Prf}_\Gamma(x, y)$ is primitive recursive, where $\mathrm{Prf}_\Gamma(x, y)$ holds iff $y$ is the Gödel number of a sentence $\varphi$ and $x$ is the code of a derivation of $\varphi$ from $\Gamma$.

By the previous proposition, the property $\mathrm{Deriv}(x)$ which holds iff $x$ is the code of a correct derivation $\delta$ is primitive recursive. However, that definition did not take into account the set $\Gamma$ as an additional way to justify lines in the derivation. Our primitive recursive test of whether a line is justified by QR also left out of consideration the requirement that the constant $c$ is not allowed to occur in $\Gamma$. It is possible to amend our definition so that it takes into account $\Gamma$ directly, but it is easier to use Deriv and the deduction theorem. $\Gamma \vdash \varphi$ iff there is some finite list of sentences $\psi_1, \ldots, \psi_n \in \Gamma$ such that $\{\psi_1, \ldots, \psi_n\} \vdash \varphi$. And by the deduction theorem, this is the case if $\vdash (\psi_1 \to (\psi_2 \to \cdots (\psi_n \to \varphi) \cdots))$. Whether a sentence with Gödel number $z$ is of this form can be tested primitive recursively. So, instead of considering $x$ as the Gödel number of a derivation of the sentence with Gödel number $y$ *from* $\Gamma$, we consider $x$ as the Gödel number of a derivation of a nested conditional of the above form from $\emptyset$.

First, if we have a sequence of sentences, we can primitive recursively form the conditional with all these sentences as antecedents and given sentence as consequent:

$$\begin{aligned}
\mathrm{hCond}(s, y, 0) &= y \\
\mathrm{hCond}(s, y, n+1) &= {}^\natural({}^\# \frown (s)_n \frown {}^\natural\to{}^\# \frown \mathrm{Cond}(s, y, n) \frown {}^\natural){}^\# \\
\mathrm{Cond}(s, y) &= \mathrm{hCond}(s, y, \mathrm{len}(s))
\end{aligned}$$

So we can define $\mathrm{Prf}_\Gamma(x, y)$ by

$$\begin{aligned}
\mathrm{Prf}_\Gamma(x, y) \Leftrightarrow (\exists s < \mathrm{sequenceBound}(x, x))\ ( & \\
& (x)_{\mathrm{len}(x)-1} = \mathrm{Cond}(s, y) \wedge \\
& (\forall i < \mathrm{len}(s))\ (s)_i \in \Gamma \wedge \\
& \mathrm{Deriv}(x)).
\end{aligned}$$

The bound on $s$ is given by considering that each $(s)_i$ is the Gödel number of a subformula of the last line of the derivation, i.e., is less than $(x)_{\mathrm{len}(x)-1}$. The number of antecedents $\psi \in \Gamma$, i.e., the length of $s$, is less than the length of the last line of $x$. □

## Problems

**Problem 31.1.** Show that the function flatten$(z)$, which turns the sequence $\langle {}^{\ulcorner}t_1{}^{\urcorner}, \ldots, {}^{\ulcorner}t_n{}^{\urcorner}\rangle$ into ${}^{\ulcorner}t_1, \ldots, t_n{}^{\urcorner}$, is primitive recursive.

**Problem 31.2.** Give a detailed proof of Proposition 31.8 along the lines of the first proof of Proposition 31.5

**Problem 31.3.** Give a detailed proof of Proposition 31.8 along the lines of the alternate proof of Proposition 31.5

**Problem 31.4.** Prove Proposition 31.9. You may make use of the fact that any substring of a formula which is a formula is a sub-formula of it.

**Problem 31.5.** Prove Proposition 31.12

**Problem 31.6.** Define the following relations as in Proposition 31.14:

1. FollowsBy$_{\wedge \mathrm{R}}(s, s', s'')$,
2. FollowsBy$_{=}(s, s')$,
3. FollowsBy$_{\forall \mathrm{R}}(s, s')$.

**Problem 31.7.** Define the following relations as in Proposition 31.18:

1. FollowsBy$_{\rightarrow \mathrm{Elim}}(d)$,
2. FollowsBy$_{=\mathrm{Elim}}(d)$,
3. FollowsBy$_{\vee \mathrm{Elim}}(d)$,
4. FollowsBy$_{\forall \mathrm{Intro}}(d)$.

For the last one, you will have to also show that you can test primitive recursively if the the last inference of the derivation with Gödel number $d$ satisfies the eigenvariable condition, i.e., the eigenvariable $a$ of the $\forall$Intro inference occurs neither in the end-formula of $d$ nor in an open assumption of $d$. You may use the primitive recursive predicate OpenAssum from Proposition 31.21 for this.

**Problem 31.8.** Define the following relations as in Proposition 31.25:

1. IsAx$_{\varphi \rightarrow (\psi \rightarrow (\varphi \wedge \psi))}(n)$,
2. IsAx$_{\forall x\, \varphi(x) \rightarrow \varphi(t)}(n)$,
3. QR$_2(d, i)$ (for the other version of QR).

# Chapter 32

# Representability in Q

## 32.1 Introduction

The incompleteness theorems apply to theories in which basic facts about computable functions can be expressed and proved. We will describe a very minimal such theory called "**Q**" (or, sometimes, "Robinson's $Q$," after Raphael Robinson). We will say what it means for a function to be *representable* in **Q**, and then we will prove the following:

> A function is representable in **Q** if and only if it is computable.

For one thing, this provides us with another model of computability. But we will also use it to show that the set $\{\varphi : \mathbf{Q} \vdash \varphi\}$ is not decidable, by reducing the halting problem to it. By the time we are done, we will have proved much stronger things than this.

The language of **Q** is the language of arithmetic; **Q** consists of the following axioms (to be used in conjunction with the other axioms and rules of first-order logic with identity predicate):

$$\forall x \, \forall y \, (x' = y' \to x = y) \tag{$Q_1$}$$
$$\forall x \, 0 \neq x' \tag{$Q_2$}$$
$$\forall x \, (x \neq 0 \to \exists y \, x = y') \tag{$Q_3$}$$
$$\forall x \, (x + 0) = x \tag{$Q_4$}$$
$$\forall x \, \forall y \, (x + y') = (x + y)' \tag{$Q_5$}$$
$$\forall x \, (x \times 0) = 0 \tag{$Q_6$}$$
$$\forall x \, \forall y \, (x \times y') = ((x \times y) + x) \tag{$Q_7$}$$
$$\forall x \, \forall y \, (x < y \leftrightarrow \exists z \, (z' + x) = y) \tag{$Q_8$}$$

For each natural number $n$, define the numeral $\overline{n}$ to be the term $0^{\prime\cdots\prime}$ where there are $n$ tick marks in all. So, $\overline{0}$ is the constant symbol $0$ by itself, $\overline{1}$ is $0'$, $\overline{2}$ is $0''$, etc.

As a theory of arithmetic, **Q** is *extremely* weak; for example, you can't even prove very simple facts like $\forall x\, x \neq x'$ or $\forall x \forall y\, (x + y) = (y + x)$. But we will see that much of the reason that **Q** is so interesting is *because* it is so weak. In fact, it is just barely strong enough for the incompleteness theorem to hold. Another reason **Q** is interesting is because it has a *finite* set of axioms.

A stronger theory than **Q** (called *Peano arithmetic* **PA**) is obtained by adding a schema of induction to **Q**:

$$(\varphi(0) \wedge \forall x\, (\varphi(x) \to \varphi(x'))) \to \forall x\, \varphi(x)$$

where $\varphi(x)$ is any formula. If $\varphi(x)$ contains free variables other than $x$, we add universal quantifiers to the front to bind all of them (so that the corresponding instance of the induction schema is a sentence). For instance, if $\varphi(x, y)$ also contains the variable $y$ free, the corresponding instance is

$$\forall y\, ((\varphi(0) \wedge \forall x\, (\varphi(x) \to \varphi(x'))) \to \forall x\, \varphi(x))$$

Using instances of the induction schema, one can prove much more from the axioms of **PA** than from those of **Q**. In fact, it takes a good deal of work to find "natural" statements about the natural numbers that can't be proved in Peano arithmetic!

**Definition 32.1.** A function $f(x_0, \ldots, x_k)$ from the natural numbers to the natural numbers is said to be *representable in* **Q** if there is a formula $\varphi_f(x_0, \ldots, x_k, y)$ such that whenever $f(n_0, \ldots, n_k) = m$, **Q** proves

1. $\varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{m})$
2. $\forall y\, (\varphi_f(\overline{n_0}, \ldots, \overline{n_k}, y) \to \overline{m} = y)$.

There are other ways of stating the definition; for example, we could equivalently require that **Q** proves $\forall y\, (\varphi_f(\overline{n_0}, \ldots, \overline{n_k}, y) \leftrightarrow y = \overline{m})$.

**Theorem 32.2.** *A function is representable in* **Q** *if and only if it is computable.*

There are two directions to proving the theorem. The left-to-right direction is fairly straightforward once arithmetization of syntax is in place. The other direction requires more work. Here is the basic idea: we pick "general recursive" as a way of making "computable" precise, and show that every general recursive function is representable in **Q**. Recall that a function is general recursive if it can be defined from zero, the successor function succ, and the projection functions $P^n_i$, using composition, primitive recursion, and regular minimization. So one way of showing that every general recursive function is representable in **Q** is to show that the basic functions are representable, and whenever some functions are representable, then so are the functions defined from them using composition, primitive recursion, and regular minimization. In other words, we might show that the basic functions are representable, and

that the representable functions are "closed under" composition, primitive recursion, and regular minimization. This guarantees that every general recursive function is representable.

It turns out that the step where we would show that representable functions are closed under primitive recursion is hard. In order to avoid this step, we show first that in fact we can do without primitive recursion. That is, we show that every general recursive function can be defined from basic functions using composition and regular minimization alone. To do this, we show that primitive recursion can actually be done by a specific regular minimization. However, for this to work, we have to add some additional basic functions: addition, multiplication, and the characteristic function of the identity relation $\chi_=$. Then, we can prove the theorem by showing that all of *these* basic functions are representable in **Q**, and the representable functions are closed under composition and regular minimization.

## 32.2 Functions Representable in Q are Computable

**Lemma 32.3.** *Every function that is representable in* **Q** *is computable.*

*Proof.* Let's first give the intuitive idea for why this is true. If $f(x_0, \ldots, x_k)$ is representable in **Q**, there is a formula $\varphi(x_0, \ldots, x_k, y)$ such that

$$\mathbf{Q} \vdash \varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{m}) \quad \text{iff} \quad m = f(n_0, \ldots, n_k).$$

To compute $f$, we do the following. List all the possible derivations $\delta$ in the language of arithmetic. This is possible to do mechanically. For each one, check if it is a derivation of a formula of the form $\varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{m})$. If it is, $m$ must be $= f(n_0, \ldots, n_k)$ and we've found the value of $f$. The search terminates because $\mathbf{Q} \vdash \varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{f(n_0, \ldots, n_k)})$, so eventually we find a $\delta$ of the right sort.

This is not quite precise because our procedure operates on derivations and formulas instead of just on numbers, and we haven't explained exactly why "listing all possible derivations" is mechanically possible. But as we've seen, it is possible to code terms, formulas, and derivations by Gödel numbers. We've also introduced a precise model of computation, the general recursive functions. And we've seen that the relation $\mathrm{Prf}_{\mathbf{Q}}(d, y)$, which holds iff $d$ is the Gödel number of a derivation of the formula with Gödel number $x$ from the axioms of **Q**, is (primitive) recursive. Other primitive recursive functions we'll need are num (Proposition 31.6) and Subst (Proposition 31.11). From these, it is possible to define $f$ by minimization; thus, $f$ is recursive.

First, define

$$A(n_0, \ldots, n_k, m) = \\ \text{Subst}(\text{Subst}(\ldots \text{Subst}(^{\#}\varphi_f{}^{\#}, \text{num}(n_0), {}^{\#}x_0{}^{\#}), \\ \ldots), \text{num}(n_k), {}^{\#}x_k{}^{\#}), \text{num}(m), {}^{\#}y^{\#})$$

This looks complicated, but it's just the function $A(n_0, \ldots, n_k, m) = {}^{\#}\varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{m})^{\#}$.

Now, consider the relation $R(n_0, \ldots, n_k, s)$ which holds if $(s)_0$ is the Gödel number of a derivation from **Q** of $\varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{(s)_1})$:

$$R(n_0, \ldots, n_k, s) \quad \text{iff} \quad \text{Prf}_{\mathbf{Q}}((s)_0, A(n_0, \ldots, n_k, (s)_1)$$

If we can find an $s$ such that $R(n_0, \ldots, n_k, s)$ hold, we have found a pair of numbers—$(s)_0$ and $(s_1)$—such that $(s)_0$ is the Gödel number of a derivation of $A_f(\overline{n_0}, \ldots, \overline{n_k}, (s)_1)$. So looking for $s$ is like looking for the pair $d$ and $m$ in the informal proof. And a computable function that "looks for" such an $s$ can be defined by regular minimization. Note that $R$ is regular: for every $n_0, \ldots, n_k$, there is a derivation $\delta$ of $\mathbf{Q} \vdash \varphi_f(\overline{n_0}, \ldots, \overline{n_k}, \overline{f(n_0, \ldots, n_k)})$, so $R(n_0, \ldots, n_k, s)$ holds for $s = \langle {}^{\#}\delta^{\#}, f(n_0, \ldots, n_k)\rangle$. So, we can write $f$ as

$$f(n_0, \ldots, n_k) = (\mu s\ R(n_0, \ldots, n_k, s))_1.$$

□

## 32.3 The Beta Function Lemma

In order to show that we can carry out primitive recursion if addition, multiplication, and $\chi_=$ are available, we need to develop functions that handle sequences. (If we had exponentiation as well, our task would be easier.) When we had primitive recursion, we could define things like the "$n$-th prime," and pick a fairly straightforward coding. But here we do not have primitive recursion—in fact we want to show that we can do primitive recursion using minimization—so we need to be more clever.

**Lemma 32.4.** *There is a function $\beta(d, i)$ such that for every sequence $a_0, \ldots, a_n$ there is a number $d$, such that for every $i \leq n$, $\beta(d, i) = a_i$. Moreover, $\beta$ can be defined from the basic functions using just composition and regular minimization.*

Think of $d$ as coding the sequence $\langle a_0, \ldots, a_n\rangle$, and $\beta(d, i)$ returning the $i$-th element. (Note that this "coding" does *not* use the prower-of-primes coding we're already familiar with!). The lemma is fairly minimal; it doesn't say we can concatenate sequences or append elements, or even that we can *compute* $d$ from $a_0, \ldots, a_n$ using functions definable by composition and regular minimization. All it says is that there is a "decoding" function such that every sequence is "coded."

The use of the notation $\beta$ is Gödel's. To repeat, the hard part of proving the lemma is defining a suitable $\beta$ using the seemingly restricted resources, i.e., using just composition and minimization—however, we're allowed to use addition, multiplication, and $\chi_=$. There are various ways to prove this lemma, but one of the cleanest is still Gödel's original method, which used a number-theoretic fact called the Chinese Remainder theorem.

**Definition 32.5.** Two natural numbers $a$ and $b$ are *relatively prime* if their greatest common divisor is 1; in other words, they have no other divisors in common.

**Definition 32.6.** $a \equiv b \mod c$ means $c \mid (a - b)$, i.e., $a$ and $b$ have the same remainder when divided by $c$.

Here is the *Chinese Remainder theorem*:

**Theorem 32.7.** *Suppose $x_0, \ldots, x_n$ are (pairwise) relatively prime. Let $y_0, \ldots, y_n$ be any numbers. Then there is a number $z$ such that*

$$\begin{aligned} z &\equiv y_0 \mod x_0 \\ z &\equiv y_1 \mod x_1 \\ &\vdots \\ z &\equiv y_n \mod x_n. \end{aligned}$$

Here is how we will use the Chinese Remainder theorem: if $x_0, \ldots, x_n$ are bigger than $y_0, \ldots, y_n$ respectively, then we can take $z$ to code the sequence $\langle y_0, \ldots, y_n \rangle$. To recover $y_i$, we need only divide $z$ by $x_i$ and take the remainder. To use this coding, we will need to find suitable values for $x_0, \ldots, x_n$.

A couple of observations will help us in this regard. Given $y_0, \ldots, y_n$, let

$$j = \max(n, y_0, \ldots, y_n) + 1,$$

and let

$$\begin{aligned} x_0 &= 1 + j! \\ x_1 &= 1 + 2 \cdot j! \\ x_2 &= 1 + 3 \cdot j! \\ &\vdots \\ x_n &= 1 + (n+1) \cdot j! \end{aligned}$$

Then two things are true:

1. $x_0, \ldots, x_n$ are relatively prime.
2. For each $i$, $y_i < x_i$.

To see that (1) is true, note that if $p$ is a prime number and $p \mid x_i$ and $p \mid x_k$, then $p \mid 1 + (i+1)j!$ and $p \mid 1 + (k+1)j!$. But then $p$ divides their difference,

$$(1 + (i+1)j!) - (1 + (k+1)j!) = (i-k)j!.$$

Since $p$ divides $1 + (i+1)j!$, it can't divide $j!$ as well (otherwise, the first division would leave a remainder of 1). So $p$ divides $i - k$, since $p$ divides $(i-k)j!$. But $|i-k|$ is at most $n$, and we have chosen $j > n$, so this implies that $p \mid j!$, again a contradiction. So there is no prime number dividing both $x_i$ and $x_k$. Clause (2) is easy: we have $y_i < j < j! < x_i$.

Now let us prove the $\beta$ function lemma. Remember that we can use 0, successor, plus, times, $\chi_=$, projections, and any function defined from them using composition and minimization applied to regular functions. We can also use a relation if its characteristic function is so definable. As before we can show that these relations are closed under boolean combinations and bounded quantification; for example:

1. $\text{not}(x) = \chi_=(x, 0)$
2. $(\min x \leq z)\, R(x, y) = \mu x\ (R(x, y) \vee x = z)$
3. $(\exists x \leq z)\ R(x, y) \Leftrightarrow R((\min x \leq z)\, R(x, y), y)$

We can then show that all of the following are also definable without primitive recursion:

1. The pairing function, $J(x, y) = \frac{1}{2}[(x + y)(x + y + 1)] + x$
2. Projections

   $$K(z) = (\min x \leq q)\, (\exists y \leq z\, [z = J(x, y)])$$

   and

   $$L(z) = (\min y \leq q)\, (\exists x \leq z\, [z = J(x, y)]).$$

3. $x < y$
4. $x \mid y$
5. The function $\text{rem}(x, y)$ which returns the remainder when $y$ is divided by $x$

Now define

$$\beta^*(d_0, d_1, i) = \text{rem}(1 + (i+1)d_1, d_0)$$

and

$$\beta(d, i) = \beta^*(K(d), L(d), i).$$

This is the function we need. Given $a_0, \ldots, a_n$, as above, let

$$j = \max(n, a_0, \ldots, a_n) + 1,$$

and let $d_1 = j!$. By the observations above, we know that $1 + d_1, 1 + 2d_1, \ldots, 1 + (n+1)d_1$ are relatively prime and all are bigger than $a_0, \ldots, a_n$. By the Chinese Remainder theorem there is a value $d_0$ such that for each $i$,

$$d_0 \equiv a_i \mod (1 + (i+1)d_1)$$

and so (because $d_1$ is greater than $a_i$),

$$a_i = \text{rem}(1 + (i+1)d_1, d_0).$$

Let $d = J(d_0, d_1)$. Then for each $i \leq n$, we have

$$\begin{aligned} \beta(d, i) &= \beta^*(d_0, d_1, i) \\ &= \text{rem}(1 + (i+1)d_1, d_0) \\ &= a_i \end{aligned}$$

which is what we need. This completes the proof of the $\beta$-function lemma.

## 32.4 Simulating Primitive Recursion

Now we can show that definition by primitive recursion can be "simulated" by regular minimization using the beta function. Suppose we have $f(\vec{x})$ and $g(\vec{x}, y, z)$. Then the function $h(x, \vec{z})$ defined from $f$ and $g$ by primitive recursion is

$$\begin{aligned} h(\vec{x}, y) &= f(\vec{z}) \\ h(\vec{x}, y+1) &= g(\vec{x}, y, h(\vec{x}, y)). \end{aligned}$$

We need to show that $h$ can be defined from $f$ and $g$ using just composition and regular minimization, using the basic functions and functions defined from them using composition and regular minimization (such as $\beta$).

**Lemma 32.8.** *If $h$ can be defined from $f$ and $g$ using primitive recursion, it can be defined from $f$, $g$, the functions* zero, succ, $P^n_i$, add, mult, $\chi_=$, *using composition and regular minimization.*

*Proof.* First, define an auxiliary function $\hat{h}(\vec{x}, y)$ which returns the least number $d$ such that $d$ codes a sequence which satisfies

1. $(d)_0 = f(\vec{x})$, and
2. for each $i < x$, $(d)_{i+1} = g(\vec{x}, i, (d)_i)$,

where now $(d)_i$ is short for $\beta(d,i)$. In other words, $\hat{h}$ returns the sequence $\langle h(\vec{x},0), h(\vec{x},1), \ldots, h(\vec{x},y)\rangle$. We can write $\hat{h}$ as

$$\hat{h}(\vec{x},y) = \mu d\ (\beta(d,0) = f(\vec{x}) \wedge (\forall i < y)\ \beta(d,i+1) = g(\vec{x},i,\beta(d,i))).$$

Note: no primitive recursion is needed here, just minimization. The function we minimize is regular because of the beta function lemma Lemma 32.4.

But now we have

$$h(\vec{x},y) = \beta(\hat{h}(\vec{x},y),y),$$

so $h$ can be defined from the basic functions using just composition and regular minimization. □

## 32.5 Basic Functions are Representable in Q

First we have to show that all the basic functions are representable in **Q**. In the end, we need to show how to assign to each $k$-ary basic function $f(x_0,\ldots,x_{k-1})$ a formula $\varphi_f(x_0,\ldots,x_{k-1},y)$ that represents it.

We will be able to represent zero, successor, plus, times, the characteristic function for equality, and projections. In each case, the appropriate representing function is entirely straightforward; for example, zero is represented by the formula $y = 0$, successor is represented by the formula $x_0' = y$, and addition is represented by the formula $(x_0 + x_1) = y$. The work involves showing that **Q** can prove the relevant sentences; for example, saying that addition is represented by the formula above involves showing that for every pair of natural numbers $m$ and $n$, **Q** proves

$$\overline{n} + \overline{m} = \overline{n+m} \text{ and}$$
$$\forall y\,((\overline{n} + \overline{m}) = y \to y = \overline{n+m}).$$

**Proposition 32.9.** *The zero function* $\mathrm{zero}(x) = 0$ *is represented in* **Q** *by* $y = 0$.

**Proposition 32.10.** *The successor function* $\mathrm{succ}(x) = x + 1$ *is represented in* **Q** *by* $y = x'$.

**Proposition 32.11.** *The projection function* $P^n_i(x_0,\ldots,x_{n-1}) = x_i$ *is represented in* **Q** *by* $y = x_i$.

**Proposition 32.12.** *The characteristic function of* $=$,

$$\chi_=(x_0,x_1) = \begin{cases} 1 & \text{if } x_0 = x_1 \\ 0 & \text{otherwise} \end{cases}$$

*is represented in* **Q** *by*

$$(x_0 = x_1 \wedge y = \overline{1}) \vee (x_0 \neq x_1 \wedge y = \overline{0}).$$

The proof requires the following lemma.

**Lemma 32.13.** *Given natural numbers $n$ and $m$, if $n \neq m$, then $\mathbf{Q} \vdash \overline{n} \neq \overline{m}$.*

*Proof.* Use induction on $n$ to show that for every $m$, if $n \neq m$, then $Q \vdash \overline{n} \neq \overline{m}$.

In the base case, $n = 0$. If $m$ is not equal to 0, then $m = k + 1$ for some natural number $k$. We have an axiom that says $\forall x\, 0 \neq x'$. By a quantifier axiom, replacing $x$ by $\overline{k}$, we can conclude $0 \neq \overline{k}'$. But $\overline{k}'$ is just $\overline{m}$.

In the induction step, we can assume the claim is true for $n$, and consider $n + 1$. Let $m$ be any natural number. There are two possibilities: either $m = 0$ or for some $k$ we have $m = k + 1$. The first case is handled as above. In the second case, suppose $n + 1 \neq k + 1$. Then $n \neq k$. By the induction hypothesis for $n$ we have $\mathbf{Q} \vdash \overline{n} \neq \overline{k}$. We have an axiom that says $\forall x\, \forall y\, x' = y' \rightarrow x = y$. Using a quantifier axiom, we have $\overline{n}' = \overline{k}' \rightarrow \overline{n} = \overline{k}$. Using propositional logic, we can conclude, in $\mathbf{Q}$, $\overline{n} \neq \overline{k} \rightarrow \overline{n}' \neq \overline{k}'$. Using modus ponens, we can conclude $\overline{n}' \neq \overline{k}'$, which is what we want, since $\overline{k}'$ is $\overline{m}$. □

Note that the lemma does not say much: in essence it says that **Q** can prove that different numerals denote different objects. For example, **Q** proves $0'' \neq 0'''$. But showing that this holds in general requires some care. Note also that although we are using induction, it is induction *outside* of **Q**.

*Proof of Proposition 32.12.* If $n = m$, then $\overline{n}$ and $\overline{m}$ are the same term, and $\chi_=(n, m) = 1$. But $\mathbf{Q} \vdash (\overline{n} = \overline{m} \wedge \overline{1} = \overline{1})$, so it proves $\varphi_=(\overline{n}, \overline{m}, \overline{1})$. If $n \neq m$, then $\chi_=(n, m) = 0$. By Lemma 32.13, $\mathbf{Q} \vdash \overline{n} \neq \overline{m}$ and so also $(\overline{n} \neq \overline{m} \wedge \mathsf{o} = \mathsf{o})$. Thus $\mathbf{Q} \vdash \varphi_=(\overline{n}, \overline{m}, \overline{0})$.

For the second part, we also have two cases. If $n = m$, we have to show that that $\mathbf{Q} \vdash \forall(\varphi_=(\overline{n}, \overline{m}, y) \rightarrow y = \overline{1})$. Arguing informally, suppose $\varphi_=(\overline{n}, \overline{m}, y)$, i.e.,

$$(\overline{n} = \overline{n} \wedge y = \overline{1}) \vee (\overline{n} \neq \overline{n} \wedge y = \overline{0})$$

The left disjunct implies $y = \overline{1}$ by logic; the right contradicts $\overline{n} = \overline{n}$ which is provable by logic.

Suppose, on the other hand, that $n \neq m$. Then $\varphi_=(\overline{n}, \overline{m}, y)$ is

$$(\overline{n} = \overline{m} \wedge y = \overline{1}) \vee (\overline{n} \neq \overline{m} \wedge y = \overline{0})$$

Here, the left disjunct contradicts $\overline{n} \neq \overline{m}$, which is provable in **Q** by Lemma 32.13; the right disjunct entails $y = \overline{0}$. □

**Proposition 32.14.** *The addition function* $\mathrm{add}(x_0, x_1) = x_0 + x_1$ *is represented in* **Q** *by*

$$y = (x_0 + x_1).$$

**Lemma 32.15.** $\mathbf{Q} \vdash (\overline{n} + \overline{m}) = \overline{n + m}$

*Proof.* We prove this by induction on $m$. If $m = 0$, the claim is that $\mathbf{Q} \vdash (\overline{n} + \mathsf{o}) = \overline{n}$. This follows by axiom $Q_4$. Now suppose the claim for $m$; let's prove the claim for $m+1$, i.e., prove that $\mathbf{Q} \vdash (\overline{n} + \overline{m+1}) = \overline{n+m+1}$. Note that $\overline{m+1}$ is just $\overline{m}'$, and $\overline{n+m+1}$ is just $\overline{n+m}'$. By axiom $Q_5$, $\mathbf{Q} \vdash (\overline{n} + \overline{m}') = (\overline{n} + \overline{m})'$. By induction hypothesis, $\mathbf{Q} \vdash (\overline{n} + \overline{m}) = \overline{n+m}$. So $\mathbf{Q} \vdash (\overline{n} + \overline{m}') = \overline{n+m}'$. □

*Proof of Proposition 32.14.* The formula $\varphi_{\mathrm{add}}(x_0, x_1, y)$ representing add is $y = (x_0 + x_1)$. First we show that if $\mathrm{add}(n, m) = k$, then $\mathbf{Q} \vdash \varphi_{\mathrm{add}}(\overline{n}, \overline{m}, \overline{k})$, i.e., $\mathbf{Q} \vdash \overline{k} = (\overline{n} + \overline{m})$. But since $k = n + m$, $\overline{k}$ just is $\overline{n+m}$, and we've shown in Lemma 32.15 that $\mathbf{Q} \vdash (\overline{n} + \overline{m}) = \overline{n+m}$.

We also have to show that if $\mathrm{add}(n, m) = k$, then

$$\mathbf{Q} \vdash \forall y\, (\varphi_{\mathrm{add}}(\overline{n}, \overline{m}, y) \to y = \overline{k}).$$

Suppose we have $(\overline{n} + \overline{m}) = y$. Since

$$\mathbf{Q} \vdash (\overline{n} + \overline{m}) = \overline{n+m},$$

we can replace the left side with $\overline{n+m}$ and get $\overline{n+m} = y$, for arbitrary $y$. □

**Proposition 32.16.** *The multiplication function* $\mathrm{mult}(x_0, x_1) = x_0 \cdot x_1$ *is represented in* **Q** *by*

$$y = (x_0 \times x_1).$$

*Proof.* Exercise. □

**Lemma 32.17.** $\mathbf{Q} \vdash (\overline{n} \times \overline{m}) = \overline{n \cdot m}$

*Proof.* Exercise. □

Recall that we use $\times$ for the function symbol of the language of arithmetic, and $\cdot$ for the ordinary multiplication operation on numbers. So $\cdot$ can appear between expressions for numbers (such as in $m \cdot n$) while $\times$ appears only between terms of the language of arithmetic (such as in $(\overline{m} \times \overline{n})$). EVen more confusingly, $+$ is used for both the function symbol and the addition operation. When it appears between terms—e.g., in $(\overline{n} + \overline{m})$—it is the 2-place function symbol of the language of arithmetic, and when it appears between numbers—e.g., in $n + m$—it is the operation. This includes the case $\overline{n+m}$: this is the standard numeral corresponding to the number $n + m$.

## 32.6 Composition is Representable in Q

Suppose $h$ is defined by

$$h(x_0, \ldots, x_{l-1}) = f(g_0(x_0, \ldots, x_{l-1}), \ldots, g_{k-1}(x_0, \ldots, x_{l-1})).$$

where we have already found formulas $\varphi_f, \varphi_{g_0}, \ldots, \varphi_{g_{k-1}}$ representing the functions $f$, and $g_0, \ldots, g_{k-1}$, respectively. We have to find a formula $\varphi_h$ representing $h$.

Let's start with a simple case, where all functions are 1-place, i.e., consider $h(x) = f(g(x))$. If $\varphi_f(y, z)$ represents $f$, and $\varphi_g(x, y)$ represents $g$, we need a formula $\varphi_h(x, z)$ that represents $h$. Note that $h(x) = z$ iff there is a $y$ such that both $z = f(y)$ and $y = g(x)$. (If $h(x) = z$, then $g(x)$ is such a $y$; if such a $y$ exists, then since $y = g(x)$ and $z = f(y)$, $z = f(g(x))$.) This suggests that $\exists y\,(\varphi_g(x, y) \wedge \varphi_f(y, z))$ is a good candidate for $\varphi_h(x, z)$. We just have to verify that **Q** proves the relevant formulas.

**Proposition 32.18.** *If $h(n) = m$, then $\mathbf{Q} \vdash \varphi_h(\overline{n}, \overline{m})$.*

*Proof.* Suppose $h(n) = m$, i.e., $f(g(n)) = m$. Let $k = g(n)$. Then

$$\mathbf{Q} \vdash \varphi_g(\overline{n}, \overline{k})$$

since $\varphi_g$ represents $g$, and

$$\mathbf{Q} \vdash \varphi_f(\overline{k}, \overline{m})$$

since $\varphi_f$ represents $f$. Thus,

$$\mathbf{Q} \vdash \varphi_g(\overline{n}, \overline{k}) \wedge \varphi_f(\overline{k}, \overline{m})$$

and consequently also

$$\mathbf{Q} \vdash \exists y\,(\varphi_g(\overline{n}, y) \wedge \varphi_f(y, \overline{m})),$$

i.e., $\mathbf{Q} \vdash \varphi_h(\overline{n}, \overline{m})$. □

**Proposition 32.19.** *If $h(n) = m$, then $\mathbf{Q} \vdash \forall z\,(\varphi_h(\overline{n}, z) \to z = \overline{m})$.*

*Proof.* Suppose $h(n) = m$, i.e., $f(g(n)) = m$. Let $k = g(n)$. Then

$$\mathbf{Q} \vdash \forall y\,(\varphi_g(\overline{n}, y) \to y = \overline{k})$$

since $\varphi_g$ represents $g$, and

$$\mathbf{Q} \vdash \forall z\,(\varphi_f(\overline{k}, z) \to z = \overline{m})$$

since $\varphi_f$ represents $f$. Using just a little bit of logic, we can show that also

$$\mathbf{Q} \vdash \forall z\, (\exists y\, (\varphi_g(\overline{n}, y) \wedge \varphi_f(y, z)) \to z = \overline{m}).$$

i.e., $\mathbf{Q} \vdash \forall y\, (\varphi_h(\overline{n}, y) \to y = \overline{m})$. □

The same idea works in the more complex case where $f$ and $g_i$ have arity greater than 1.

**Proposition 32.20.** *If $\varphi_f(y_0, \ldots, y_{k-1}, z)$ represents $f(y_0, \ldots, y_{k-1})$ in **Q**, and $\varphi_{g_i}(x_0, \ldots, x_{l-1}, y)$ represents $g_i(x_0, \ldots, x_{l-1})$ in **Q**, then*

$$\begin{aligned}\exists y_0, \ldots\, \exists y_{k-1}\, (&\varphi_{g_0}(x_0, \ldots, x_{l-1}, y_0) \wedge \cdots \wedge \\ &\varphi_{g_{k-1}}(x_0, \ldots, x_{l-1}, y_{k-1}) \wedge \varphi_f(y_0, \ldots, y_{k-1}, z))\end{aligned}$$

*represents*

$$h(x_0, \ldots, x_{k-1}) = f(g_0(x_0, \ldots, x_{k-1}), \ldots, g_0(x_0, \ldots, x_{k-1})).$$

*Proof.* Exercise. □

## 32.7 Regular Minimization is Representable in Q

Let's consider unbounded search. Suppose $g(x, z)$ is regular and representable in **Q**, say by the formula $\varphi_g(x, z, y)$. Let $f$ be defined by $f(z) = \mu x\, [g(x, z) = 0]$. We would like to find a formula $\varphi_f(z, y)$ representing $f$. The value of $f(z)$ is that number $x$ which (a) satisfies $g(x, z) = 0$ and (b) is the least such, i.e., for any $w < x$, $g(w, z) \neq 0$. So the following is a natural choice:

$$\varphi_f(z, y) \equiv \varphi_g(y, z, 0) \wedge \forall w\, (w < y \to \neg \varphi_g(w, z, 0)).$$

In the general case, of course, we would have to replace $z$ with $z_0, \ldots, z_k$.

The proof, again, will involve some lemmas about things **Q** is strong enough to prove.

**Lemma 32.21.** *For every constant symbol $a$ and every natural number $n$,*

$$\mathbf{Q} \vdash (a' + \overline{n}) = (a + \overline{n})'.$$

*Proof.* The proof is, as usual, by induction on $n$. In the base case, $n = 0$, we need to show that **Q** proves $(a' + 0) = (a + 0)'$. But we have:

$$\mathbf{Q} \vdash (a' + 0) = a' \quad \text{by axiom } Q_4 \tag{32.1}$$

$$\mathbf{Q} \vdash (a + 0) = a \quad \text{by axiom } Q_4 \tag{32.2}$$

$$\mathbf{Q} \vdash (a + 0)' = a' \quad \text{by eq. (32.2)} \tag{32.3}$$

$$\mathbf{Q} \vdash (a' + 0) = (a + 0)' \quad \text{by eq. (32.1) and eq. (32.3)}$$

In the induction step, we can assume that we have shown that $\mathbf{Q} \vdash (a' + \overline{n}) = (a + \overline{n})'$. Since $\overline{n+1}$ is $\overline{n}'$, we need to show that $\mathbf{Q}$ proves $(a' + \overline{n}') = (a + \overline{n}')'$. We have:

$$\mathbf{Q} \vdash (a' + \overline{n}') = (a' + \overline{n})' \quad \text{by axiom } Q_5 \tag{32.4}$$
$$\mathbf{Q} \vdash (a' + \overline{n}') = (a + \overline{n}')' \quad \text{inductive hypothesis} \tag{32.5}$$
$$\mathbf{Q} \vdash (a' + \overline{n})' = (a + \overline{n}')' \quad \text{by eq. (32.4) and eq. (32.5).}$$

□

It is again worth mentioning that this is weaker than saying that **Q** proves $\forall x \, \forall y \, (x' + y) = (x + y)'$. Although this sentence is true in $\mathfrak{N}$, **Q** does not prove it.

**Lemma 32.22.** *1.* $\mathbf{Q} \vdash \forall x \, \neg x < 0$.

*2. For every natural number $n$,*

$$\mathbf{Q} \vdash \forall x \, (x < \overline{n+1} \to (x = 0 \vee \cdots \vee x = \overline{n})).$$

*Proof.* Let us do 1 and part of 2, informally (i.e., only giving hints as to how to construct the formal derivation).

For part 1, by the definition of $<$, we need to prove $\neg \exists y \, (y' + a) = 0$ in **Q**, which is equivalent (using the axioms and rules of first-order logic) to $\forall b \, (y' + a) \neq 0$. Here is the idea: suppose $(y' + b) = 0$. If $a = 0$, we have $(y' + 0) = 0$. But by axiom $Q_4$ of **Q**, we have $(b' + 0) = b'$, and by axiom $Q_2$ we have $b' \neq 0$, a contradiction. So $\forall y \, (y' + a) \neq 0$. If $a \neq 0$, by axiom $Q_3$, there is a $c$ such that $a = c'$. But then we have $(b' + c') = 0$. By axiom $Q_5$, we have $(b' + c)' = 0$, again contradicting axiom $Q_2$.

For part 2, use induction on $n$. Let us consider the base case, when $n = 0$. In that case, we need to show $a < \overline{1} \to a = 0$. Suppose $a < \overline{1}$. Then by the defining axiom for $<$, we have $\exists y \, (y' + a) = 0'$.

Suppose $b$ has that property, i.e., we have $b' + a = 0'$. We need to show $a = 0$. By axiom $Q_3$, if $a \neq 0$, we get $a = c'$ for some $z$. Then we have $(b' + c') = 0'$. By axiom $Q_5$ of **Q**, we have $(b' + c)' = 0'$. By axiom $Q_1$, we have $(b' + c) = 0$. But this means, by definition, $z < 0$, contradicting part 1. □

**Lemma 32.23.** *For every $m \in \mathbb{N}$,*

$$\mathbf{Q} \vdash \forall y \, ((y < \overline{m} \vee \overline{m} < y) \vee y = \overline{m}).$$

*Proof.* By induction on $m$. First, consider the case $m = 0$. $\mathbf{Q} \vdash \forall y \, (y \neq 0 \to \exists z \, y = z')$ by $Q_3$. But if $b = c'$, then $(c' + 0) = (b + 0)$ by the logic of $=$. By $Q_4$, $(b + 0) = b$, so we have $(c' + 0) = b$, and hence $\exists z \, (z' + 0) = b$. By the

definition of $<$ in $Q_8$, $0 < b$. If $0 < b$, then also $0 < b \vee b < 0$. We obtain: $b \neq 0 \to (0 < b \vee b < 0)$, which is equivalent to $(0 < b \vee b < 0) \vee b = 0$.

Now suppose we have

$$\mathbf{Q} \vdash \forall y\, ((y < \overline{m} \vee \overline{m} < y) \vee y = \overline{m})$$

and we want to show

$$\mathbf{Q} \vdash \forall y\, ((y < \overline{m+1} \vee \overline{m+1} < y) \vee y = \overline{m+1})$$

The first disjunct $b < \overline{m}$ is equivalent (by $Q_8$) to $\exists z\, (z' + b) = \overline{m}$. Suppose $c$ has this property. If $(c' + b) = \overline{m}$, then also $(c' + b)' = \overline{m}'$. By $Q_4$, $(c' + b)' = (c'' + b)$. Hence, $(c'' + b) = \overline{m}'$. We get $\exists u\, (u' + b) = \overline{m+1}$ by existentially generalizing on $c'$ and keeping in mind that $\overline{m}'$ is $\overline{m+1}$. Hence, if $b < \overline{m}$ then $b < \overline{m+1}$.

Now suppose $\overline{m} < b$, i.e., $\exists z\, (z' + \overline{m}) = b$. Suppose $c$ is such a $z$. By $Q_3$ and some logic, we have $c = 0 \vee \exists u\, c = u'$. If $c = 0$, we have $(0' + \overline{m}) = b$. Since $\mathbf{Q} \vdash (0' + \overline{m}) = \overline{m+1}$, we have $b = \overline{m+1}$. Now suppose $\exists u\, c = u'$. Let $d$ be such a $u$. Then:

$$\begin{aligned}
b &= (c' + \overline{m}) && \text{by assumption}\\
(c' + \overline{m}) &= (d'' + \overline{m}) && \text{from } c = d'\\
(d'' + \overline{m}) &= (d' + \overline{m})' && \text{by Lemma 32.21}\\
(d' + \overline{m})' &= (d' + \overline{m}') && \text{by } Q_5\text{, so}\\
b &= (d' + \overline{m+1})
\end{aligned}$$

By existential generalization, $\exists u\, (u' + \overline{m+1}) = b$, i.e., $\overline{m+1} < b$. So, if $\overline{m} < b$, then $\overline{m+1} < b \vee b = \overline{m+1}$.

Finally, assume $b = \overline{m}$. Then, since $\mathbf{Q} \vdash (0' + \overline{m}) = \overline{m+1}$, $(0' + b) = \overline{m+1}$. From this we get $\exists z\, (z' + b) = \overline{m+1}$, or $b < \overline{m+1}$.

Hence, from each disjunct of the case for $m$, we can obtain the case for $m + 1$. □

**Proposition 32.24.** *If $\varphi_g(x, z, y)$ represents $g(x, y)$ in $\mathbf{Q}$, then*

$$\varphi_f(z, y) \equiv \varphi_g(y, z, 0) \wedge \forall w\, (w < y \to \neg\varphi_g(w, z, 0)).$$

*represents $f(z) = \mu x\, [g(x, z) = 0]$.*

*Proof.* First we show that if $f(n) = m$, then $\mathbf{Q} \vdash \varphi_f(\overline{n}, \overline{m})$, i.e.,

$$\mathbf{Q} \vdash \varphi_g(\overline{m}, \overline{n}, 0) \wedge \forall w\, (w < \overline{m} \to \neg\varphi_g(w, \overline{n}, 0)).$$

Since $\varphi_g(x, z, y)$ represents $g(x, z)$ and $g(m, n) = 0$ if $f(n) = m$, we have

$$\mathbf{Q} \vdash \varphi_g(\overline{m}, \overline{n}, 0).$$

If $f(n) = m$, then for every $k < m$, $g(k, n) \neq 0$. So

$$\mathbf{Q} \vdash \neg\varphi_g(\overline{k}, \overline{n}, 0).$$

We get that

$$\mathbf{Q} \vdash \forall w\,(w < \overline{m} \to \neg\varphi_g(w, \overline{n}, 0)). \tag{32.6}$$

by Lemma 32.22 (by (1) in case $m = 0$ and by (2) otherwise).

Now let's show that if $f(n) = m$, then $\mathbf{Q} \vdash \forall y\,(\varphi_f(\overline{n}, y) \to y = \overline{m})$. We again sketch the argument informally, leaving the formalization to the reader.

Suppose $\varphi_f(\overline{n}, b)$. From this we get (a) $\varphi_g(b, \overline{n}, 0)$ and (b) $\forall w\,(w < b \to \neg\varphi_g(w, \overline{n}, 0))$. By Lemma 32.23, $(b < \overline{m} \vee \overline{m} < b) \vee b = \overline{m}$. We'll show that both $b < \overline{m}$ and $\overline{m} < b$ leads to a contradiction.

If $\overline{m} < b$, then $\neg\varphi_g(\overline{m}, \overline{n}, 0)$ from (b). But $m = f(n)$, so $g(m, n) = 0$, and so $\mathbf{Q} \vdash \varphi_g(\overline{m}, \overline{n}, 0)$ since $\varphi_g$ represents $g$. So we have a contradiction.

Now suppose $b < \overline{m}$. Then since $\mathbf{Q} \vdash \forall w\,(w < \overline{m} \to \neg\varphi_g(w, \overline{n}, 0))$ by eq. (32.6), we get $\neg\varphi_g(b, \overline{n}, 0)$. This again contradicts (a). □

## 32.8 Computable Functions are Representable in Q

**Theorem 32.25.** *Every computable function is representable in* **Q**.

*Proof.* For definiteness, and using the Church-Turing Thesis, let's say that a function is computable iff it is general recursive. The general recursive functions are those which can be defined from the zero function zero, the successor function succ, and the projection function $P^n_i$ using composition, primitive recursion, and regular minimization. By Lemma 32.8, any function $h$ that can be defined from $f$ and $g$ can also be defined using composition and regular minimization from $f$, $g$, and zero, succ, $P^n_i$, add, mult, $\chi_=$. Consequently, a function is general recursive iff it can be defined from zero, succ, $P^n_i$, add, mult, $\chi_=$ using composition and regular minimization.

We've furthermore shown that the basic functions in question are representable in **Q** (Propositions 32.9 to 32.12, 32.14 and 32.16), and that any function defined from representable functions by composition or regular minimization (Proposition 32.20, Proposition 32.24) is also representable. Thus every general recursive function is representable in **Q**. □

We have shown that the set of computable functions can be characterized as the set of functions representable in **Q**. In fact, the proof is more general. From the definition of representability, it is not hard to see that any theory extending **Q** (or in which one can interpret **Q**) can represent the computable functions. But, conversely, in any proof system in which the notion of proof is computable, every representable function is computable. So, for example, the

set of computable functions can be characterized as the set of functions representable in Peano arithmetic, or even Zermelo-Fraenkel set theory. As Gödel noted, this is somewhat surprising. We will see that when it comes to provability, questions are very sensitive to which theory you consider; roughly, the stronger the axioms, the more you can prove. But across a wide range of axiomatic theories, the representable functions are exactly the computable ones; stronger theories do not represent more functions as long as they are axiomatizable.

## 32.9 Representing Relations

Let us say what it means for a *relation* to be representable.

**Definition 32.26.** A relation $R(x_0, \ldots, x_k)$ on the natural numbers is *representable in* **Q** if there is a formula $\varphi_R(x_0, \ldots, x_k)$ such that whenever $R(n_0, \ldots, n_k)$ is true, **Q** proves $\varphi_R(\overline{n_0}, \ldots, \overline{n_k})$, and whenever $R(n_0, \ldots, n_k)$ is false, **Q** proves $\neg\varphi_R(\overline{n_0}, \ldots, \overline{n_k})$.

**Theorem 32.27.** *A relation is representable in* **Q** *if and only if it is computable.*

*Proof.* For the forwards direction, suppose $R(x_0, \ldots, x_k)$ is represented by the formula $\varphi_R(x_0, \ldots, x_k)$. Here is an algorithm for computing $R$: on input $n_0$, $\ldots$, $n_k$, simultaneously search for a proof of $\varphi_R(\overline{n_0}, \ldots, \overline{n_k})$ and a proof of $\neg\varphi_R(\overline{n_0}, \ldots, \overline{n_k})$. By our hypothesis, the search is bound to find one or the other; if it is the first, report "yes," and otherwise, report "no."

In the other direction, suppose $R(x_0, \ldots, x_k)$ is computable. By definition, this means that the function $\chi_R(x_0, \ldots, x_k)$ is computable. By Theorem 32.2, $\chi_R$ is represented by a formula, say $\varphi_{\chi_R}(x_0, \ldots, x_k, y)$. Let $\varphi_R(x_0, \ldots, x_k)$ be the formula $\varphi_{\chi_R}(x_0, \ldots, x_k, \overline{1})$. Then for any $n_0, \ldots, n_k$, if $R(n_0, \ldots, n_k)$ is true, then $\chi_R(n_0, \ldots, n_k) = 1$, in which case **Q** proves $\varphi_{\chi_R}(\overline{n_0}, \ldots, \overline{n_k}, \overline{1})$, and so **Q** proves $\varphi_R(\overline{n_0}, \ldots, \overline{n_k})$. On the other hand, if $R(n_0, \ldots, n_k)$ is false, then $\chi_R(n_0, \ldots, n_k) = 0$. This means that **Q** proves

$$\forall y\, (\varphi_{\chi_R}(\overline{n_0}, \ldots, \overline{n_k}, y) \to y = \overline{0}).$$

Since **Q** proves $\overline{0} \neq \overline{1}$, **Q** proves $\neg\varphi_{\chi_R}(\overline{n_0}, \ldots, \overline{n_k}, \overline{1})$, and so it proves $\neg\varphi_R(\overline{n_0}, \ldots, \overline{n_k})$. □

## 32.10 Undecidability

We call a theory **T** *undecidable* if there is no computational procedure which, after finitely many steps and unfailingly, provides a correct answer to the question "does **T** prove $\varphi$?" for any sentence $\varphi$ in the language of **T**. So **Q** would be decidable iff there were a computational procedure which decides, given a sentence $\varphi$ in the language of arithmetic, whether $\mathbf{Q} \vdash \varphi$ or not. We can make

this more precise by asking: Is the relation $\mathrm{Prov}_{\mathbf{Q}}(y)$, which holds of $y$ iff $y$ is the Gödel number of a sentence provable in $\mathbf{Q}$, recursive? The answer is: no.

**Theorem 32.28.** $\mathbf{Q}$ *is undecidable, i.e., the relation*

$$\mathrm{Prov}_{\mathbf{Q}}(y) \Leftrightarrow \mathrm{Sent}(y) \wedge \exists x\, \mathrm{Prf}_{\mathbf{Q}}(x, y)$$

*is not recursive.*

*Proof.* Suppose it were. Then we could solve the halting problem as follows: Given $e$ and $n$, we know that $\varphi_e(n) \downarrow$ iff there is an $s$ such that $T(e, n, s)$, where $T$ is Kleene's predicate from Theorem 25.28. Since $T$ is primitive recursive it is representable in $\mathbf{Q}$ by a formula $\psi_T$, that is, $\mathbf{Q} \vdash \psi_T(\overline{e}, \overline{n}, \overline{s})$ iff $T(e, n, s)$. If $\mathbf{Q} \vdash \psi_T(\overline{e}, \overline{n}, \overline{s})$ then also $\mathbf{Q} \vdash \exists y\, \psi_T(\overline{e}, \overline{n}, y)$. If no such $s$ exists, then $\mathbf{Q} \vdash \neg\psi_T(\overline{e}, \overline{n}, \overline{s})$ for every $s$. But $\mathbf{Q}$ is $\omega$-consistent, i.e., if $\mathbf{Q} \vdash \neg\varphi(\overline{n})$ for every $n \in \mathbb{N}$, then $\mathbf{Q} \nvdash \exists y\, \varphi(y)$. We know this because the axioms of $\mathbf{Q}$ are true in the standard model $\mathfrak{N}$. So, $\mathbf{Q} \nvdash \exists y\, \psi_T(\overline{e}, \overline{n}, y)$. In other words, $\mathbf{Q} \vdash \exists y\, \psi_T(\overline{e}, \overline{n}, y)$ iff there is an $s$ such that $T(e, n, s)$, i.e., iff $\varphi_e(n) \downarrow$. From $e$ and $n$ we can compute ${}^{\#}\exists y\, \psi_T(\overline{e}, \overline{n}, y)^{\#}$, let $g(e, n)$ be the primitive recursive function which does that. So

$$h(e, n) = \begin{cases} 1 & \text{if } \mathrm{Prov}_{\mathbf{Q}}(g(e, n)) \\ 0 & \text{otherwise.} \end{cases}$$

This would show that $h$ is recursive if $\mathrm{Prov}_{\mathbf{Q}}$ is. But $h$ is not recursive, by Theorem 25.29, so $\mathrm{Prov}_{\mathbf{Q}}$ cannot be either. □

**Corollary 32.29.** *First-order logic is undecidable.*

*Proof.* If first-order logic were decidable, provability in $\mathbf{Q}$ would be as well, since $\mathbf{Q} \vdash \varphi$ iff $\vdash \omega \to \varphi$, where $\omega$ is the conjunction of the axioms of $\mathbf{Q}$. □

## Problems

**Problem 32.1.** Prove that $y = \mathrm{o}$, $y = x'$, and $y = x_i$ represent zero, succ, and $P^n_i$, respectively.

**Problem 32.2.** Prove Lemma 32.17.

**Problem 32.3.** Use Lemma 32.17 to prove Proposition 32.16.

**Problem 32.4.** Using the proofs of Proposition 32.19 and Proposition 32.19 as a guide, carry out the proof of Proposition 32.20 in detail.

**Problem 32.5.** Show that if $R$ is representable in $\mathbf{Q}$, so is $\chi_R$.

This chapter depends on material in the chapter on computability theory, but can be left out if that hasn't been covered. It's currently a basic conversion of Jeremy Avigad's notes, has not been revised, and is missing exercises.

# Chapter 33

# Theories and Computability

## 33.1 Introduction

> This section should be rewritten.

We have the following:

1. A definition of what it means for a function to be representable in **Q** (Definition 32.1)
2. a definition of what it means for a relation to be representable in **Q** (Definition 32.26)
3. a theorem asserting that the representable functions of **Q** are exactly the computable ones (Theorem 32.2)
4. a theorem asserting that the representable relations of **Q** are exactly the computable ones Theorem 32.27)

A *theory* is a set of sentences that is deductively closed, that is, with the property that whenever $T$ proves $\varphi$ then $\varphi$ is in $T$. It is probably best to think of a theory as being a collection of sentences, together with all the things that these sentences imply. From now on, I will use **Q** to refer to the *theory* consisting of the set of sentences derivable from the eight axioms in section 32.1. Remember that we can code formula of **Q** as numbers; if $\varphi$ is such a formula, let ${}^{\#}\varphi^{\#}$ denote the number coding $\varphi$. Modulo this coding, we can now ask whether various sets of formulas are computable or not.

## 33.2 Q is C.e.-Complete

**Theorem 33.1.** **Q** *is c.e. but not decidable. In fact, it is a complete c.e. set.*

*Proof.* It is not hard to see that **Q** is c.e., since it is the set of (codes for) sentences $y$ such that there is a proof $x$ of $y$ in **Q**:

$$Q = \{y : \exists x \, \mathrm{Prf}_{\mathbf{Q}}(x, y)\}.$$

But we know that $\mathrm{Prf}_{\mathbf{Q}}(x, y)$ is computable (in fact, primitive recursive), and any set that can be written in the above form is c.e.

Saying that it is a complete c.e. set is equivalent to saying that $K \leq_m Q$, where $K = \{x : \varphi_x(x) \downarrow\}$. So let us show that $K$ is reducible to **Q**. Since Kleene's predicate $T(e, x, s)$ is primitive recursive, it is representable in **Q**, say, by $\varphi_T$. Then for every $x$, we have

$$\begin{aligned} x \in K &\to \exists s \, T(x, x, s) \\ &\to \exists s \, (\mathbf{Q} \vdash \varphi_T(\overline{x}, \overline{x}, \overline{s})) \\ &\to \mathbf{Q} \vdash \exists s \, \varphi_T(\overline{x}, \overline{x}, s). \end{aligned}$$

Conversely, if $\mathbf{Q} \vdash \exists s \, \varphi_T(\overline{x}, \overline{x}, s)$, then, in fact, for some natural number $n$ the formula $\varphi_T(\overline{x}, \overline{x}, \overline{n})$ must be true. Now, if $T(x, x, n)$ were false, **Q** would prove $\neg\varphi_T(\overline{x}, \overline{x}, \overline{n})$, since $\varphi_T$ represents $T$. But then **Q** proves a false formula, which is a contradiction. So $T(x, x, n)$ must be true, which implies $\varphi_x(x) \downarrow$.

In short, we have that for every $x$, $x$ is in $K$ if and only if **Q** proves $\exists s \, T(\overline{x}, \overline{x}, s)$. So the function $f$ which takes $x$ to (a code for) the sentence $\exists s \, T(\overline{x}, \overline{x}, s)$ is a reduction of $K$ to **Q**. □

## 33.3 $\omega$-Consistent Extensions of Q are Undecidable

The proof that **Q** is c.e.-complete relied on the fact that any sentence provable in **Q** is "true" of the natural numbers. The next definition and theorem strengthen this theorem, by pinpointing just those aspects of "truth" that were needed in the proof above. Don't dwell on this theorem too long, though, because we will soon strengthen it even further. We include it mainly for historical purposes: Gödel's original paper used the notion of $\omega$-consistency, but his result was strengthened by replacing $\omega$-consistency with ordinary consistency soon after.

**Definition 33.2.** A theory **T** is $\omega$-consistent if the following holds: if $\exists x \, \varphi(x)$ is any sentence and **T** proves $\neg\varphi(\overline{0})$, $\neg\varphi(\overline{1})$, $\neg\varphi(\overline{2})$, ... then **T** does not prove $\exists x \, \varphi(x)$.

**Theorem 33.3.** *Let* **T** *be any $\omega$-consistent theory that includes* **Q**. *Then* **T** *is not decidable.*

*Proof.* If **T** includes **Q**, then **T** represents the computable functions and relations. We need only modify the previous proof. As above, if $x \in K$, then **T** proves $\exists s \, \varphi_T(\overline{x}, \overline{x}, s)$. Conversely, suppose **T** proves $\exists s \, \varphi_T(\overline{x}, \overline{x}, s)$. Then $x$

must be in $K$: otherwise, there is no halting computation of machine $x$ on input $x$; since $\varphi_T$ represents Kleene's $T$ relation, **T** proves $\neg\varphi_T(\overline{x}, \overline{x}, \overline{0})$, $\neg\varphi_T(\overline{x}, \overline{x}, \overline{1})$, ..., making **T** $\omega$-inconsistent. □

## 33.4 Consistent Extensions of Q are Undecidable

Remember that a theory is *consistent* if it does not prove both $\varphi$ and $\neg\varphi$ for any formula $\varphi$. Since anything follows from a contradiction, an inconsistent theory is trivial: every sentence is provable. Clearly, if a theory if $\omega$-consistent, then it is consistent. But being consistent is a weaker requirement (i.e., there are theories that are consistent but not $\omega$-consistent.). We can weaken the assumption in Definition 33.2 to simple consistency to obtain a stronger theorem.

**Lemma 33.4.** *There is no "universal computable relation." That is, there is no binary computable relation $R(x, y)$, with the following property: whenever $S(y)$ is a unary computable relation, there is some $k$ such that for every $y$, $S(y)$ is true if and only if $R(k, y)$ is true.*

*Proof.* Suppose $R(x, y)$ is a universal computable relation. Let $S(y)$ be the relation $\neg R(y, y)$. Since $S(y)$ is computable, for some $k$, $S(y)$ is equivalent to $R(k, y)$. But then we have that $S(k)$ is equivalent to both $R(k, k)$ and $\neg R(k, k)$, which is a contradiction. □

**Theorem 33.5.** *Let* **T** *be any consistent theory that includes* **Q**. *Then* **T** *is not decidable.*

*Proof.* Suppose **T** is a consistent, decidable extension of **Q**. We will obtain a contradiction by using **T** to define a universal computable relation.

Let $R(x, y)$ hold if and only if

> $x$ codes a formula $\theta(u)$, and **T** proves $\theta(\overline{y})$.

Since we are assuming that **T** is decidable, $R$ is computable. Let us show that $R$ is universal. If $S(y)$ is any computable relation, then it is representable in **Q** (and hence **T**) by a formula $\theta_S(u)$. Then for every $n$, we have

$$\begin{aligned} S(\overline{n}) &\rightarrow T \vdash \theta_S(\overline{n}) \\ &\rightarrow R(^{\#}\theta_S(u)^{\#}, n) \end{aligned}$$

and

$$\begin{aligned} \neg S(\overline{n}) &\rightarrow T \vdash \neg\theta_S(\overline{n}) \\ &\rightarrow T \nvdash \theta_S(\overline{n}) \quad \text{(since } \mathbf{T} \text{ is consistent)} \\ &\rightarrow \neg R(^{\#}\theta_S(u)^{\#}, n). \end{aligned}$$

That is, for every $y$, $S(y)$ is true if and only if $R(^{\#}\theta_S(u)^{\#}, y)$ is. So $R$ is universal, and we have the contradiction we were looking for. □

Let "true arithmetic" be the theory $\{\varphi : \mathbb{N} \vDash \varphi\}$, that is, the set of sentences in the language of arithmetic that are true in the standard interpretation.

**Corollary 33.6.** *True arithmetic is not decidable.*

## 33.5 Axiomatizable Theories

A theory **T** is said to be *axiomatizable* if it has a computable set of axioms $A$. (Saying that $A$ is a set of axioms for **T** means $T = \{\varphi : A \vdash \varphi\}$.) Any "reasonable" axiomatization of the natural numbers will have this property. In particular, any theory with a finite set of axioms is axiomatizable.

**Lemma 33.7.** *Suppose* **T** *is axiomatizable. Then* **T** *is computably enumerable.*

*Proof.* Suppose $A$ is a computable set of axioms for **T**. To determine if $\varphi \in T$, just search for a proof of $\varphi$ from the axioms.

Put slightly differently, $\varphi$ is in **T** if and only if there is a finite list of axioms $\psi_1, \ldots, \psi_k$ in $A$ and a proof of $(\psi_1 \wedge \cdots \wedge \psi_k) \to \varphi$ in first-order logic. But we already know that any set with a definition of the form "there exists ... such that ..." is c.e., provided the second "..." is computable. □

## 33.6 Axiomatizable Complete Theories are Decidable

A theory is said to be *complete* if for every sentence $\varphi$, either $\varphi$ or $\neg\varphi$ is provable.

**Lemma 33.8.** *Suppose a theory* **T** *is complete and axiomatizable. Then* **T** *is decidable.*

*Proof.* Suppose **T** is complete and $A$ is a computable set of axioms. If **T** is inconsistent, it is clearly computable. (Algorithm: "just say yes.") So we can assume that **T** is also consistent.

To decide whether or not a sentence $\varphi$ is in **T**, simultaneously search for a proof of $\varphi$ from $A$ and a proof of $\neg\varphi$. Since **T** is complete, you are bound to find one or another; and since **T** is consistent, if you find a proof of $\neg\varphi$, there is no proof of $\varphi$.

Put in different terms, we already know that **T** is c.e.; so by a theorem we proved before, it suffices to show that the complement of **T** is c.e. also. But a formula $\varphi$ is in $\bar{\mathbf{T}}$ if and only if $\neg\varphi$ is in **T**; so $\bar{\mathbf{T}} \leq_m \mathbf{T}$. □

## 33.7 Q has no Complete, Consistent, Axiomatizable Extensions

**Theorem 33.9.** *There is no complete, consistent, axiomatizable extension of* **Q**.

*Proof.* We already know that there is no consistent, decidable extension of **Q**. But if **T** is complete and axiomatized, then it is decidable. □

This theorems is not that far from Gödel's original 1931 formulation of the First Incompleteness Theorem. Aside from the more modern terminology, the key differences are this: Gödel has "$\omega$-consistent" instead of "consistent"; and he could not say "axiomatizable" in full generality, since the formal notion of computability was not in place yet. (The formal models of computability were developed over the following decade, including by Gödel, and in large part to be able to characterize the kinds of theories that are susceptible to the Gödel phenomenon.)

The theorem says you can't have it all, namely, completeness, consistency, and axiomatizability. If you give up any one of these, though, you can have the other two: **Q** is consistent and computably axiomatized, but not complete; the inconsistent theory is complete, and computably axiomatized (say, by $\{0 \neq 0\}$), but not consistent; and the set of true sentence of arithmetic is complete and consistent, but it is not computably axiomatized.

## 33.8 Sentences Provable and Refutable in Q are Computably Inseparable

Let $\bar{\mathbf{Q}}$ be the set of sentences whose *negations* are provable in **Q**, i.e., $\bar{\mathbf{Q}} = \{\varphi : \mathbf{Q} \vdash \neg\varphi\}$. Remember that disjoint sets $A$ and $B$ are said to be computably inseparable if there is no computable set $C$ such that $A \subseteq C$ and $B \subseteq \overline{C}$.

**Lemma 33.10.** **Q** *and* $\bar{\mathbf{Q}}$ *are computably inseparable.*

*Proof.* Suppose $C$ is a computable set such that $\mathbf{Q} \subseteq C$ and $\bar{\mathbf{Q}} \subseteq \overline{C}$. Let $R(x, y)$ be the relation

> $x$ codes a formula $\theta(u)$ and $\theta(\overline{y})$ is in $C$.

We will show that $R(x, y)$ is a universal computable relation, yielding a contradiction.

Suppose $S(y)$ is computable, represented by $\theta_S(u)$ in **Q**. Then

$$\begin{aligned} S(\overline{n}) &\rightarrow \mathbf{Q} \vdash \theta_S(\overline{n}) \\ &\rightarrow \theta_S(\overline{n}) \in C \end{aligned}$$

and

$$\begin{aligned} \neg S(\overline{n}) &\rightarrow \mathbf{Q} \vdash \neg\theta_S(\overline{n}) \\ &\rightarrow \theta_S(\overline{n}) \in \bar{\mathbf{Q}} \\ &\rightarrow \theta_S(\overline{n}) \notin C \end{aligned}$$

So $S(y)$ is equivalent to $R(\#(\theta_S(\overline{u})), y)$. □

## 33.9 Theories Consistent with Q are Undecidable

The following theorem says that not only is **Q** undecidable, but, in fact, any theory that does not disagree with **Q** is undecidable.

**Theorem 33.11.** *Let* **T** *be any theory in the language of arithmetic that is consistent with* **Q** *(i.e.,* $\mathbf{T} \cup \mathbf{Q}$ *is consistent). Then* **T** *is undecidable.*

*Proof.* Remember that **Q** has a finite set of axioms, $Q_1, \ldots, Q_8$. We can even replace these by a single axiom, $\alpha = Q_1 \wedge \cdots \wedge Q_8$.

Suppose **T** is a decidable theory consistent with **Q**. Let

$$C = \{\varphi : \mathbf{T} \vdash \alpha \to \varphi\}.$$

We show that $C$ would be a computable separation of **Q** and $\bar{\mathbf{Q}}$, a contradiction. First, if $\varphi$ is in **Q**, then $\varphi$ is provable from the axioms of **Q**; by the deduction theorem, there is a proof of $\alpha \to \varphi$ in first-order logic. So $\varphi$ is in $C$.

On the other hand, if $\varphi$ is in $\bar{\mathbf{Q}}$, then there is a proof of $\alpha \to \neg\varphi$ in first-order logic. If **T** also proves $\alpha \to \varphi$, then **T** proves $\neg\alpha$, in which case $\mathbf{T} \cup \mathbf{Q}$ is inconsistent. But we are assuming $\mathbf{T} \cup \mathbf{Q}$ is consistent, so **T** does not prove $\alpha \to \varphi$, and so $\varphi$ is not in $C$.

We've shown that if $\varphi$ is in **Q**, then it is in $C$, and if $\varphi$ is in $\bar{\mathbf{Q}}$, then it is in $\overline{C}$. So $C$ is a computable separation, which is the contradiction we were looking for. □

This theorem is very powerful. For example, it implies:

**Corollary 33.12.** *First-order logic for the language of arithmetic (that is, the set* $\{\varphi : \varphi$ *is provable in first-order logic*$\}$*) is undecidable.*

*Proof.* First-order logic is the set of consequences of $\emptyset$, which is consistent with **Q**. □

## 33.10 Theories in which Q is Intepretable are Undecidable

We can strengthen these results even more. Informally, an interpretation of a language $\mathcal{L}_1$ in another language $\mathcal{L}_2$ involves defining the universe, relation symbols, and function symbols of $\mathcal{L}_1$ with formulas in $\mathcal{L}_2$. Though we won't take the time to do this, one can make this definition precise.

**Theorem 33.13.** *Suppose* **T** *is a theory in a language in which one can interpret the language of arithmetic, in such a way that* **T** *is consistent with the interpretation of* **Q**. *Then* **T** *is undecidable. If* **T** *proves the interpretation of the axioms of* **Q**, *then no consistent extension of* **T** *is decidable.*

The proof is just a small modification of the proof of the last theorem; one could use a counterexample to get a separation of **Q** and $\bar{\mathbf{Q}}$. One can take **ZFC**, Zermelo-Fraenkel set theory with the axiom of choice, to be an axiomatic foundation that is powerful enough to carry out a good deal of ordinary mathematics. In **ZFC** one can define the natural numbers, and via this interpretation, the axioms of **Q** are true. So we have

**Corollary 33.14.** *There is no decidable extension of* **ZFC**.

**Corollary 33.15.** *There is no complete, consistent, computably axiomatizable extension of* **ZFC**.

The language of **ZFC** has only a single binary relation, $\in$. (In fact, you don't even need equality.) So we have

**Corollary 33.16.** *First-order logic for any language with a binary relation symbol is undecidable.*

This result extends to any language with two unary function symbols, since one can use these to simulate a binary relation symbol. The results just cited are tight: it turns out that first-order logic for a language with only *unary* relation symbols and at most one *unary* function symbol is decidable.

One more bit of trivia. We know that the set of sentences in the language $\mathsf{0}, ', +, \times, <$ true in the standard model is undecidable. In fact, one can define $<$ in terms of the other symbols, and then one can define $+$ in terms of $\times$ and $'$. So the set of true sentences in the language $\mathsf{0}, ', \times$ is undecidable. On the other hand, Presburger has shown that the set of sentences in the language $\mathsf{0}, ', +$ true in the language of arithmetic is decidable. The procedure is computationally infeasible, however.

# Chapter 34

# Incompleteness and Provability

## 34.1 Introduction

Hilbert thought that a system of axioms for a mathematical structure, such as the natural numbers, is inadequate unless it allows one to derive all true statements about the structure. Combined with his later interest in formal systems of deduction, this suggests that he thought that we should guarantee that, say, the formal systems we are using to reason about the natural numbers is not only consistent, but also *complete*, i.e., every statement in its language is either derivable or its negation is. Gödel's first incompleteness theorem shows that no such system of axioms exists: there is no complete, consistent, axiomatizable formal system for arithmetic. In fact, no "sufficiently strong," consistent, axiomatizable mathematical theory is complete.

A more important goal of Hilbert's, the centerpiece of his program for the justification of modern ("classical") mathematics, was to find finitary consistency proofs for formal systems representing classical reasoning. With regard to Hilbert's program, then, Gödel's second incompleteness theorem was a much bigger blow. The second incompleteness theorem can be stated in vague terms, like the first incompleteness theorem. Roughly speaking, it says that no sufficiently strong theory of arithmetic can prove its own consistency. We will have to take "sufficiently strong" to include a little bit more than **Q**.

The idea behind Gödel's original proof of the incompleteness theorem can be found in the Epimenides paradox. Epimenides, a Cretan, asserted that all Cretans are liars; a more direct form of the paradox is the assertion "this sentence is false." Essentially, by replacing truth with derivability, Gödel was able to formalize a sentence which, in a roundabout way, asserts that it itself is not derivable. If that sentence were derivable, the theory would then be inconsistent. Gödel showed that the negation of that sentence is also not derivable from the system of axioms he was considering. (For this second part, Gödel had to assume that the theory **T** is what's called "$\omega$-consistent." $\omega$-Consistency is related to consistency, but is a stronger property. A few years

after Gödel, Rosser showed that assuming simple consistency of **T** is enough.)

The first challenge is to understand how one can construct a sentence that refers to itself. For every formula $\varphi$ in the language of **Q**, let $\ulcorner\varphi\urcorner$ denote the numeral corresponding to ${}^{\#}\varphi^{\#}$. Think about what this means: $\varphi$ is a formula in the language of **Q**, ${}^{\#}\varphi^{\#}$ is a natural number, and $\ulcorner\varphi\urcorner$ is a *term* in the language of **Q**. So every formula $\varphi$ in the language of **Q** has a *name*, $\ulcorner\varphi\urcorner$, which is a term in the language of **Q**; this provides us with a conceptual framework in which formulas in the language of **Q** can "say" things about other formulas. The following lemma is known as the fixed-point lemma.

**Lemma 34.1.** *Let* **T** *be any theory extending* **Q**, *and let* $\psi(x)$ *be any formula with only the variable* $x$ *free. Then there is a sentence* $\varphi$ *such that* $\mathbf{T} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$.

The lemma asserts that given any property $\psi(x)$, there is a sentence $\varphi$ that asserts "$\psi(x)$ is true of me," and **T** "knows" this.

How can we construct such a sentence? Consider the following version of the Epimenides paradox, due to Quine:

> "Yields falsehood when preceded by its quotation" yields falsehood when preceded by its quotation.

This sentence is not directly self-referential. It simply makes an assertion about the syntactic objects between quotes, and, in doing so, it is on par with sentences like

1. "Robert" is a nice name.
2. "I ran." is a short sentence.
3. "Has three words" has three words.

But what happens when one takes the phrase "yields falsehood when preceded by its quotation," and precedes it with a quoted version of itself? Then one has the original sentence! In short, the sentence asserts that it is false.

## 34.2 The Fixed-Point Lemma

The fixed-point lemma says that for any formula $\psi(x)$, there is a sentence $\varphi$ such that $\mathbf{T} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$, provided **T** extends **Q**. In the case of the liar sentence, we'd want $\varphi$ to be equivalent (provably in **T**) to "$\ulcorner\varphi\urcorner$ is false," i.e., the statement that ${}^{\#}\varphi^{\#}$ is the Gödel number of a false sentence. To understand the idea of the proof, it will be useful to compare it with Quine's informal gloss of $\varphi$ as, "'yields a falsehood when preceded by its own quotation' yields a falsehood when preceded by its own quotation." The operation of taking an expression, and then forming a sentence by preceding this expression by its own quotation may be called *diagonalizing* the expression, and the result its

diagonalization. So, the diagonalization of 'yields a falsehood when preceded by its own quotation' is "'yields a falsehood when preceded by its own quotation' yields a falsehood when preceded by its own quotation." Now note that Quine's liar sentence is not the diagonalization of 'yields a falsehood' but of 'yields a falsehood when preceded by its own quotation.' So the property being diagonalized to yield the liar sentence itself involves diagonalization!

In the language of arithmetic, we form quotations of a formula with one free variable by computing its Gödel numbers and then substituting the standard numeral for that Gödel number into the free variable. The diagonalization of $\alpha(x)$ is $\alpha(\overline{n})$, where $n = {}^{\#}\alpha(x)^{\#}$. (From now on, let's abbreviate $\overline{{}^{\#}\alpha(x)^{\#}}$ as $\ulcorner\alpha(x)\urcorner$.) So if $\psi(x)$ is "is a falsehood," then "yields a falsehood if preceded by its own quotation," would be "yields a falsehood when applied to the Gödel number of its diagonalization." If we had a symbol $diag$ for the function $\mathrm{diag}(n)$ which computes the Gödel number of the diagonalization of the formula with Gödel number $n$, we could write $\alpha(x)$ as $\psi(diag(x))$. And Quine's version of the liar sentence would then be the diagonalization of it, i.e., $\alpha(\ulcorner\alpha\urcorner)$ or $\psi(diag(\ulcorner\psi(diag(x))\urcorner))$. Of course, $\psi(x)$ could now be any other property, and the same construction would work. For the incompleteness theorem, we'll take $\psi(x)$ to be "$x$ is not derivable in $\mathbf{T}$." Then $\alpha(x)$ would be "yields a sentence not derivable in $\mathbf{T}$ when applied to the Gödel number of its diagonalization."

To formalize this in $\mathbf{T}$, we have to find a way to formalize diag. The function $\mathrm{diag}(n)$ is computable, in fact, it is primitive recursive: if $n$ is the Gödel number of a formula $\alpha(x)$, $\mathrm{diag}(n)$ returns the Gödel number of $\alpha(\ulcorner\alpha(x)\urcorner)$. (Recall, $\ulcorner\alpha(x)\urcorner$ is the standard numeral of the Gödel number of $\alpha(x)$, i.e., $\overline{{}^{\#}\alpha(x)^{\#}}$). If $diag$ were a function symbol in $\mathbf{T}$ representing the function diag, we could take $\varphi$ to be the formula $\psi(diag(\ulcorner\psi(diag(x))\urcorner))$. Notice that

$$\begin{aligned}\mathrm{diag}({}^{\#}\psi(diag(x))^{\#}) &= {}^{\#}\psi(diag(\ulcorner\psi(diag(x))\urcorner)^{\#}\\ &= {}^{\#}\varphi^{\#}.\end{aligned}$$

Assuming $\mathbf{T}$ can derive

$$diag(\ulcorner\psi(diag(x))\urcorner) = \ulcorner\varphi\urcorner,$$

it can derive $\psi(diag(\ulcorner\psi(diag(x))\urcorner)) \leftrightarrow \psi(\ulcorner\varphi\urcorner)$. But the left hand side is, by definition, $\varphi$.

Of course, $diag$ will in general not be a function symbol of $\mathbf{T}$, and certainly is not one of $\mathbf{Q}$. But, since diag is computable, it is *representable* in $\mathbf{Q}$ by some formula $\theta_{\mathrm{diag}}(x, y)$. So instead of writing $\psi(diag(x))$ we can write $\exists y\,(\theta_{\mathrm{diag}}(x, y) \wedge \psi(y))$. Otherwise, the proof sketched above goes through, and in fact, it goes through already in $\mathbf{Q}$.

**Lemma 34.2.** *Let $\psi(x)$ be any formula with one free variable $x$. Then there is a sentence $\varphi$ such that $\mathbf{Q} \vdash \varphi \leftrightarrow \psi(\ulcorner\varphi\urcorner)$.*

*Proof.* Given $\psi(x)$, let $\alpha(x)$ be the formula $\exists y\,(\theta_{\text{diag}}(x,y) \wedge \psi(y))$ and let $\varphi$ be its diagonalization, i.e., the formula $\alpha(\ulcorner \alpha(x) \urcorner)$.

Since $\theta_{\text{diag}}$ represents diag, and $\text{diag}(\ulcorner \alpha(x) \urcorner) = \ulcorner \varphi \urcorner$, **Q** can derive

$$\theta_{\text{diag}}(\ulcorner \alpha(x) \urcorner, \ulcorner \varphi \urcorner) \tag{34.1}$$

$$\forall y\,(\theta_{\text{diag}}(\ulcorner \alpha(x) \urcorner, y) \to y = \ulcorner \varphi \urcorner). \tag{34.2}$$

Now we show that $\mathbf{Q} \vdash \varphi \leftrightarrow \psi(\ulcorner \varphi \urcorner)$. We argue informally, using just logic and facts derivable in **Q**.

First, suppose $\varphi$, i.e., $\alpha(\ulcorner \alpha(x) \urcorner)$. Going back to the definition of $\alpha(x)$, we see that $\alpha(\ulcorner \alpha(x) \urcorner)$ just is

$$\exists y\,(\theta_{\text{diag}}(\ulcorner \alpha(x) \urcorner, y) \wedge \psi(y)).$$

Consider such a $y$. Since $\theta_{\text{diag}}(\ulcorner \alpha(x) \urcorner, y)$, by eq. (34.2), $y = \ulcorner \varphi \urcorner$. So, from $\psi(y)$ we have $\psi(\ulcorner \varphi \urcorner)$.

Now suppose $\psi(\ulcorner \varphi \urcorner)$. By eq. (34.1), we have $\theta_{\text{diag}}(\ulcorner \alpha(x) \urcorner, \ulcorner \varphi \urcorner) \wedge \psi(\ulcorner \varphi \urcorner)$. It follows that $\exists y\,(\theta_{\text{diag}}(\ulcorner \alpha(x) \urcorner, y) \wedge \psi(y))$. But that's just $\alpha(\ulcorner \alpha \urcorner)$, i.e., $\varphi$. □

You should compare this to the proof of the fixed-point lemma in computability theory. The difference is that here we want to define a *statement* in terms of itself, whereas there we wanted to define a *function* in terms of itself; this difference aside, it is really the same idea.

## 34.3 The First Incompleteness Theorem

We can now describe Gödel's original proof of the first incompleteness theorem. Let **T** be any computably axiomatized theory in a language extending the language of arithmetic, such that **T** includes the axioms of **Q**. This means that, in particular, **T** represents computable functions and relations.

We have argued that, given a reasonable coding of formulas and proofs as numbers, the relation $\text{Prf}_T(x,y)$ is computable, where $\text{Prf}_T(x,y)$ holds if and only if $x$ is the Gödel number of a derivation of the formula with Gödel number $y$ in **T**. In fact, for the particular theory that Gödel had in mind, Gödel was able to show that this relation is primitive recursive, using the list of 45 functions and relations in his paper. The 45th relation, $xBy$, is just $\text{Prf}_T(x,y)$ for his particular choice of **T**. Remember that where Gödel uses the word "recursive" in his paper, we would now use the phrase "primitive recursive."

Since $\text{Prf}_T(x,y)$ is computable, it is representable in **T**. We will use $\mathsf{Prf}_T(x,y)$ to refer to the formula that represents it. Let $\mathsf{Prov}_T(y)$ be the formula $\exists x\, \mathsf{Prf}_T(x,y)$. This describes the 46th relation, $\text{Bew}(y)$, on Gödel's list. As Gödel notes, this is the only relation that "cannot be asserted to be recursive." What he probably meant is this: from the definition, it is not clear that it is computable; and later developments, in fact, show that it isn't.

Let $\mathbf{T}$ be an axiomatizable theory containing $\mathbf{Q}$. Then $\mathrm{Prf}_T(x, y)$ is decidable, hence representable in $\mathbf{Q}$ by a formula $\mathsf{Prf}_T(x, y)$. Let $\mathsf{Prov}_T(y)$ be the formula we described above. By the fixed-point lemma, there is a formula $\gamma_{\mathbf{T}}$ such that $\mathbf{Q}$ (and hence $\mathbf{T}$) derives

$$\gamma_{\mathbf{T}} \leftrightarrow \neg \mathsf{Prov}_T(\ulcorner \gamma_{\mathbf{T}} \urcorner). \tag{34.3}$$

Note that $\gamma_{\mathbf{T}}$ says, in essence, "$\gamma_{\mathbf{T}}$ is not derivable in $\mathbf{T}$."

**Lemma 34.3.** *If $\mathbf{T}$ is a consistent, axiomatizable theory extending $\mathbf{Q}$, then $\mathbf{T} \nvdash \gamma_{\mathbf{T}}$.*

*Proof.* Suppose $\mathbf{T}$ derives $\gamma_{\mathbf{T}}$. Then there *is* a derivation, and so, for some number $m$, the relation $\mathrm{Prf}_T(m, {}^{\#}\gamma_{\mathbf{T}}{}^{\#})$ holds. But then $\mathbf{Q}$ derives the sentence $\mathsf{Prf}_T(\overline{m}, \ulcorner \gamma_{\mathbf{T}} \urcorner)$. So $\mathbf{Q}$ derives $\exists x\, \mathsf{Prf}_T(x, \ulcorner \gamma_{\mathbf{T}} \urcorner)$, which is, by definition, $\mathsf{Prov}_T(\ulcorner \gamma_{\mathbf{T}} \urcorner)$. By eq. (34.3), $\mathbf{Q}$ derives $\neg \gamma_{\mathbf{T}}$, and since $\mathbf{T}$ extends $\mathbf{Q}$, so does $\mathbf{T}$. We have shown that if $\mathbf{T}$ derives $\gamma_{\mathbf{T}}$, then it also derives $\neg \gamma_{\mathbf{T}}$, and hence it would be inconsistent. □

**Definition 34.4.** A theory $\mathbf{T}$ is *$\omega$-consistent* if the following holds: if $\exists x\, \varphi(x)$ is any sentence and $\mathbf{T}$ derives $\neg \varphi(\overline{0})$, $\neg \varphi(\overline{1})$, $\neg \varphi(\overline{2})$, ... then $\mathbf{T}$ does not prove $\exists x\, \varphi(x)$.

Note that every $\omega$-consistent theory is also consistent. This follows simply from the fact that if $\mathbf{T}$ is inconsistent, then $\mathbf{T} \vdash \varphi$ for every $\varphi$. In particular, if $\mathbf{T}$ is inconsistent, it derives both $\neg \varphi(\overline{n})$ for every $n$ and also derives $\exists x\, \varphi(x)$. So, if $\mathbf{T}$ is inconsistent, it is $\omega$-inconsistent. By contraposition, if $\mathbf{T}$ is $\omega$-consistent, it must be consistent.

**Lemma 34.5.** *If $\mathbf{T}$ is an $\omega$-consistent, axiomatizable theory extending $\mathbf{Q}$, then $\mathbf{T} \nvdash \gamma_{\mathbf{T}}$.*

*Proof.* We show that if $\mathbf{T}$ derives $\neg \gamma_{\mathbf{T}}$, then it is $\omega$-inconsistent. Suppose $\mathbf{T}$ derives $\neg \gamma_{\mathbf{T}}$. If $\mathbf{T}$ is inconsistent, it is $\omega$-inconsistent, and we are done. Otherwise, $\mathbf{T}$ is consistent, so it does not derive $\gamma_{\mathbf{T}}$ by Lemma 34.3. Since there is no derivation of $\gamma_{\mathbf{T}}$ in $\mathbf{T}$, $\mathbf{Q}$ derives

$$\neg \mathsf{Prf}_T(\overline{0}, \ulcorner \gamma_{\mathbf{T}} \urcorner), \neg \mathsf{Prf}_T(\overline{1}, \ulcorner \gamma_{\mathbf{T}} \urcorner), \neg \mathsf{Prf}_T(\overline{2}, \ulcorner \gamma_{\mathbf{T}} \urcorner), \ldots$$

and so does $\mathbf{T}$. On the other hand, by eq. (34.3), $\neg \gamma_{\mathbf{T}}$ is equivalent to $\exists x\, \mathsf{Prf}_T(x, \ulcorner \gamma_{\mathbf{T}} \urcorner)$. So $\mathbf{T}$ is $\omega$-inconsistent. □

**Theorem 34.6.** *Let $\mathbf{T}$ be any $\omega$-consistent, axiomatizable theory extending $\mathbf{Q}$. Then $\mathbf{T}$ is not complete.*

*Proof.* If $\mathbf{T}$ is $\omega$-consistent, it is consistent, so $\mathbf{T} \nvdash \gamma_{\mathbf{T}}$ by Lemma 34.3. By Lemma 34.5, $\mathbf{T} \nvdash \neg \gamma_{\mathbf{T}}$. This means that $\mathbf{T}$ is incomplete, since it derives neither $\gamma_{\mathbf{T}}$ nor $\neg \gamma_{\mathbf{T}}$. □

## 34.4 Rosser's Theorem

Can we modify Gödel's proof to get a stronger result, replacing "$\omega$-consistent" with simply "consistent"? The answer is "yes," using a trick discovered by Rosser. Rosser's trick is to use a "modified" derivability predicate $\mathsf{RProv}_T(y)$ instead of $\mathsf{Prov}_T(y)$.

**Theorem 34.7.** *Let* **T** *be any consistent, axiomatizable theory extending* **Q**. *Then* **T** *is not complete.*

*Proof.* Recall that $\mathsf{Prov}_T(y)$ is defined as $\exists x\, \mathsf{Prf}_T(x, y)$, where $\mathsf{Prf}_T(x, y)$ represents the decidable relation which holds iff $x$ is the Gödel number of a derivation of the sentence with Gödel number $y$. The relation that holds between $x$ and $y$ if $x$ is the Gödel number of a *refutation* of the sentence with Gödel number $y$ is also decidable. Let $\mathrm{not}(x)$ be the primitive recursive function which does the following: if $x$ is the code of a formula $\varphi$, $\mathrm{not}(x)$ is a code of $\neg\varphi$. Then $\mathsf{Ref}_T(x, y)$ holds iff $\mathsf{Prf}_T(x, \mathrm{not}(y))$. Let $\mathsf{Ref}_T(x, y)$ represent it. Then, if $\mathbf{T} \vdash \neg\varphi$ and $\delta$ is a corresponding derivation, $\mathbf{Q} \vdash \mathsf{Ref}_T(\ulcorner\delta\urcorner, \ulcorner\varphi\urcorner)$. We define $\mathsf{RProv}_T(y)$ as

$$\exists x\, (\mathsf{Prf}_T(x, y) \wedge \forall z\, (z < x \to \neg\mathsf{Ref}_T(z, y))).$$

Roughly, $\mathsf{RProv}_T(y)$ says "there is a proof of $y$ in **T**, and there is no shorter refutation of $y$." Assuming **T** is consistent, $\mathsf{RProv}_T(y)$ is true of the same numbers as $\mathsf{Prov}_T(y)$; but from the point of view of *provability* in **T** (and we now know that there is a difference between truth and provability!) the two have different properties. If **T** is *in*consistent, then the two do *not* hold of the same numbers! ($\mathsf{RProv}_T(y)$ is often read as "$y$ is Rosser provable." Since, as just discussed, Rosser provability is not some special kind of provability—in inconsistent theories, there are sentences that are provable but not Rosser provable—this may be confusing. To avoid the confusion, you could instead read it as "$y$ is shmovable.")

By the fixed-point lemma, there is a formula $\rho_{\mathbf{T}}$ such that

$$\mathbf{Q} \vdash \rho_{\mathbf{T}} \leftrightarrow \neg\mathsf{RProv}_T(\ulcorner\rho_{\mathbf{T}}\urcorner). \tag{34.4}$$

In contrast to the proof of Theorem 34.6, here we claim that if **T** is consistent, **T** doesn't derive $\rho_{\mathbf{T}}$, and **T** also doesn't derive $\neg\rho_{\mathbf{T}}$. (In other words, we don't need the assumption of $\omega$-consistency.)

First, let's show that $\mathbf{T} \nvdash \rho_T$. Suppose it did, so there is a derivation of $\rho_T$ from $T$; let $n$ be its Gödel number. Then $\mathbf{Q} \vdash \mathsf{Prf}_T(\overline{n}, \ulcorner\rho_T\urcorner)$, since $\mathsf{Prf}_T$ represents $\mathrm{Prf}_T$ in **Q**. Also, for each $k < n$, $k$ is not the Gödel number of $\neg\rho_T$, since **T** is consistent. So for each $k < n$, $\mathbf{Q} \vdash \neg\mathsf{Ref}_T(\overline{k}, \ulcorner\rho_T\urcorner)$. By Lemma 32.22(2), $\mathbf{Q} \vdash \forall z\, (z < \overline{n} \to \neg\mathsf{Ref}_T(z, \ulcorner\rho_T\urcorner))$. Thus,

$$\mathbf{Q} \vdash \exists x\, (\mathsf{Prf}_T(x, \ulcorner\rho_T\urcorner) \wedge \forall z\, (z < x \to \neg\mathsf{Ref}_T(z, \ulcorner\rho_T\urcorner))),$$

but that's just $\mathsf{RProv}_T(\ulcorner \rho_T \urcorner)$. By eq. (34.4), $\mathbf{Q} \vdash \neg\rho_T$. Since $\mathbf{T}$ extends $\mathbf{Q}$, also $\mathbf{T} \vdash \neg\rho_T$. We've assumed that $\mathbf{T} \vdash \rho_T$, so $\mathbf{T}$ would be inconsistent, contrary to the assumption of the theorem.

Now, let's show that $\mathbf{T} \nvdash \neg\rho_T$. Again, suppose it did, and suppose $n$ is the Gödel number of a derivation of $\neg\rho_T$. Then $\mathsf{Ref}_T(n, {}^{\#}\rho_T{}^{\#})$ holds, and since $\mathsf{Ref}_T$ represents $\mathsf{Ref}_T$ in $\mathbf{Q}$, $\mathbf{Q} \vdash \mathsf{Ref}_T(\overline{n}, \ulcorner \rho_T \urcorner)$. We'll again show that $\mathbf{T}$ would then be inconsistent because it would also derive $\rho_T$. Since $\mathbf{Q} \vdash \rho_T \leftrightarrow \neg\mathsf{RProv}_T(\ulcorner \rho_T \urcorner)$, and since $\mathbf{T}$ extends $\mathbf{Q}$, it suffices to show that $\mathbf{Q} \vdash \neg\mathsf{RProv}_T(\ulcorner \rho_T \urcorner)$. The sentence $\neg\mathsf{RProv}_T(\ulcorner \rho_T \urcorner)$, i.e.,

$$\neg\exists x\,(\mathsf{Prf}_T(x, \ulcorner \rho_T \urcorner) \wedge \forall z\,(z < x \to \neg\mathsf{Ref}_T(z, \ulcorner \rho_T \urcorner)))$$

is logically equivalent to

$$\forall x\,(\mathsf{Prf}_T(x, \ulcorner \rho_T \urcorner) \to \exists z\,(z < x \wedge \mathsf{Ref}_T(z, \ulcorner \rho_T \urcorner)))$$

We argue informally using logic, making use of facts about what $\mathbf{Q}$ derives. Suppose $x$ is arbitrary and $\mathsf{Prf}_T(x, \ulcorner \rho_T \urcorner)$. We already know that $\mathbf{T} \nvdash \rho_T$, and so for every $k$, $\mathbf{Q} \vdash \neg\mathsf{Prf}_T(\overline{k}, \ulcorner \rho_T \urcorner)$. Thus, for every $k$ it follows that $x \neq \overline{k}$. In particular, we have (a) that $x \neq \overline{n}$. We also have $\neg(x = \overline{0} \vee x = \overline{1} \vee \dots \vee x = \overline{n-1})$ and so by Lemma 32.22(2), (b) $\neg(x < \overline{n})$. By Lemma 32.23, $\overline{n} < x$. Since $\mathbf{Q} \vdash \mathsf{Ref}_T(\overline{n}, \ulcorner \rho_T \urcorner)$, we have $\overline{n} < x \wedge \mathsf{Ref}_T(\overline{n}, \ulcorner \rho_T \urcorner)$, and from that $\exists z\,(z < x \wedge \mathsf{Ref}_T(z, \ulcorner \rho_T \urcorner))$. Since $x$ was arbitrary we get

$$\forall x\,(\mathsf{Prf}_T(x, \ulcorner \rho_T \urcorner) \to \exists z\,(z < x \wedge \mathsf{Ref}_T(z, \ulcorner \rho_T \urcorner)))$$

as required. □

## 34.5 Comparison with Gödel's Original Paper

It is worthwhile to spend some time with Gödel's 1931 paper. The introduction sketches the ideas we have just discussed. Even if you just skim through the paper, it is easy to see what is going on at each stage: first Gödel describes the formal system $P$ (syntax, axioms, proof rules); then he defines the primitive recursive functions and relations; then he shows that $xBy$ is primitive recursive, and argues that the primitive recursive functions and relations are represented in $\mathbf{P}$. He then goes on to prove the incompleteness theorem, as above. In section 3, he shows that one can take the unprovable assertion to be a sentence in the language of arithmetic. This is the origin of the $\beta$-lemma, which is what we also used to handle sequences in showing that the recursive functions are representable in $\mathbf{Q}$. Gödel doesn't go so far to isolate a minimal set of axioms that suffice, but we now know that $\mathbf{Q}$ will do the trick. Finally, in Section 4, he sketches a proof of the second incompleteness theorem.

## 34.6 The Derivability Conditions for PA

Peano arithmetic, or **PA**, is the theory extending **Q** with induction axioms for all formulas. In other words, one adds to **Q** axioms of the form

$$(\varphi(0) \wedge \forall x\,(\varphi(x) \to \varphi(x'))) \to \forall x\,\varphi(x)$$

for every formula $\varphi$. Notice that this is really a *schema*, which is to say, infinitely many axioms (and it turns out that **PA** is *not* finitely axiomatizable). But since one can effectively determine whether or not a string of symbols is an instance of an induction axiom, the set of axioms for **PA** is computable. **PA** is a much more robust theory than **Q**. For example, one can easily prove that addition and multiplication are commutative, using induction in the usual way. In fact, most finitary number-theoretic and combinatorial arguments can be carried out in **PA**.

Since **PA** is computably axiomatized, the derivability predicate $\mathsf{Prf}_{\mathbf{PA}}(x, y)$ is computable and hence represented in **Q** (and so, in **PA**). As before, I will take $\mathsf{Prf}_{\mathbf{PA}}(x, y)$ to denote the formula representing the relation. Let $\mathsf{Prov}_{\mathbf{PA}}(y)$ be the formula $\exists x\,\mathsf{Prf}_{\mathbf{PA}}(x, y)$, which, intuitively says, "$y$ is provable from the axioms of **PA**." The reason we need a little bit more than the axioms of **Q** is we need to know that the theory we are using is strong enough to derive a few basic facts about this derivability predicate. In fact, what we need are the following facts:

P1. If $\mathbf{PA} \vdash \varphi$, then $\mathbf{PA} \vdash \mathsf{Prov}_{\mathbf{PA}}(\ulcorner\varphi\urcorner)$

P2. For all formulas $\varphi$ and $\psi$,

$$\mathbf{PA} \vdash \mathsf{Prov}_{\mathbf{PA}}(\ulcorner\varphi \to \psi\urcorner) \to (\mathsf{Prov}_{\mathbf{PA}}(\ulcorner\varphi\urcorner) \to \mathsf{Prov}_{\mathbf{PA}}(\ulcorner\psi\urcorner))$$

P3. For every formula $\varphi$,

$$\mathbf{PA} \vdash \mathsf{Prov}_{\mathbf{PA}}(\ulcorner\varphi\urcorner) \to \mathsf{Prov}_{\mathbf{PA}}(\ulcorner\mathsf{Prov}_{\mathbf{PA}}(\ulcorner\varphi\urcorner)\urcorner).$$

The only way to verify that these three properties hold is to describe the formula $\mathsf{Prov}_{\mathbf{PA}}(y)$ carefully and use the axioms of **PA** to describe the relevant formal proofs. Conditions (1) and (2) are easy; it is really condition (3) that requires work. (Think about what kind of work it entails ...) Carrying out the details would be tedious and uninteresting, so here we will ask you to take it on faith that **PA** has the three properties listed above. A reasonable choice of $\mathsf{Prov}_{\mathbf{PA}}(y)$ will also satisfy

P4. If $\mathbf{PA} \vdash \mathsf{Prov}_{\mathbf{PA}}(\ulcorner\varphi\urcorner)$, then $\mathbf{PA} \vdash \varphi$.

But we will not need this fact.

Incidentally, Gödel was lazy in the same way we are being now. At the end of the 1931 paper, he sketches the proof of the second incompleteness

theorem, and promises the details in a later paper. He never got around to it; since everyone who understood the argument believed that it could be carried out (he did not need to fill in the details.)

## 34.7 The Second Incompleteness Theorem

How can we express the assertion that **PA** doesn't prove its own consistency? Saying **PA** is inconsistent amounts to saying that $\mathbf{PA} \vdash 0 = 1$. So we can take the consistency statement $\mathsf{Con}_{\mathbf{PA}}$ to be the sentence $\neg\mathsf{Prov}_{\mathbf{PA}}(\ulcorner 0 = 1 \urcorner)$, and then the following theorem does the job:

**Theorem 34.8.** *Assuming* **PA** *is consistent, then* **PA** *does not derive* $\mathsf{Con}_{\mathbf{PA}}$.

It is important to note that the theorem depends on the particular representation of $\mathsf{Con}_{\mathbf{PA}}$ (i.e., the particular representation of $\mathsf{Prov}_{\mathbf{PA}}(y)$). All we will use is that the representation of $\mathsf{Prov}_{\mathbf{PA}}(y)$ satisfies the three derivability conditions, so the theorem generalizes to any theory with a derivability predicate having these properties.

It is informative to read Gödel's sketch of an argument, since the theorem follows like a good punch line. It goes like this. Let $\gamma_{\mathbf{PA}}$ be the Gödel sentence that we constructed in the proof of Theorem 34.6. We have shown "If **PA** is consistent, then **PA** does not derive $\gamma_{\mathbf{PA}}$." If we formalize this *in* **PA**, we have a proof of

$$\mathsf{Con}_{\mathbf{PA}} \to \neg\mathsf{Prov}_{\mathbf{PA}}(\ulcorner \gamma_{\mathbf{PA}} \urcorner).$$

Now suppose **PA** derives $\mathsf{Con}_{\mathbf{PA}}$. Then it derives $\neg\mathsf{Prov}_{\mathbf{PA}}(\ulcorner \gamma_{\mathbf{PA}} \urcorner)$. But since $\gamma_{\mathbf{PA}}$ is a Gödel sentence, this is equivalent to $\gamma_{\mathbf{PA}}$. So **PA** derives $\gamma_{\mathbf{PA}}$.

But: we know that if **PA** is consistent, it doesn't derive $\gamma_{\mathbf{PA}}$! So if **PA** is consistent, it can't derive $\mathsf{Con}_{\mathbf{PA}}$.

To make the argument more precise, we will let $\gamma_{\mathbf{PA}}$ be the Gödel sentence for **PA** and use the derivability conditions (P1)–(P3) to show that **PA** derives $\mathsf{Con}_{\mathbf{PA}} \to \gamma_{\mathbf{PA}}$. This will show that **PA** doesn't derive $\mathsf{Con}_{\mathbf{PA}}$. Here is a sketch

of the proof, in **PA**. (For simplicity, we drop the **PA** subscripts.)

$$\gamma \leftrightarrow \neg \mathsf{Prov}(\ulcorner \gamma \urcorner) \tag{34.5}$$

$\gamma$ is a Gödel sentence

$$\gamma \to \neg \mathsf{Prov}(\ulcorner \gamma \urcorner) \tag{34.6}$$

from eq. (34.5)

$$\gamma \to (\mathsf{Prov}(\ulcorner \gamma \urcorner) \to \bot) \tag{34.7}$$

from eq. (34.6) by logic

$$\mathsf{Prov}(\ulcorner \gamma \to (\mathsf{Prov}(\ulcorner \gamma \urcorner) \to \bot) \urcorner) \tag{34.8}$$

by from eq. (34.7) by condition P1

$$\mathsf{Prov}(\ulcorner \gamma \urcorner) \to \mathsf{Prov}(\ulcorner (\mathsf{Prov}(\ulcorner \gamma \urcorner) \to \bot) \urcorner) \tag{34.9}$$

from eq. (34.8) by condition P2

$$\mathsf{Prov}(\ulcorner \gamma \urcorner) \to (\mathsf{Prov}(\ulcorner \mathsf{Prov}(\ulcorner \gamma \urcorner) \urcorner) \to \mathsf{Prov}(\ulcorner \bot \urcorner)) \tag{34.10}$$

from eq. (34.9) by condition P2 and logic

$$\mathsf{Prov}(\ulcorner \gamma \urcorner) \to \mathsf{Prov}(\ulcorner \mathsf{Prov}(\ulcorner \gamma \urcorner) \urcorner) \tag{34.11}$$

by P3

$$\mathsf{Prov}(\ulcorner \gamma \urcorner) \to \mathsf{Prov}(\ulcorner \bot \urcorner) \tag{34.12}$$

from eq. (34.10) and eq. (34.11) by logic

$$\mathsf{Con} \to \neg \mathsf{Prov}(\ulcorner \gamma \urcorner) \tag{34.13}$$

contraposition of eq. (34.12) and $\mathsf{Con} \equiv \neg \mathsf{Prov}(\ulcorner \bot \urcorner)$

$$\mathsf{Con} \to \gamma$$

from eq. (34.5) and eq. (34.13) by logic

The use of logic in the above just elementary facts from propositional logic, e.g., eq. (34.7) uses $\vdash \neg\varphi \leftrightarrow (\varphi \to \bot)$ and eq. (34.12) uses $\varphi \to (\psi \to \chi), \varphi \to \psi \vdash \varphi \to \chi$. The use of condition P2 in eq. (34.9) and eq. (34.10) relies on instances of P2, $\mathsf{Prov}(\ulcorner \varphi \to \psi \urcorner) \to (\mathsf{Prov}(\ulcorner \varphi \urcorner) \to \mathsf{Prov}(\ulcorner \psi \urcorner))$. In the first one, $\varphi \equiv \gamma$ and $\psi \equiv \mathsf{Prov}(\ulcorner \gamma \urcorner) \to \bot$; in the second, $\varphi \equiv \mathsf{Prov}(\ulcorner G \urcorner)$ and $\psi \equiv \bot$.

The more abstract version of the second incompleteness theorem is as follows:

**Theorem 34.9.** *Let* **T** *be any consistent, axiomatized theory extending* **Q** *and let* $\mathsf{Prov}_T(y)$ *be any formula satisfying derivability conditions P1–P3 for* **T**. *Then* **T** *does not derive* $\mathsf{Con}_T$.

The moral of the story is that no "reasonable" consistent theory for mathematics can derive its own consistency statement. Suppose **T** is a theory of mathematics that includes **Q** and Hilbert's "finitary" reasoning (whatever that may be). Then, the whole of **T** cannot derive the consistency statement of **T**, and so, a fortiori, the finitary fragment can't derive the consistency statement

of **T** either. In that sense, there cannot be a finitary consistency proof for "all of mathematics."

There is some leeway in interpreting the term "finitary," and Gödel, in the 1931 paper, grants the possibility that something we may consider "finitary" may lie outside the kinds of mathematics Hilbert wanted to formalize. But Gödel was being charitable; today, it is hard to see how we might find something that can reasonably be called finitary but is not formalizable in, say, **ZFC**.

## 34.8 Löb's Theorem

The Gödel sentence for a theory **T** is a fixed point of $\neg\mathsf{Prov}_T(x)$, i.e., a sentence $\gamma$ such that

$$\mathbf{T} \vdash \neg\mathsf{Prov}_T(\ulcorner\gamma\urcorner) \leftrightarrow \gamma.$$

It is not derivable, because if $\mathbf{T} \vdash \gamma$, (a) by derivability condition (1), $\mathbf{T} \vdash \mathsf{Prov}_T(\ulcorner\gamma\urcorner)$, and (b) $\mathbf{T} \vdash \gamma$ together with $\mathbf{T} \vdash \neg\mathsf{Prov}_T(\ulcorner\gamma\urcorner) \leftrightarrow \gamma$ gives $\mathbf{T} \vdash \neg\mathsf{Prov}_T(\ulcorner\gamma\urcorner)$, and so **T** would be inconsistent. Now it is natural to ask about the status of a fixed point of $\mathsf{Prov}_T(x)$, i.e., a sentence $\delta$ such that

$$\mathbf{T} \vdash \mathsf{Prov}_T(\ulcorner\delta\urcorner) \leftrightarrow \delta.$$

If it were derivable, $\mathbf{T} \vdash \mathsf{Prov}_T(\ulcorner\delta\urcorner)$ by condition (1), but the same conclusion follows if we apply modus ponens to the equivalence above. Hence, we don't get that **T** is inconsistent, at least not by the same argument as in the case of the Gödel sentence. This of course does not show that **T** *does* derive $\delta$.

We can make headway on this question if we generalize it a bit. The left-to-right direction of the fixed point equivalence, $\mathsf{Prov}_T(\ulcorner\delta\urcorner) \to \delta$, is an instance of a general schema called a *reflection principle*: $\mathsf{Prov}_T(\ulcorner\varphi\urcorner) \to \varphi$. It is called that because it expresses, in a sense, that **T** can "reflect" about what it can derive; basically it says, "If **T** can derive $\varphi$, then $\varphi$ is true," for any $\varphi$. This is true for sound theories only, of course, and this suggests that theories will in general not derive every instance of it. So which instances can a theory (strong enough, and satisfying the derivability conditions) derive? Certainly all those where $\varphi$ itself is derivable. And that's it, as the next result shows.

**Theorem 34.10.** *Let* **T** *be an axiomatizable theory extending* **Q**, *and suppose* $\mathsf{Prov}_T(y)$ *is a formula satisfying conditions P1–P3 from section 34.7. If* **T** *derives* $\mathsf{Prov}_T(\ulcorner\varphi\urcorner) \to \varphi$, *then in fact* **T** *derives* $\varphi$.

Put differently, if $\mathbf{T} \nvdash \varphi$, then $\mathbf{T} \nvdash \mathsf{Prov}_T(\ulcorner\varphi\urcorner) \to \varphi$. This result is known as Löb's theorem.

The heuristic for the proof of Löb's theorem is a clever proof that Santa Claus exists. (If you don't like that conclusion, you are free to substitute any other conclusion you would like.) Here it is:

1. Let $X$ be the sentence, "If $X$ is true, then Santa Claus exists."

2. Suppose $X$ is true.

3. Then what it says holds; i.e., we have: if $X$ is true, then Santa Claus exists.

4. Since we are assuming $X$ is true, we can conclude that Santa Claus exists, by modus ponens from (2) and (3).

5. We have succeeded in deriving (4), "Santa Claus exists," from the assumption (2), "$X$ is true." By conditional proof, we have shown: "If $X$ is true, then Santa Claus exists."

6. But this is just the sentence $X$. So we have shown that $X$ is true.

7. But then, by the argument (2)–(4) above, Santa Claus exists.

A formalization of this idea, replacing "is true" with "is derivable," and "Santa Claus exists" with $\varphi$, yields the proof of Löb's theorem. The trick is to apply the fixed-point lemma to the formula $\mathsf{Prov}_T(y) \to \varphi$. The fixed point of that corresponds to the sentence $X$ in the preceding sketch.

*Proof.* Suppose $\varphi$ is a sentence such that $\mathbf{T}$ derives $\mathsf{Prov}_T(\ulcorner\varphi\urcorner) \to \varphi$. Let $\psi(y)$ be the formula $\mathsf{Prov}_T(y) \to \varphi$, and use the fixed-point lemma to find a sentence $\theta$ such that $\mathbf{T}$ derives $\theta \leftrightarrow \psi(\ulcorner\theta\urcorner)$. Then each of the following is derivable

in **T**:

$$\theta \leftrightarrow (\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \varphi) \tag{34.14}$$

$\theta$ is a fixed point of $\psi(y)$

$$\theta \to (\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \varphi) \tag{34.15}$$

from eq. (34.14)

$$\mathsf{Prov}_T(\ulcorner\theta \to (\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \varphi)\urcorner) \tag{34.16}$$

from eq. (34.15) by condition P1

$$\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \mathsf{Prov}_T(\ulcorner\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \varphi\urcorner) \tag{34.17}$$

from eq. (34.16) using condition P2

$$\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to (\mathsf{Prov}_T(\ulcorner\mathsf{Prov}_T(\ulcorner\theta\urcorner)\urcorner) \to \mathsf{Prov}_T(\ulcorner\varphi\urcorner)) \tag{34.18}$$

from eq. (34.17) using P2 again

$$\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \mathsf{Prov}_T(\ulcorner\mathsf{Prov}_T(\ulcorner\theta\urcorner)\urcorner) \tag{34.19}$$

by derivability condition P3

$$\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \mathsf{Prov}_T(\ulcorner\varphi\urcorner) \tag{34.20}$$

from eq. (34.18) and eq. (34.19)

$$\mathsf{Prov}_T(\ulcorner\varphi\urcorner) \to \varphi \tag{34.21}$$

by assumption of the theorem

$$\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \varphi \tag{34.22}$$

from eq. (34.20) and eq. (34.21)

$$(\mathsf{Prov}_T(\ulcorner\theta\urcorner) \to \varphi) \to \theta \tag{34.23}$$

from eq. (34.14)

$$\theta \tag{34.24}$$

from eq. (34.22) and eq. (34.23)

$$\mathsf{Prov}_T(\ulcorner\theta\urcorner) \tag{34.25}$$

from eq. (34.24) by condition P1

$\varphi$ from eq. (34.21) and eq. (34.25)

□

With Löb's theorem in hand, there is a short proof of the first incompleteness theorem (for theories having a derivability predicate satisfying conditions P1–P3: if $\mathbf{T} \vdash \mathsf{Prov}_T(\ulcorner\bot\urcorner) \to \bot$, then $\mathbf{T} \vdash \bot$. If **T** is consistent, $\mathbf{T} \nvdash \bot$. So, $\mathbf{T} \nvdash \mathsf{Prov}_T(\ulcorner\bot\urcorner) \to \bot$, i.e., $\mathbf{T} \nvdash \mathsf{Con}_{\mathbf{T}}$. We can also apply it to show that $\delta$, the fixed point of $\mathsf{Prov}_T(x)$, is derivable. For since

$$\mathbf{T} \vdash \mathsf{Prov}_T(\ulcorner\delta\urcorner) \leftrightarrow \delta$$

in particular

$$\mathbf{T} \vdash \mathsf{Prov}_T(\ulcorner \delta \urcorner) \to \delta$$

and so by Löb's theorem, $\mathbf{T} \vdash \delta$.

## 34.9 The Undefinability of Truth

The notion of *definability* depends on having a formal semantics for the language of arithmetic. We have described a set of formulas and sentences in the language of arithmetic. The "intended interpretation" is to read such sentences as making assertions about the natural numbers, and such an assertion can be true or false. Let $\mathfrak{N}$ be the structure with domain $\mathbb{N}$ and the standard interpretation for the symbols in the language of arithmetic. Then $\mathfrak{N} \vDash \varphi$ means "$\varphi$ is true in the standard interpretation."

**Definition 34.11.** A relation $R(x_1, \ldots, x_k)$ of natural numbers is *definable* in $\mathfrak{N}$ if and only if there is a formula $\varphi(x_1, \ldots, x_k)$ in the language of arithmetic such that for every $n_1, \ldots, n_k$, $R(n_1, \ldots, n_k)$ if and only if $\mathfrak{N} \vDash \varphi(\overline{n}_1, \ldots, \overline{n}_k)$.

Put differently, a relation is definable in in $\mathfrak{N}$ if and only if it is representable in the theory **TA**, where $\mathbf{TA} = \{\varphi : \mathfrak{N} \vDash \varphi\}$ is the set of true sentences of arithmetic. (If this is not immediately clear to you, you should go back and check the definitions and convince yourself that this is the case.)

**Lemma 34.12.** *Every computable relation is definable in* $\mathfrak{N}$.

*Proof.* It is easy to check that the formula representing a relation in **Q** defines the same relation in $\mathfrak{N}$. □

Now one can ask, is the converse also true? That is, is every relation definable in $\mathfrak{N}$ computable? The answer is no. For example:

**Lemma 34.13.** *The halting relation is definable in* $\mathfrak{N}$.

*Proof.* Let $H$ be the halting relation, i.e.,

$$H = \{\langle e, x\rangle : \exists s\, T(e, x, s)\}.$$

Let $\theta_T$ define $T$ in $\mathfrak{N}$. Then

$$H = \{\langle e, x\rangle : \mathfrak{N} \vDash \exists s\, \theta_T(\overline{e}, \overline{x}, s)\},$$

so $\exists s\, \theta_T(z, x, s)$ defines $H$ in $\mathfrak{N}$. □

What about **TA** itself? Is it definable in arithmetic? That is: is the set $\{{}^{\#}\varphi^{\#} : \mathfrak{N} \vDash \varphi\}$ definable in arithmetic? Tarski's theorem answers this in the negative.

**Theorem 34.14.** *The set of true statements of arithmetic is not definable in arithmetic.*

*Proof.* Suppose $\theta(x)$ defined it. By the fixed-point lemma, there is a formula $\varphi$ such that $\mathbf{Q} \vdash \varphi \leftrightarrow \neg\theta(\ulcorner\varphi\urcorner)$, and hence $\mathfrak{N} \vDash \varphi \leftrightarrow \neg\theta(\ulcorner\varphi\urcorner)$. But then $\mathfrak{N} \vDash \varphi$ if and only if $\mathfrak{N} \vDash \neg\theta(\ulcorner\varphi\urcorner)$, which contradicts the fact that $\theta(y)$ is supposed to define the set of true statements of arithmetic. □

Tarski applied this analysis to a more general philosophical notion of truth. Given any language $L$, Tarski argued that an adequate notion of truth for $L$ would have to satisfy, for each sentence $X$,

> '$X$' is true if and only if $X$.

Tarski's oft-quoted example, for English, is the sentence

> 'Snow is white' is true if and only if snow is white.

However, for any language strong enough to represent the diagonal function, and any linguistic predicate $T(x)$, we can construct a sentence $X$ satisfying "$X$ if and only if not $T(\text{'}X\text{'})$." Given that we do not want a truth predicate to declare some sentences to be both true and false, Tarski concluded that one cannot specify a truth predicate for all sentences in a language without, somehow, stepping outside the bounds of the language. In other words, a the truth predicate for a language cannot be defined in the language itself.

## Problems

**Problem 34.1.** Every $\omega$-consistent theory is consistent. Show that the converse does not hold, i.e., that there are consistent but $\omega$-inconsistent theories. Do this by showing that $\mathbf{Q} \cup \{\neg\gamma_{\mathbf{Q}}\}$ is consistent but $\omega$-inconsistent.

**Problem 34.2.** Show that **PA** derives $\gamma_{\mathbf{PA}} \to \mathsf{Con}_{\mathbf{PA}}$.

**Problem 34.3.** Let **T** be a computably axiomatized theory, and let $\mathsf{Prov}_T$ be a derivability predicate for **T**. Consider the following four statements:

1. If $T \vdash \varphi$, then $T \vdash \mathsf{Prov}_T(\ulcorner\varphi\urcorner)$.
2. $T \vdash \varphi \to \mathsf{Prov}_T(\ulcorner\varphi\urcorner)$.
3. If $T \vdash \mathsf{Prov}_T(\ulcorner\varphi\urcorner)$, then $T \vdash \varphi$.
4. $T \vdash \mathsf{Prov}_T(\ulcorner\varphi\urcorner) \to \varphi$

Under what conditions are each of these statements true?

**Problem 34.4.** Show that $Q(n) \Leftrightarrow n \in \{{}^{\#}\varphi^{\#} : \mathbf{Q} \vdash \varphi\}$ is definable in arithmetic.

# Part VIII

# Second-order Logic

This is the beginnings of a part on second-order logic.

# Chapter 35

# Syntax and Semantics

Basic syntax and semantics for SOL covered so far. As a chapter it's too short. Substitution for second-order variables has to be covered to be able to talk about derivation systems for SOL, and there's some subtle issues there.

## 35.1 Introduction

In first-order logic, we combine the non-logical symbols of a given language, i.e., its constant symbols, function symbols, and predicate symbols, with the logical symbols to express things about first-order structures. This is done using the notion of satisfaction, which relates a structure $\mathfrak{M}$, together with a variable assignment $s$, and a formula $\varphi$: $\mathfrak{M}, s \vDash \varphi$ holds iff what $\varphi$ expresses when its constant symbols, function symbols, and predicate symbols are interpreted as $\mathfrak{M}$ says, and its free variables are interpreted as $s$ says, is true. The interpretation of the identity predicate $=$ is built into the definition of $\mathfrak{M}, s \vDash \varphi$, as is the interpretation of $\forall$ and $\exists$. The former is always interpreted as the identity relation on the domain $|\mathfrak{M}|$ of the structure, and the quantifiers are always interpreted as ranging over the entire domain. But, crucially, quantification is only allowed over elements of the domain, and so only object variables are allowed to follow a quantifier.

In second-order logic, both the language and the definition of satisfaction are extended to include free and bound function and predicate variables, and quantification over them. These variables are related to function symbols and predicate symbols the same way that object variables are related to constant symbols. They play the same role in the formation of terms and formulas of second-order logic, and quantification over them is handled in a similar way. In the *standard* semantics, the second-order quantifiers range over all possible objects of the right type ($n$-place functions from $|\mathfrak{M}|$ to $|\mathfrak{M}|$ for func-

tion variables, $n$-place relations for predicate variables). For instance, while $\forall v_0 \, (P^1_0(v_0) \vee \neg P^1_0(v_0))$ is a formula in both first- and second-order logic, in the latter we can also consider $\forall V^1_0 \, \forall v_0 \, (V^1_0(v_0) \vee \neg V^1_0(v_0))$ and $\exists V^1_0 \, \forall v_0 \, (V^1_0(v_0) \vee \neg V^1_0(v_0))$. Since these contain no free varaibles, they are sentences of second-order logic. Here, $V^1_0$ is a second-order 1-place predicate variable. The allowable interpretations of $V^1_0$ are the same that we can assign to a 1-place predicate symbol like $P^1_0$, i.e., subsets of $|\mathfrak{M}|$. Quantification over them then amounts to saying that $\forall v_0 \, (V^1_0(v_0) \vee \neg V^1_0(v_0))$ holds for all ways of assigning a subset of $|\mathfrak{M}|$ as the value of $V^1_0$, or for at least one. Since every set either contains or fails to contain a given object, both are true in any structure.

## 35.2 Terms and Formulas

Like in first-order logic, expressions of second-order logic are built up from a basic vocabulary containing *variables*, *constant symbols*, *predicate symbols* and sometimes *function symbols*. From them, together with logical connectives, quantifiers, and punctuation symbols such as parentheses and commas, *terms* and *formulas* are formed. The difference is that in addition to variables for objects, second-order logic also contains variables for relations and functions, and allows quantification over them. So the logical symbols of second-order logic are those of first-order logic, plus:

1. A denumerable set of second-order relation variables of every arity $n$: $V^n_0, V^n_1, V^n_2, \ldots$
2. A denumerable set of second-order function variables: $u^n_0, u^n_1, u^n_2, \ldots$

Just as we use $x, y, z$ as meta-variables for first-order variables $v_i$, we'll use $X, Y, Z$, etc., as metavariables for $V^n_i$ and $u, v$, etc., as meta-variables for $u^n_i$.

The non-logical symbols of a second-order language are specified the same way a first-order language is: by listing its constant symbols, function symbols, and predicate symbols

In first-order logic, the identity predicate $=$ is usually included. In first-order logic, the non-logical symbols of a language $\mathcal{L}$ are crucial to allow us to express anything interesting. There are of course sentences that use no non-logical symbols, but with only $=$ it is hard to say anything interesting. In second-order logic, since we have an unlimited supply of relation and function variables, we can say anything we can say in a first-order language even without a special supply of non-logical symbols.

**Definition 35.1** (Second-order Terms)**.** The set of *second-order terms* of $\mathcal{L}$, $\mathrm{Trm}^2(\mathcal{L})$, is defined by adding to Definition 12.4 the clause

1. If $u$ is an $n$-place function variable and $t_1, \ldots, t_n$ are terms, then $u(t_1, \ldots, t_n)$ is a term.

So, a second-order term looks just like a first-order term, except that where a first-order term contains a function symbol $f_i^n$, a second-order term may contain a function variable $u_i^n$ in its place.

**Definition 35.2** (Second-order formula)**.** The set of *second-order formulas* $\mathrm{Frm}^2(\mathcal{L})$ of the language $\mathcal{L}$ is defined by adding to Definition 12.4 the clauses

1. If $X$ is an $n$-place predicate variable and $t_1, \ldots, t_n$ are second-order terms of $\mathcal{L}$, then $X(t_1, \ldots, t_n)$ is an atomic formula.
2. If $\varphi$ is a formula and $u$ is a function variable, then $\forall u\, \varphi$ is a formula.
3. If $\varphi$ is a formula and $X$ is a predicate variable, then $\forall X\, \varphi$ is a formula.
4. If $\varphi$ is a formula and $u$ is a function variable, then $\exists u\, \varphi$ is a formula.
5. If $\varphi$ is a formula and $X$ is a predicate variable, then $\exists X\, \varphi$ is a formula.

## 35.3 Satisfaction

To define the satisfaction relation $\mathfrak{M}, s \vDash \varphi$ for second-order formulas, we have to extend the definitions to cover second-order variables. The notion of a structure is the same for second-order logic as it is for first-order logic. There is only a diffence for variable assignments $s$: these now must not just provide values for the first-order variables, but also for the second-order variables.

**Definition 35.3** (Variable Assignment)**.** A *variable assignment* $s$ for a structure $\mathfrak{M}$ is a function which maps each

1. object variable $v_i$ to an element of $|\mathfrak{M}|$, i.e., $s(v_i) \in |\mathfrak{M}|$
2. $n$-place relation variable $V_i^n$ to an $n$-place relation on $|\mathfrak{M}|$, i.e., $s(V_i^n) \subseteq |\mathfrak{M}|^n$;
3. $n$-place function variable $u_i^n$ to an $n$-place function from $|\mathfrak{M}|$ to $|\mathfrak{M}|$, i.e., $s(u_i^n)\colon |\mathfrak{M}|^n \to |\mathfrak{M}|$;

A structure assigns a value to each constant symbol and function symbol, and a second-order variable assigns objects and functions to each object and function variable. Together, they let us assign a value to every term.

**Definition 35.4** (Value of a Term)**.** If $t$ is a term of the language $\mathcal{L}$, $\mathfrak{M}$ is a structure for $\mathcal{L}$, and $s$ is a variable assignment for $\mathfrak{M}$, the *value* $\mathrm{Val}_s^{\mathfrak{M}}(t)$ is defined as for first-order terms, plus the following clause:

$t \equiv u(t_1, \ldots, t_n)$:

$$\mathrm{Val}_s^{\mathfrak{M}}(t) = s(u)(\mathrm{Val}_s^{\mathfrak{M}}(t_1), \ldots, \mathrm{Val}_s^{\mathfrak{M}}(t_n)).$$

**Definition 35.5** ($x$-Variant)**.** If $s$ is a variable assignment for a structure $\mathfrak{M}$, then any variable assignment $s'$ for $\mathfrak{M}$ which differs from $s$ at most in what it assigns to $x$ is called an *$x$-variant* of $s$. If $s'$ is an $x$-variant of $s$ we write $s \sim_x s'$. (Similarly for second-order variables $X$ or $u$.)

**Definition 35.6** (Satisfaction)**.** For second-order formulas $\varphi$, the definition of satisfaction is like Definition 12.35 with the addition of:

1. $\varphi \equiv X^n(t_1, \ldots, t_n)$: $\mathfrak{M}, s \vDash \varphi$ iff $\langle \mathrm{Val}^{\mathfrak{M}}_s(t_1), \ldots, \mathrm{Val}^{\mathfrak{M}}_s(t_n)\rangle \in s(X^n)$.
2. $\varphi \equiv \forall X\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for every $X$-variant $s'$ of $s$, $\mathfrak{M}, s' \vDash \psi$.
3. $\varphi \equiv \exists X\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff there is an $X$-variant $s'$ of $s$ so that $\mathfrak{M}, s' \vDash \psi$.
4. $\varphi \equiv \forall u\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff for every $u$-variant $s'$ of $s$, $\mathfrak{M}, s' \vDash \psi$.
5. $\varphi \equiv \exists u\, \psi$: $\mathfrak{M}, s \vDash \varphi$ iff there is an $u$-variant $s'$ of $s$ so that $\mathfrak{M}, s' \vDash \psi$.

**Example 35.7.** Consider the formula $\forall z\, (X(z) \leftrightarrow \neg Y(z))$. It contains no second-order quantifiers, but does contain the second-order variables $X$ and $Y$ (here understood to be one-place). The corresponding first-order sentence $\forall z\, (P(z) \leftrightarrow \neg R(z))$ says that whatever falls under the interpretation of $P$ does not fall under the interpretation of $R$ and vice versa. In a structure, the interpretation of a predicate symbol $P$ is given by the interpretation $M^{\mathfrak{P}}$. But for second-order variables like $X$ and $Y$, the interpretation is provided, not by the structure itself, but by a variable assignment. Since the second-order formula is not a sentence (in includes freee variables $X$ and $Y$), it is only satisfied relative to a structure $\mathfrak{M}$ together with a variable assignment $s$.

$\mathfrak{M}, s \vDash \forall z\, (Xz \leftrightarrow \neg Yz)$ whenever the elements of $s(X)$ are not elements of $s(Y)$, and vice versa, i.e., iff $s(Y) = |\mathfrak{M}| \setminus s(X)$. So for instance, take $|\mathfrak{M}| = \{1, 2, 3\}$. Since no predicate symbols, function symbols, or constant symbols are involved, the domain of $\mathfrak{M}$ is all that is relevant. Now for $s_1(X) = \{1, 2\}$ and $s_1(Y) = \{3\}$, we have $\mathfrak{M}, s_1 \vDash \forall z\, (X(z) \leftrightarrow \neg Y(z))$.

By contrast, if we have $s_2(X) = \{1, 2\}$ and $s_2(Y) = \{2, 3\}$, $\mathfrak{M}, s_2 \nvDash \forall z\, (X(z) \leftrightarrow \neg Y(z))$. That's because there is a $z$-variant $s_2'$ of $s_2$ with $s_2'(z) = 2$ where $\mathfrak{M}, s_2' \vDash X(z)$ (since $2 \in s_2'(X)$) but $\mathfrak{M}, s_2' \nvDash \neg Y(z)$ (since also $s_2'(z) \in s_2'(Y)$).

**Example 35.8.** $\mathfrak{M}, s \vDash \exists Y\, (\exists y\, Y(y) \wedge \forall z\, (X(z) \leftrightarrow \neg Y(z)))$ if there is an $s' \sim_Y s$ such that $\mathfrak{M}, s' \vDash (\exists y\, Y(y) \wedge \forall z\, (X(z) \leftrightarrow \neg Y(z)))$. And that is the case iff $s'(Y) \neq \emptyset$ (so that $\mathfrak{M}, s' \vDash \exists y\, Y(y)$) and, as in the previous example, $s'(Y) = |\mathfrak{M}| \setminus s'(X)$. In other words, $\mathfrak{M}, s \vDash \exists Y\, (\exists y\, Y(y) \wedge \forall z\, (X(z) \leftrightarrow \neg Y(z)))$ iff $|\mathfrak{M}| \setminus s(X)$ is non-empty, i.e., $s(X) \neq |\mathfrak{M}|$. So, the formula is satisfied, e.g., if $|\mathfrak{M}| = \{1, 2, 3\}$ and $s(X) = \{1, 2\}$, but not if $s(X) = \{1, 2, 3\} = |\mathfrak{M}|$.

Since the formula is not satisfied whenever $s(X) = |\mathfrak{M}|$, the sentence

$$\forall X\, \exists Y\, (\exists y\, Y(y) \wedge \forall z\, (X(z) \leftrightarrow \neg Y(z)))$$

is never satisfied: For any structure $\mathfrak{M}$, the assignment $s(X) = |\mathfrak{M}|$ will make the sentence false. On the other hand, the sentence

$$\exists X \, \exists Y \, (\exists y \, Y(y) \wedge \forall z \, (X(z) \leftrightarrow \neg Y(z)))$$

is satisfied relative to any assignment $s$, since we can always find an $X$-variant $s'$ of $s$ with $s'(X) \neq |\mathfrak{M}|$.

## 35.4 Semantic Notions

The central logical notions of *validity*, *entailment*, and *satisfiability* are defined the same way for second-order logic as they are for first-order logic, except that the underlying satisfaction relation is now that for second-order formulas. A second-order sentence, of course, is a formula in which all variables, including predicate and function variables, are bound.

**Definition 35.9** (Validity)**.** A sentence $\varphi$ is *valid*, $\models \varphi$, iff $\mathfrak{M} \models \varphi$ for every structure $\mathfrak{M}$.

**Definition 35.10** (Entailment)**.** A set of sentences $\Gamma$ *entails* a sentence $\varphi$, $\Gamma \models \varphi$, iff for every structure $\mathfrak{M}$ with $\mathfrak{M} \models \Gamma$, $\mathfrak{M} \models \varphi$.

**Definition 35.11** (Satisfiability)**.** A set of sentences $\Gamma$ is *satisfiable* if $\mathfrak{M} \models \Gamma$ for some structure $\mathfrak{M}$. If $\Gamma$ is not satisfiable it is called *unsatisfiable*.

## 35.5 Expressive Power

Quantification over second-order variables is responsible for an immense increase in the expressive power of the language over that of first-order logic. Second-order existential quantification lets us say that functions or relations with certain properties exists. In first-order logic, the only way to do that is to specify a non-logical symbol (i.e., a function symbol or predicate symbol) for this purpose. Second-order universal quantification lets us say that all subsets of, relations on, or functions from the domain to the domain have a property. In first-order logic, we can only say that the subsets, relations, or functions assigned to one of the non-logical symbols of the language have a property. And when we say that subsets, relations, functions exist that have a property, or that all of them have it, we can use second-order quantification in specifying this property as well. This lets us define relations not definable in first-order logic, and express properties of the domain not expressible in first-order logic.

**Definition 35.12.** If $\mathfrak{M}$ is a structure for a language $\mathcal{L}$, a relation $R \subseteq |\mathfrak{M}|^2$ is *definable* in $\mathcal{L}$ if there is some formula $\varphi_R(x, y)$ with only the variables $x$ and $y$ free, such that $R(a, b)$ holds (i.e., $\langle a, b \rangle \in R$) iff $\mathfrak{M}, s \models \varphi_R(x, y)$ for $s(x) = a$ and $s(y) = b$.

**Example 35.13.** In first-order logic we can define the identity relation $\mathrm{Id}_{|\mathfrak{M}|}$ (i.e., $\{\langle a, a\rangle : a \in |\mathfrak{M}|\}$) by the formula $x = y$. In second-order logic, we can define this relation *without* $=$. For if $a$ and $b$ are the same element of $|\mathfrak{M}|$, then they are elements of the same subsets of $|\mathfrak{M}|$ (since sets are determined by their elements). Conversely, if $a$ and $b$ are different, then they are not elements of the same subsets: e.g., $a \in \{a\}$ but $b \notin \{a\}$ if $a \neq b$. So "being elements of the same subsets of $|\mathfrak{M}|$" is a relation that holds of $a$ and $b$ iff $a = b$. It is a relation that can be expressed in second-order logic, since we can quantify over all subsets of $|\mathfrak{M}|$. Hence, the following formula defines $\mathrm{Id}_{|\mathfrak{M}|}$:

$$\forall X\,(X(x) \leftrightarrow X(y))$$

**Example 35.14.** If $R$ is a two-place predicate symbol, $R^{\mathfrak{M}}$ is a two-place relation on $|\mathfrak{M}|$. Perhaps somewhat confusingly, we'll use $R$ as the predicate symbol for $R$ and for the relation $R^{\mathfrak{M}}$ itself. The *transitive closure* $R^*$ of $R$ is the relation that holds between $a$ and $b$ iff for some $c_1, \ldots, c_k$, $R(a, c_1)$, $R(c_1, c_2)$, $\ldots$, $R(c_k, b)$ holds. This includes the case if $k = 0$, i.e., if $R(a, b)$ holds, so does $R^*(a, b)$. This means that $R \subseteq R^*$. In fact, $R^*$ is the smallest relation that includes $R$ and that is transitive. We can say in second-order logic that $X$ is a transitive relation that includes $R$:

$$\psi_R(X) \equiv \forall x\,\forall y\,(R(x,y) \to X(x,y)) \wedge \forall x\,\forall y\,\forall z\,((X(x,y) \wedge X(y,z)) \to X(x,z))$$

The first conjunct says that $R \subseteq X$ and the second that $X$ is transitive.

To say that $X$ is the smallest such relation is to say that it is itself included in every relation that includes $R$ and is transitive. So we can define the transitive closure of $R$ by the formula

$$R^*(X) \equiv \psi_R(X) \wedge \forall Y\,(\psi_R(Y) \to \forall x\,\forall y\,(X(x,y) \to Y(x,y)))$$

We have $\mathfrak{M}, s \vDash R^*(X)$ iff $s(X) = R^*$. The transitive closure of $R$ cannot be expressed in first-order logic.

## 35.6 Describing Infinite and Enumerable Domains

A set $M$ is (Dedekind) infinite iff there is an injective function $f\colon M \to M$ which is not surjective, i.e., with $\mathrm{dom}(f) \neq M$. In first-order logic, we can consider a one-place function symbol $f$ and say that the function $f^{\mathfrak{M}}$ assigned to it in a structure $\mathfrak{M}$ is injective and $\mathrm{ran}(f) \neq |\mathfrak{M}|$:

$$\forall x\,\forall y\,(f(x) = f(y) \to x = y) \wedge \exists y\,\forall x\,y \neq f(x)$$

If $\mathfrak{M}$ satisfies this sentence, $f^{\mathfrak{M}} : |\mathfrak{M}| \to |\mathfrak{M}|$ is injective, and so $|\mathfrak{M}|$ must be infinite. If $|\mathfrak{M}|$ is infinite, and hence such a function exists, we can let $f^{\mathfrak{M}}$

be that function and $\mathfrak{M}$ will satisfy the sentence. However, this requires that our language contains the non-logical symbol $f$ we use for this purpose. In second-order logic, we can simply say that such a function *exists*. This no-longer requires $f$, and we obtain the sentence in pure second-order logic

$$\text{Inf} \equiv \exists u\, (\forall x\, \forall y\, (u(x) = u(y) \to x = y) \wedge \exists y\, \forall x\, y \neq u(x)).$$

$\mathfrak{M} \vDash \text{Inf}$ iff $|\mathfrak{M}|$ is infinite. We can then define $\text{Fin} \equiv \neg\text{Inf}$; $\mathfrak{M} \vDash \text{Fin}$ iff $|\mathfrak{M}|$ is finite. No single sentence of pure first-order logic can express that the domain is infinite although an infinite set of them can. There is no set of sentences of pure first-order logic that is satisfied in a structure iff its domain is finite.

**Proposition 35.15.** $\mathfrak{M} \vDash \text{Inf}$ *iff* $|\mathfrak{M}|$ *is infinite.*

*Proof.* $\mathfrak{M} \vDash \text{Inf}$ iff $\mathfrak{M}, s \vDash \forall x\, \forall y\, (u(x) = u(y) \to x = y) \wedge \exists y\, \forall x\, y \neq u(x)$ for some $s$. If it does, $s(u)$ is an injective function, and some $y \in |\mathfrak{M}|$ is not in the domain of $s(u)$. Conversely, if there is an injective $f\colon |\mathfrak{M}| \to |\mathfrak{M}|$ with $\text{dom}(f) \neq |\mathfrak{M}|$, then $s(u) = f$ is such a variable assignment. □

A set $M$ is enumerable if there is an enumeration

$$m_0, m_1, m_2, \ldots$$

of its elements (without repetitions). Such an enumeration exists iff there is an element $z \in M$ and a function $f\colon M \to M$ such that $z$, $f(z)$, $f(f(z))$ are all the elements of $M$. For if the enumeration exists, $z = m_0$ and $f(m_k) = m_{k+1}$ (or $f(m_k) = m_k$ if $m_k$ is the last element of the enumeration) are the requisite element and function. On the other hand, if such a $z$ and $f$ exist, then $z$, $f(z)$, $f(f(z))$, ..., is an enumeration of $M$, and $M$ is enumerable. We can express the existence of $z$ and $f$ in second-order logic to produce a sentence true in a structure iff the structure is enumerable:

$$\text{Count} \equiv \exists z\, \exists u\, \forall X\, ((X(z) \wedge \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

**Proposition 35.16.** $\mathfrak{M} \vDash \text{Count}$ *iff* $|\mathfrak{M}|$ *is enumerable.*

*Proof.* Suppose $|\mathfrak{M}|$ is enumerable, and let $m_0$, $m_1$, ..., be an enumeration. By removing repetions we can guarantee that no $m_k$ appears twice. Define $f(m_k) = m_{k+1}$ and let $s(z) = m_0$ and $s(u) = f$. We show that

$$\mathfrak{M}, s \vDash \forall X\, ((X(z) \wedge \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

Suppose $s' \sim_X s$ is arbitrary, and let $M = s'(X)$. Suppose further that $\mathfrak{M}, s' \vDash (X(z) \wedge \forall x\, (X(x) \to X(u(x))))$. Then $s'(z) \in M$ and whenever $x \in M$, also $s'(u)(x) \in M$. In other words, since $s' \sim_X s$, $m_0 \in M$ and if $x \in M$ then $f(x) \in M$, so $m_0 \in M$, $m_1 = f(m_0) \in M$, $m_2 = f(f(m_0)) \in M$, etc. Thus,

$M = |\mathfrak{M}|$, and so $\mathfrak{M} \vDash \forall x\, X(x) s'$. Since $s'$ was an arbitrary $X$-variant of $s$, we are done: $\mathfrak{M} \vDash \text{Count}$.

Now assume that $\mathfrak{M} \vDash \text{Count}$, i.e.,

$$\mathfrak{M}, s \vDash \forall X\, ((X(z) \wedge \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

for some $s$. Let $m = s(z)$ and $f = s(u)$ and consider $M = \{m, f(m), f(f(m)), \dots\}$. Let $s'$ be the $X$-variant of $s$ with $s(X) = M$. Then

$$\mathfrak{M}, s' \vDash (X(z) \wedge \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x)$$

by assumption. Also, $\mathfrak{M}, s' \vDash X(z)$ since $s'(X) = M \ni m = s'(z)$, and also $\mathfrak{M}, s' \vDash \forall x\, (X(x) \to X(u(x)))$ since whenever $x \in M$ also $f(x) \in M$. So, since both antecedent and conditional are satisfied, the consequent must also be: $\mathfrak{M}, s' \vDash \forall x\, X(x)$. But that means that $M = |\mathfrak{M}|$, and so $|\mathfrak{M}|$ is enumerable since $M$ is, by definition. □

## Problems

**Problem 35.1.** Show that $\forall X\, (X(x) \to X(y))$ (note: $\to$ not $\leftrightarrow$!) defines $\text{Id}_{|\mathfrak{M}|}$.

**Problem 35.2.** The sentence $\text{Inf} \wedge \text{Count}$ is true in all and only denumerable domains. Adjust the definition of Count so that it becomes a different sentence that directly expresses that the domain is denumerable, and prove that it does.

# Chapter 36

# Metatheory of Second-order Logic

## 36.1 Introduction

First-order logic has a number of nice properties. We know it is not decidable, but at least it is axiomatizable. That is, there are proof systems for first-order logic which are sound and complete, i.e., they give rise to a derivability relation $\vdash$ with the property that for any set of sentences $\Gamma$ and sentence $Q$, $\Gamma \vDash \varphi$ iff $\Gamma \vdash \varphi$. This means in particular that the validities of first-order logic are computably enumerable. There is a computable function $f : \mathbb{N} \to \mathrm{Sent}(\mathcal{L})$ such that the values of $f$ are all and only the valid sentences of $\mathcal{L}$. This is so because derivations can be enumerated, and those that derive a single sentence are then mapped to that sentence. Second-order logic is more expressive than first-order logic, and so it is in general more complicated to capture its validities. In fact, we'll show that second-order logic is not only undecidable, but its validities are not even computably enumerable. This means there can be no sound and complete proof system for second-order logic (although sound, but incomplete proof systems are available and in fact are important objects of research).

First-order logic also has two more properties: it is compact (if every finite subset of a set $\Gamma$ of sentences is satisfiable, $\Gamma$ itself is satisfiable) and the Löwenheim-Skolem Theorem holds for it (if $\Gamma$ has an infinite model it has a denumerable model). Both of these results fail for second-order logic. Again, the reason is that second-order logic can express facts about the size of domains that first-order logic cannot.

## 36.2 Second-order Arithmetic

Recall that the theory **PA** of Peano arithmetic includes the eight axioms of **Q**,

$$\begin{aligned}
&\forall x\, x' \neq \mathsf{o}\\
&\forall x\, \forall y\, (x' = y' \to x = y)\\
&\forall x\, \forall y\, (x < y \leftrightarrow \exists z\, (x + z') = y)\\
&\forall x\, (x + \mathsf{o}) = x\\
&\forall x\, \forall y\, (x + y') = (x + y)'\\
&\forall x\, (x \times \mathsf{o}) = \mathsf{o}\\
&\forall x\, \forall y\, (x \times y') = ((x \times y) + x)
\end{aligned}$$

plus all sentences of the form

$$(\varphi(\mathsf{o}) \wedge \forall x\, (\varphi(x) \to \varphi(x'))) \to \forall x\, \varphi(x)$$

The latter is a "schema," i.e., a pattern that generates infinitely many sentences of the language of arithmetic, one for each formula $\varphi(x)$. We call this schema the (first-order) *axiom schema of induction*. In *second-order* Peano arithmetic $\mathbf{PA^2}$, induction can be stated as a single sentence. $\mathbf{PA^2}$ consists of the first eight axioms above plus the (second-order) *induction axiom*:

$$\forall X\, (X(\mathsf{o}) \wedge \forall x\, (X(x) \to X(x'))) \to \forall x\, X(x))$$

It says that if a subset $X$ of the domain contains $\mathsf{o}^{\mathfrak{M}}$ and with any $x \in |\mathfrak{M}|$ also contains $\prime^{\mathfrak{M}}(x)$ (i.e., it is "closed under successor") it contains everything in the domain (i.e., $X = |\mathfrak{M}|$).

The induction axiom guarantees that any structure satisfying it contains only those elements of $|\mathfrak{M}|$ the axioms require to be there, i.e., the values of $\overline{n}$ for $n \in \mathbb{N}$. A model of $\mathbf{PA^2}$ contains no non-standard numbers.

**Theorem 36.1.** *If* $\mathfrak{M} \vDash \mathbf{PA^2}$ *then* $|\mathfrak{M}| = \{\mathrm{Val}^n(M) : n \in \mathbb{N}\}$.

*Proof.* Let $N = \{\mathrm{Val}^{\mathfrak{M}}(\overline{n}) : n \in \mathbb{N}\}$, and suppose $\mathfrak{M} \vDash \mathbf{PA^2}$. Of course, for any $n \in \mathbb{N}$, $\mathrm{Val}^{\mathfrak{M}}(\overline{n}) \in |\mathfrak{M}|$, so $N \subseteq |\mathfrak{M}|$.

Now for inclusion in the other direction. Consider a variable assignment $s$ with $s(X) = N$. By assumption,

$$\begin{aligned}
&\mathfrak{M} \vDash \forall X\, (X(\mathsf{o}) \wedge \forall x\, (X(x) \to X(x'))) \to \forall x\, X(x), \text{ thus}\\
&\mathfrak{M}, s \vDash (X(\mathsf{o}) \wedge \forall x\, (X(x) \to X(x'))) \to \forall x\, X(x).
\end{aligned}$$

Consider the antecedent of this conditional. $\mathrm{Val}^{\mathfrak{M}}(\mathsf{o}) \in N$, and so $\mathfrak{M}, s \vDash X(\mathsf{o})$. The second conjunct, $\forall x\, (X(x) \to X(x'))$ is also satisfied. For suppose $x \in N$. By definition of $N$, $x = \mathrm{Val}^{\mathfrak{M}}(\overline{n})$ for some $n$. That gives $\prime^{\mathfrak{M}}(x) = \mathrm{Val}^{\mathfrak{M}}(\overline{n+1}) \in N$. So, $\prime^{\mathfrak{M}}(x) \in N$.

We have that $\mathfrak{M}, s \vDash X(\mathsf{o}) \wedge \forall x\,(X(x) \to X(x'))$. Consequently, $\mathfrak{M}, s \vDash \forall x\, X(x)$. But that means that for every $x \in |\mathfrak{M}|$ we have $x \in s(X) = N$. So, $|\mathfrak{M}| \subseteq N$. □

**Corollary 36.2.** *Any two models of* $\mathbf{PA^2}$ *are isomorphic.*

*Proof.* By Theorem 36.1, the domain of any model of $\mathbf{PA^2}$ is exhausted by $\mathrm{Val}^{\mathfrak{M}}(\overline{n})$. Any such model is also a model of $\mathbf{Q}$. By Proposition 22.3, any such model is standard, i.e., isomorphic to $\mathfrak{N}$. □

Above we defined $\mathbf{PA^2}$ as the theory that contains the first eight arithmetical axioms plus the second-order induction axiom. In fact, thanks to the expressive power of second-order logic, only the *first two* of the arithmetical axioms plus induction are needed for second-order Peano arithmetic.

**Proposition 36.3.** *Let* $\mathbf{PA^{2\dagger}}$ *be the second-order theory containing the first two arithmetical axioms (the successor axioms) and the second-order induction axiom. Then* $\leq$, $+$, *and* $\times$ *are definable in* $\mathbf{PA^{2\dagger}}$.

*Proof.* To show that $\leq$ is definable, we have to find a formula $\varphi_{\leq}(x, y)$ such that $\mathfrak{N} \vDash \varphi(\overline{n}, \overline{m})$ iff $n < m$. Consider the formula

$$\psi(x, Y) \equiv Y(x) \wedge \forall y\,(Y(y) \to Y(y'))$$

Clearly, $\psi(\overline{n}, Y)$ is satisfied by a set $Y \subseteq \mathbb{N}$ iff $\{m : n \leq m\} \subseteq Y$, so we can take $\varphi_{\leq}(x, y) \equiv \forall Y\,(\psi(x, Y) \to Y(y))$. □

**Corollary 36.4.** $\mathfrak{M} \vDash \mathbf{PA^2}$ *iff* $\mathfrak{M} \vDash \mathbf{PA^{2\dagger}}$.

*Proof.* Immediate from Proposition 36.3. □

## 36.3 Second-order Logic is not Axiomatizable

**Theorem 36.5.** *Second-order logic is undecidable.*

*Proof.* A first-order sentence is valid in first-order logic iff it is valid in second-order logic, and first-order logic is undecidable. □

**Theorem 36.6.** *There is no sound and complete proof system for second-order logic.*

*Proof.* Let $\varphi$ be a sentence in the language of arihmetic. $\mathfrak{N} \vDash \varphi$ iff $\mathbf{PA^2} \vDash \varphi$. Let $P$ be the conjunction of the nine axioms of $\mathbf{PA^2}$. $\mathbf{PA^2} \vDash \varphi$ iff $\vDash P \to \varphi$, i.e., $\mathfrak{M} \vDash P \to \varphi$ . Now consider the sentence $\forall z\, \forall u\, \forall u'\, \forall u''\, \forall L\,(P' \to \varphi')$ resulting by replacing $\mathsf{o}$ by $z$, $'$ by the one-place function variable $u$, $+$ and $\times$ by the two-place function-variables $u'$ and $u''$, respectively, and $<$ by the two-place relation variable $L$ and universally quantifying. It is a valid sentence of pure second-order logic iff the original sentence was valid iff $\mathbf{PA^2} \vDash \varphi$ iff $\mathfrak{N} \vDash \varphi$.

Thus if there were a sound and complete proof system for second-order logic, we could use it to define a computable enumeration $f\colon \mathbb{N} \to \mathrm{Sent}(\mathcal{L}_A)$ of the sentences true in $\mathfrak{N}$. This function would be representable in **Q** by some first-order formula $\psi_f(x, y)$. Then the formula $\exists x\, \psi_f(x, y)$ would define the set of true first-order sentences of $\mathfrak{N}$, contradicting Tarski's Theorem. □

## 36.4 Second-order Logic is not Compact

Call a set of sentences $\Gamma$ *finitely satisfiable* if every one of its finite subsets is satisfiable. First-order logic has the property that if a set of sentences $\Gamma$ is finitely satisfiable, it is satisfiable. This property is called *compactness*. It has an equivalent version involving entailment: if $\Gamma \models \varphi$, then already $\Gamma_0 \models \varphi$ for some finite subset $\Gamma_0 \subseteq \Gamma$. In this version it is an immediate corollary of the completeness theorem: for if $\Gamma \models \varphi$, by completeness $\Gamma \vdash \varphi$. But a derivation can only make use of finitely many sentences of $\Gamma$.

Compactness is not true for second-order logic. There are sets of second-order sentences that are finitely satisfiable but not satisfiable, and that entail some $\varphi$ without a finite subset entailing $\varphi$.

**Theorem 36.7.** *Second-order logic is not compact.*

*Proof.* Recall that

$$\mathrm{Inf} \equiv \exists u\, \forall x\, \forall y\, (u(x) = u(y) \to x = y)$$

is satisfied in a structure iff its domain is infinite. Let $\varphi^{\geq n}$ be a sentence that asserts that the domain has at least $n$ elements, e.g.,

$$\varphi^{\geq n} \equiv \exists x_1 \ldots \exists x_n\, (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge \cdots \wedge x_{n-1} \neq x_n)$$

Consider

$$\Gamma = \{\neg\mathrm{Inf}, \varphi^{\geq 1}, \varphi^{\geq 2}, \varphi^{\geq 3}, \ldots\}$$

It is finitely satisfiable, since for any finite subset $\Gamma_0$ there is some $k$ so that $\varphi^{\geq k} \in \Gamma$ but no $\varphi^{\geq n} \in \Gamma$ for $n > k$. If $|\mathfrak{M}|$ has $k$ elements, $\mathfrak{M} \models \Gamma_0$. But, $\Gamma$ is not satisfiable: if $\mathfrak{M} \models \neg\mathrm{Inf}$, $|\mathfrak{M}|$ must be finite, say, of size $k$. Then $\mathfrak{M} \not\models \varphi^{\geq k+1}$. □

## 36.5 The Löwenheim-Skolem Theorem Fails for Second-order Logic

The (Downward) Löwenheim-Skolem Theorem states that every set of sentences with an infinite model has an enumerable model. It, too, is a consequence of the completeness theorem: the proof of completeness generates a model for any consistent set of sentences, and that model is enumerable. There is also an Upward Löwenheim-Skolem Theorem, which guarantees that if a set of sentences has a denumerable model it also has a non-enumerable model. Both theorems fail in second-order logic.

**Theorem 36.8.** *The Löwenheim-Skolem Theorem fails for second-order logic: There are sentences with infinite models but no enumerable models.*

*Proof.* Recall that

$$\text{Count} \equiv \exists z\, \exists u\, \forall X\, ((X(z) \wedge \forall x\, (X(x) \to X(u(x)))) \to \forall x\, X(x))$$

is true in a structure $\mathfrak{M}$ iff $|\mathfrak{M}|$ is enumerable. So $\text{Inf} \wedge \neg\text{Count}$ is true in $\mathfrak{M}$ iff $|\mathfrak{M}|$ is both infinite and not enumerable. There are such structures—take any non-enumerable set as the domain, e.g., $\wp(\mathbb{N})$ or $\mathbb{R}$. So $\text{Inf} \wedge \neg\text{Count}$ has infinite models but no enumerable models. □

**Theorem 36.9.** *There are sentences with denumerable but no non-enumerable models.*

*Proof.* $\text{Count} \wedge \text{Inf}$ is true in $\mathbb{N}$ but not in any structure $\mathfrak{M}$ with $|\mathfrak{M}|$ non-enumerable. □

## Problems

**Problem 36.1.** Complete the proof of Proposition 36.3.

**Problem 36.2.** Give an example of a set $\Gamma$ and a sentence $\varphi$ so that $\Gamma \models \varphi$ but for every finite subset $\Gamma_0 \subseteq \Gamma$, $\Gamma_0 \not\models \varphi$.

# Chapter 37

# Second-order Logic and Set Theory

This section deals with coding powersets and the continuum in second-order logic. The results are stated but proofs have yet to be filled in. There are no problems yet—and the definitions and results themselves may have problems. Use with caution and report anything that's false or unclear.

## 37.1 Introduction

Since second-order logic can quantify over subsets of the domain as well as functions, it is to be expected that some amount, at least, of set theory can be carried out in second-order logic. By "carry out," we mean that it is possible to express set theoretic properties and statements in second-order logic, and is possible without any special, non-logical vocabulary for sets (e.g., the membership predicate symbol of set theory). For instance, we can define unions and intersections of sets and the subset relationship, but also compare the sizes of sets, and state results such as Cantor's Theorem.

## 37.2 Comparing Sets

**Proposition 37.1.** *The formula* $\forall x\,(X(x) \to Y(x))$ *defines the subset relation, i.e.,* $\mathfrak{M}, s \vDash \forall x\,(X(x) \to Y(x))$ *iff* $s(X) \subseteq S(y)$.

**Proposition 37.2.** *The formula* $\forall x\,(X(x) \leftrightarrow Y(x))$ *defines the identity relation on sets, i.e.,* $\mathfrak{M}, s \vDash \forall x\,(X(x) \leftrightarrow Y(x))$ *iff* $s(X) = S(y)$.

**Proposition 37.3.** *The formula* $\exists x\, X(x)$ *defines the property of being non-empty, i.e.,* $\mathfrak{M}, s \vDash \exists x\, X(x)$ *iff* $s(X) \neq \emptyset$.

A set $X$ is no larger than a set $Y$, $X \preceq Y$, iff there is an injective function $f\colon X \to Y$. Since we can express that a function is injective, and also that its values for arguments in $X$ are in $Y$, we can also define the relation of being no larger than on subsets of the domain.

**Proposition 37.4.** *The formula*

$$\exists u\,(\forall x\,(X(x) \to Y(u(x))) \wedge \forall x\,\forall y\,(u(x) = u(y) \to x = y))$$

*defines the relation of being no larger than.*

Two sets are the same size, or "equinumerous," $X \approx Y$, iff there is a bijective function $f\colon X \to Y$.

**Proposition 37.5.** *The formula*

$$\begin{aligned}\exists u\,(\forall x\,(X(x) \to Y(u(x))) \wedge \\ \forall x\,\forall y\,(u(x) = u(y) \to x = y) \wedge \\ \forall y\,(Y(y) \to \exists x\,(X(x) \wedge y = u(x))))\end{aligned}$$

*defines the relation of being equinumerous with.*

We will abbreviate these formulas, respectively, as $X \subseteq Y$, $X = Y$, $X \neq \emptyset$, $X \prec Y$, and $X \approx Y$. (This may be slightly confusing, since we use the same notation when we speak informally about sets $X$ and $Y$—but here the notation is an abbreviation for formulas in second-order logic involving one-place relation variables $X$ and $Y$.)

**Proposition 37.6.** *The sentence* $\forall X\,\forall Y\,((X \preceq Y \wedge Y \preceq X) \to X \approx Y)$ *is valid.*

*Proof.* The sentence is satisfied in a structure $\mathfrak{M}$ if, for any subsets $X \subseteq |\mathfrak{M}|$ and $Y \subseteq |\mathfrak{M}|$, if $X \preceq Y$ and $Y \preceq X$ then $X \approx Y$. But this holds for *any* sets $X$ and $Y$—it is the Schröder-Bernstein Theorem. □

## 37.3 Cardinalities of Sets

Just as we can express that the domain is finite or infinite, enumerable or non-enumerable, we can define the property of a subset of $|\mathfrak{M}|$ being finite or infinite, enumerable or non-enumerable.

**Proposition 37.7.** *The formula* $\mathrm{Inf}(X) \equiv$

$$\begin{aligned}\exists u\,(\forall x\,\forall y\,(u(x) = u(y) \to x = y) \wedge \\ \exists y\,(X(y) \wedge \forall x\,(X(x) \to y \neq u(x)))\end{aligned}$$

*is satisfied with respect to a variable assignment $s$ iff $s(X)$ is infinite.*

**Proposition 37.8.** *The formula* $\mathrm{Count}(X) \equiv$

$$\exists z\, \exists u\, (X(z) \wedge \forall x\, (X(x) \to X(u(x))) \wedge \forall Y\, ((Y(z) \wedge \forall x\, (Y(x) \to Y(u(x)))) \to X = Y))$$

*is satisfied with respect to a variable assignment $s$ iff $s(X)$ is enumerable*

We know from Cantor's Theorem that there are non-enumerable sets, and in fact, that there are infinitely many different levels of infinite sizes. Set theory develops an entire arithmetic of sizes of sets, and assigns infinite cardinal numbers to sets. The natural numbers serve as the cardinal numbers measuring the sizes of finite sets. The cardinality of denumerable sets is the first infinite cardinality, called $\aleph_0$ ("aleph-nought" or "aleph-zero"). The next infinite size is $\aleph_1$. It is the smallest size a set can be without being countable (i.e., of size $\aleph_0$). We can define "$X$ has size $\aleph_0$" as $\mathrm{Aleph}_0(X) \leftrightarrow \mathrm{Inf}(X) \wedge \mathrm{Count}(X)$. $X$ has size $\aleph_1$ iff all its subsets are finite or have size $\aleph_0$, but is not itself of size $\aleph_0$. Hence we can express this by the formula $\mathrm{Aleph}_1(X) \equiv \forall Y\, (Y \subseteq X \to (\neg\mathrm{Inf}(Y) \vee \mathrm{Aleph}_0(Y))) \wedge \neg\mathrm{Aleph}_0(X)$. Being of size $\aleph_2$ is defined similarly, etc.

There is one size of special interest, the so-called cardinality of the continuum. It is the size of $\wp(\mathbb{N})$, or, equivalently, the size of $\mathbb{R}$. That a set is the size of the continuum can also be expressed in second-order logic, but requires a bit more work.

## 37.4 The Power of the Continuum

In second-order logic we can quantify over subsets of the domain, but not over sets of subsets of the domain. To do this directly, we would need *third-order* logic. For instance, if we wanted to state Cantor's Theorem that there is no injective function from the power set of a set to the set itself, we might try to formulate it as "for every set $X$, and every set $P$, if $P$ is the power set of $X$, then not $P \preceq X$. And to say that $P$ is the power set of $X$ would require formalizing that the elements of $P$ are all and only the subsets of $X$, so something like $\forall Y\, (P(Y) \leftrightarrow Y \subseteq X)$. The problem lies in $P(Y)$: that is not a formula of second-order logic, since only terms can be arguments to one-place relation variables like $P$.

We can, however, *simulate* quantification over sets of sets, if the domain is large enough. The idea is to make use of the fact that two-place relations $R$ relates elements of the domain to elements of the domain. Given such an $R$, we can collect all the elements to which some $x$ is $R$-related: $\{y \in |\mathfrak{M}| : R(x, y)\}$ is the set "coded by" $x$. Converseley, if $Z \subseteq \wp(|\mathfrak{M}|)$ is some collection of subsets of $|\mathfrak{M}|$, and there are at least as many elements of $|\mathfrak{M}|$ as there are sets in $Z$, then there is also a relation $R \subseteq |\mathfrak{M}|^2$ such that every $Y \in Z$ is coded by some $x$ using $R$.

**Definition 37.9.** If $R \subseteq |\mathfrak{M}|^2$, then $x$ *R-codes* $\{y \in |\mathfrak{M}| : R(x,y)\}$. $Y$ *R*-codes $\wp(X)$ iff for every $Z \subseteq X$, some $x \in Y$ *R*-codes $Y$, and every $x \in Y$ *R*-codes some $Y \in Z$.

**Proposition 37.10.** *The formula*

$$\mathrm{Codes}(x, R, Y) \equiv \forall y\, (Y(y) \leftrightarrow R(x,y))$$

*expresses that $s(x)$ $s(R)$-codes $s(Y)$. The formula*

$$\begin{aligned}\mathrm{Pow}(Y, R, X) \equiv \\ \forall Z\, (Z \subseteq X \to \exists x\, (Y(x) \wedge \mathrm{Codes}(x, R, Z))) \wedge \\ \forall x\, (Y(x) \to \forall Z\, (\mathrm{Codes}(x, R, Z) \to Z \subseteq X)\end{aligned}$$

*expresses that $s(Y)$ $s(R)$-codes the power set of $s(X)$.*

With this trick, we can express statements about the power set by quantifying over the codes of subsets rather than the subsets themselves. For instance, Cantor's Theorem can now be expressed by saying that there is no injective function from the domain of any relation that codes the power set of $X$ to $X$ itself.

**Proposition 37.11.** *The sentence*

$$\begin{aligned}\forall X\, \forall R\, (\mathrm{Pow}(R, X) \to \\ \neg\exists u\, (\forall x\, \forall y\, (u(x) = u(y) \to x = y) \wedge \\ \forall Y\, (\mathrm{Codes}(x, R, Y) \to X(u(x)))))\end{aligned}$$

*is valid.*

The power set of a denumerable set is non-enumerable, and so its cardinality is larger than that of any denumerable set (which is $\aleph_0$). The size of $\wp(\mathbb{R})$ is called the "power of the continuum," since it is the same size as the points on the real number line, $\mathbb{R}$. If the domain is large enough to code the power set of a denumerable set, we can express that a set is the size of the continuum by saying that it is equinumerous with any set $Y$ that codes the power set of set $X$ of size $\aleph_0$. (If the domain is not large enough, i.e., it contains no subset equinumerous with $\mathbb{R}$, then there can also be no relation that codes $\wp(X)$.)

**Proposition 37.12.** *If $\mathbb{R} \preceq |\mathfrak{M}|$, then the formula*

$$\mathrm{Cont}(X) \equiv \forall X\, \forall Y\, \forall R\, ((\mathrm{Aleph}_0(X) \wedge \mathrm{Pow}(Y, R, X)) \to \neg Y \preceq X)$$

*expresses that $s(X) \approx \mathbb{R}$.*

**Proposition 37.13.** $|\mathfrak{M}| \approx \mathbb{R}$ *iff*

$$\begin{aligned}\mathfrak{M} \vDash \exists X\, \exists Y\, \exists R\, (&\mathrm{Aleph}_0(X) \wedge \mathrm{Pow}(Y, R, X) \wedge \\ &\exists u\, (\forall x\, \forall y\, (u(x) = u(y) \to x = y) \wedge \\ &\qquad \forall y\, (Y(y) \to \exists x\, y = u(x))))\end{aligned}$$

The Continuum Hypothesis is the statement that the size of the continuum is the first non-enumerable cardinality, i.e, that $\wp(\mathbb{N})$ has size $\aleph_1$.

**Proposition 37.14.** *The Continuum Hypothesis is true iff*

$$\mathrm{CH} \equiv \forall X\, (\mathrm{Aleph}_1(X) \leftrightarrow \mathrm{Cont}(x))$$

*is valid.*

Note that it isn't true that $\neg$CH is valid iff the Continuum Hypothesis is false. In an enumerable domain, there are no subsets of size $\aleph_1$ and also no subsets of the size of the continuum, so CH is always true in an enumerable domain. However, we can give a different sentence that is valid iff the Continuum Hypothesis is false:

**Proposition 37.15.** *The Continuum Hypothesis is false iff*

$$\mathrm{NCH} \equiv \forall X\, (\mathrm{Cont}(X) \to \exists Y\, (Y \subseteq X \wedge \neg\mathrm{Count}(X) \wedge \neg X \approx Y))$$

*is valid.*

# Part IX

# Normal Modal Logics

This part covers the metatheory of normal modal logics. It currently consists of Aldo Antonelli's notes on classical correspondence theory for basic modal logic.

# Chapter 38

# Syntax and Semantics of Normal Modal Logics

## 38.1 Introduction

Modal Logic deals with *modal propositions* and the entailment relations among them. Examples of modal propositions are the following:

1. It is necessary that $2 + 2 = 4$.

2. It is necessarily possible that it will rain tomorrow.

3. If it is necessarily possible that $\varphi$ then it is possible that $\varphi$.

Possibility and necessity are not the only modalities: other unary connectives are also classified as modalities, for instance, "it ought to be the case that $\varphi$," "It will be the case that $\varphi$," "Dana knows that $\varphi$," or "Dana believes that $\varphi$."

Modal logic makes its first appearance in Aristotle's *De Interpretatione*: he was the first to notice that necessity implies possibility, but not vice versa; that possibility and necessity are inter-definable; that If $\varphi \wedge \psi$ is possibly true then $\varphi$ is possibly true and $\psi$ is possibly true, but not conversely; and that if $\varphi \to \psi$ is necessary, then if $\varphi$ is necessary, so is $\psi$.

The first modern approach to modal logic was the work of C. I. Lewis, culminating with Lewis and Langford, *Symbolic Logic* (1932). Lewis & Langford were unhappy with the representation of implication by means of the material conditional: $\varphi \to \psi$ is a poor substitute for "$\varphi$ implies $\psi$." Instead, they proposed to characterize implication as "Necessarily, if $\varphi$ then $\psi$," symbolized as $\varphi \dashv\vdash \psi$. In trying to sort out the different properties, Lewis indentified five different modal systems, **S1**, ..., **S4**, **S5**, the last two of which are still in use.

The approach of Lewis and Langford was purely *syntactical*: they identified reasonable axioms and rules and investigated what was provable with those means. A semantic approach remained elusive for a long time, until a

first attempt was made by Rudolf Carnap in *Meaning and Necessity* (1947) using the notion of a *state description*, i.e., a collection of atomic sentences (those that are "true" in that state description). After lifting the truth definition to arbitrary sentences $\varphi$, Carnap defines $\varphi$ to be *necessarily true* if it is true in all state descriptions. Carnap's approach could not handle *iterated* modalities, in that sentences of the form "Possibly necessarily ... possibly $\varphi$" always reduce to the innermost modality.

The major breakthrough in modal semantics came with Saul Kripke's article "A Completeness Theorem in Modal Logic" (JSL 1959). Kripke based his work on Leibniz's idea that a statement is necessarily true if it is true "at all possible worlds." This idea, though, suffers from the same drawbacks as Carnap's, in that the truth of statement at a world $w$ (or a state description $s$) does not depend on $w$ at all. So Kripke assumed that worlds are related by an *accessibility relation* $R$, and that a statement of the form "Necessarily $\varphi$" is true at a world $w$ if and only if $\varphi$ is true at all worlds $w'$ *accessible from* $w$. Semantics that provide some version of this approach are called Kripke semantics and made possible the tumultuous development of modal logics (in the plural).

When interpreted by the Kripke semantics, modal logic shows us what *relational structures* look like "from the inside." A relational structure is just a set equipped with a binary relation (for instance, the set of students in the class ordered by their social security number is a relational structure). But in fact relational structures come in all sorts of domains: besides relative possibility of states of the world, we can have epistemic states of some agent related by epistemic possibility, or states of a dynamical system with their state transitions, etc. Modal logic can be used to model all of these: the first give us ordinary, alethic, modal logic; the others give us epistemic logic, dynamic logic, etc.

We focus on one particular angle, known to modal logicians as "correspondence theory." One of the most significant early discoveries of Kripke's is that many properties of the accessibility relation $R$ (whether it is transitive, symmetric, etc.) can be characterized *in the modal language* itself by means of appropriate "modal schemas." Modal logicians say, for instance, that the reflexivity of $R$ "corresponds" to the schema "If necessarily $\varphi$, then $\varphi$". We explore mainly the correspondence theory of a number of classical systems of modal logic (e.g., **S4** and **S5**) obtained by a combination of the schemas D, T, B, 4, and 5.

## 38.2 The Language of Basic Modal Logic

**Definition 38.1.** The basic language of modal logic contains

1. The propositional constant for falsity $\bot$.

2. A denumerable set of propositional variables: $p_0$, $p_1$, $p_2$, ...

3. The propositional connectives: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (conditional).
4. The modal operator $\Box$.
5. The modal operator $\Diamond$.

**Definition 38.2.** *Formulas* of the basic modal language are inductively defined as follows:

1. $\bot$ is an atomic formula.
2. Every propositional variable $p_i$ is an (atomic) formula.
3. If $\varphi$ and $\psi$ are formulas, then $(\varphi \wedge \psi)$ is a formula.
4. If $\varphi$ and $\psi$ are formulas, then $(\varphi \vee \psi)$ is a formula.
5. If $\varphi$ and $\psi$ are formulas, then $(\varphi \rightarrow \psi)$ is a formula.
6. If $\varphi$ is a formula, so is $\Box\varphi$.
7. If $\varphi$ is a formula, then $\Diamond\varphi$ is a formula.
8. Nothing else is a formula.

If a formula $\varphi$ does not contain $\Box$ or $\Diamond$, we say it is *modal-free*.

## 38.3 Simultaneous Substitution

An *instance* of a formula $\varphi$ is the result of replacing all occurrences of a propositional variable in $\varphi$ by some other formula. We will refer to instances of formulas often, both when discussing validity and when discussing derivability. It therefore is useful to define the notion precisely.

**Definition 38.3.** Where $\varphi$ is a modal formula all of whose propositional variables are among $p_1, \ldots, p_n$, and $\theta_1, \ldots, \theta_n$ are also modal formulas, we define $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ as the result of simultaneously substituting each $\theta_i$ for $p_i$ in $A$. Formally, this is a definition by induction on $\varphi$:

1. $\varphi \equiv \bot$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is $\bot$.
2. $\varphi \equiv q$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is $q$, provided $q \not\equiv p_i$ for $i = 1, \ldots, n$.
3. $\varphi \equiv p_i$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is $\theta_i$.
4. $\varphi \equiv \neg\psi$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is $\neg\psi[\theta_1/p_1, \ldots, \theta_n/p_n]$.
5. $\varphi \equiv (\psi \wedge \chi)$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is
$$(\psi[\theta_1/p_1, \ldots, \theta_n/p_n] \wedge \chi[\theta_1/p_1, \ldots, \theta_n/p_n]).$$

6. $\varphi \equiv (\psi \vee \chi)$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is
$$(\psi[\theta_1/p_1, \ldots, \theta_n/p_n] \vee \chi[\theta_1/p_1, \ldots, \theta_n/p_n]).$$
7. $\varphi \equiv (\psi \to \chi)$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is
$$(\psi[\theta_1/p_1, \ldots, \theta_n/p_n] \to \chi[\theta_1/p_1, \ldots, \theta_n/p_n]).$$
8. $\varphi \equiv (\psi \leftrightarrow \chi)$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is
$$(\psi[\theta_1/p_1, \ldots, \theta_n/p_n] \leftrightarrow \chi[\theta_1/p_1, \ldots, \theta_n/p_n]).$$
9. $\varphi \equiv \Box\psi$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is $\Box\psi[\theta_1/p_1, \ldots, \theta_n/p_n]$.
10. $\varphi \equiv \Diamond\psi$: $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is $\Diamond\psi[\theta_1/p_1, \ldots, \theta_n/p_n]$.

The formula $\varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$ is called a *substitution instance* of $\varphi$.

**Example 38.4.** Suppose $\varphi$ is $p_1 \to \Box(p_1 \wedge p_2)$, $\theta_1$ is $\Diamond(p_2 \to p_3)$ and $\theta_2$ is $\neg\Box p_1$. Then $\varphi[\theta_1/p_1, \theta_2/p_2]$ is
$$\Diamond(p_2 \to p_3) \to \Box(\Diamond(p_2 \to p_3) \wedge \neg\Box p_1)$$
while $\varphi[\theta_2/p_1, \theta_1/p_2]$ is
$$\neg\Box p_1 \to \Box(\neg\Box p_1 \wedge \Diamond(p_2 \to p_3))$$

Note that simultaneous substitution is in general not the same as iterated substitution, e.g., compare $\varphi[\theta_1/p_1, \theta_2/p_2]$ above with $(\varphi[\theta_1/p_1])[\theta_2/p_2]$, which is:
$$\Diamond(p_2 \to p_3) \to \Box(\Diamond(p_2 \to p_3) \wedge p_2)[\neg\Box p_1/p_2], \text{ i.e.,}$$
$$\Diamond(\neg\Box p_1 \to p_3) \to \Box(\Diamond(\neg\Box p_1 \to p_3) \wedge \neg\Box p_1)$$
and with $(\varphi[\theta_2/p_2])[\theta_1/p_1]$:
$$p_1 \to \Box(p_1 \wedge \neg\Box p_1)[\Diamond(p_2 \to p_3)/p_1], \text{ i.e.,}$$
$$\Diamond(p_2 \to p_3) \to \Box(\Diamond(p_2 \to p_3) \wedge \neg\Box\Diamond(p_2 \to p_3)).$$

## 38.4 Relational Models

The basic concept of semantics for normal modal logics is that of a *relational model*. It consists of a set of worlds, which are related by a binary "accessibility relation," together with an assignment which determines which propositional variables count as "true" at which worlds.

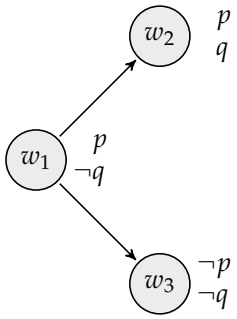



Figure 38.1: A simple model.

**Definition 38.5.** A *model* for the basic modal language is a triple $\mathfrak{M} = \langle W, R, V \rangle$, where

1. $W$ is a nonempty set of "worlds,"
2. $R$ is a binary accessibility relation on $W$, and
3. $V$ is a function assigning to each propositional variable $p$ a set $V(p)$ of possible worlds.

When $Rww'$ holds, we say that $w'$ is *accessible from* $w$. When $w \in V(p)$ we say $p$ is *true at* $w$.

The great advantage of relational semantics is that models can be represented by means of simple diagrams, such as the one in Figure 38.1. Worlds are represented by nodes, and world $w'$ is accessible from $w$ precisely when there is an arrow from $w$ to $w'$. Moreover, we label a node (world) by $p$ when $w \in V(p)$, and otherwise by $\neg p$. Figure 38.1 represents the model with $W = \{w_1, w_2, w_3\}$, $R = \{\langle w_1, w_2 \rangle, \langle w_1, w_3 \rangle\}$, $V(p) = \{w_1, w_2\}$, and $V(q) = \{w_2\}$.

## 38.5 Truth at a World

Every modal model determines which modal formulas count as true at which worlds in it. The relation "model $\mathfrak{M}$ makes formula $\varphi$ true at world $w$" is the basic notion of relational semantics. The relation is defined inductively and coincides with the usual characterization using truth tables for the non-modal operators.

**Definition 38.6.** *Truth of a formula $\varphi$ at $w$* in a $\mathfrak{M}$, in symbols: $\mathfrak{M}, w \Vdash \varphi$, is defined inductively as follows:

1. $\varphi \equiv \bot$: Never $\mathfrak{M}, w \Vdash \bot$.
2. $\mathfrak{M}, w \Vdash p$ iff $w \in V(p)$
3. $\varphi \equiv \neg\psi$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \nVdash \psi$.
4. $\varphi \equiv (\psi \wedge \chi)$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \Vdash \psi$ and $\mathfrak{M}, w \Vdash \chi$.
5. $\varphi \equiv (\psi \vee \chi)$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \Vdash \psi$ or $\mathfrak{M}, w \Vdash \chi$ (or both).
6. $\varphi \equiv (\psi \to \chi)$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \nVdash \psi$ or $\mathfrak{M}, w \Vdash \chi$.
7. $\varphi \equiv \Box\psi$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w' \Vdash \psi$ for all $w' \in W$ with $Rww'$
8. $\varphi \equiv \Diamond\psi$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w' \Vdash \psi$ for at least one $w' \in W$ with $Rww'$

Note that by clause (7), a formula $\Box\psi$ is true at $w$ whenever there are no $w'$ with $wRw'$. In such a case $\Box\psi$ is *vacuously* true at $w$. Also, $\Box\psi$ may be satisfied at $w$ even if $\psi$ is not. The truth of $\psi$ at $w$ does not guarantee the truth of $\Diamond\psi$ at $w$. This holds, however, if $Rww$, e.g., if $R$ is reflexive. If there is no $w'$ such that $Rww'$, then $\mathfrak{M}, w \nVdash \Diamond\varphi$, for any $\varphi$.

**Proposition 38.7.** *1. $\mathfrak{M}, w \Vdash \Box\varphi$ iff $\mathfrak{M}, w \Vdash \neg\Diamond\neg\varphi$.*

*2. $\mathfrak{M}, w \Vdash \Diamond\varphi$ iff $\mathfrak{M}, w \Vdash \neg\Box\neg\varphi$.*

*Proof.* 1. $\mathfrak{M}, w \Vdash \neg\Diamond\neg\varphi$ iff $\mathfrak{M} \nVdash \Diamond\neg\varphi$ by definition of $\mathfrak{M}, w \Vdash$. $\mathfrak{M}, w \Vdash \Diamond\neg\varphi$ iff for some $w'$ with $Rww'$, $\mathfrak{M}, w' \Vdash \neg\varphi$. Hence, $\mathfrak{M}, w \nVdash \Diamond\neg\varphi$ iff for all $w'$ with $Rww'$, $\mathfrak{M}, w' \nVdash \neg\varphi$. We also have $\mathfrak{M}, w' \nVdash \neg\varphi$ iff $\mathfrak{M}, w' \Vdash \varphi$. Together we have $\mathfrak{M}, w \Vdash \neg\Diamond\neg\varphi$ iff for all $w'$ with $Rww'$, $\mathfrak{M}, w' \Vdash \varphi$. Again by definition of $\mathfrak{M}, w \Vdash$, that is the case iff $\mathfrak{M}, w \Vdash \Box\varphi$.

2. Exercise.

□

## 38.6 Truth in a Model

Sometimes we are interested which formulas are true at every world in a given model. Let's introduce a notation for this.

**Definition 38.8.** A formula $\varphi$ is *true in a model* $M = \langle W, R, V\rangle$, written $\mathfrak{M} \Vdash \varphi$, if and only if $\mathfrak{M}, w \Vdash \varphi$ for every $w \in W$.

**Proposition 38.9.** *1. If $\mathfrak{M} \Vdash \varphi$ then $\mathfrak{M} \nVdash \neg\varphi$, but* not *vice-versa.*

*2. If $\mathfrak{M} \Vdash \varphi \to \psi$ then $\mathfrak{M} \Vdash \varphi$ only if $\mathfrak{M} \Vdash \psi$, but* not *vice-versa.*

*Proof.* 1. If $\mathfrak{M} \Vdash \varphi$ then $\varphi$ is true at all worlds in $W$, and since $W \neq \emptyset$, it can't be that $\mathfrak{M} \Vdash \neg\varphi$, or else $\varphi$ would have to be both true and false at some world.

On the other hand, if $\mathfrak{M} \nVdash \neg\varphi$ then $\varphi$ is true at some world $w \in W$. It does not follow that $\mathfrak{M}, w \Vdash \varphi$ for *every* $w \in W$. For instance, in the model of Figure 38.1, $\mathfrak{M} \nVdash \neg p$, and also $\mathfrak{M} \nVdash p$.

2. Assume $\mathfrak{M} \Vdash \varphi \to \psi$ and $\mathfrak{M} \Vdash \varphi$; to show $\mathfrak{M} \Vdash \psi$ let $w \in W$ be an arbitrary world. Then $\mathfrak{M}, w \Vdash \varphi \to \psi$ and $\mathfrak{M}, w \Vdash \psi$, so $\mathfrak{M}, w \Vdash \psi$, and since $w$ was arbitrary, $\mathfrak{M} \Vdash \psi$.

   To show that the converse fails, we need to find a model $\mathfrak{M}$ such that $\mathfrak{M} \Vdash \varphi$ only if $\mathfrak{M} \Vdash \psi$, but $\mathfrak{M} \nVdash \varphi \to \psi$. Consider again the model of Figure 38.1: $\mathfrak{M} \nVdash p$ and hence (vacuously) $\mathfrak{M} \Vdash p$ only if $\mathfrak{M} \Vdash q$. However, $\mathfrak{M} \nVdash p \to q$, as $p$ is true but $q$ false at $w_1$.

□

## 38.7 Validity

Formulas that are true in all models, i.e., true at every world in every model, are particularly interesting. They represent those modal propositions which are true regardless of how $\Box$ and $\Diamond$ are interpreted, as long as the interpretation is "normal" in the sense that it is generated by some accessibility relation on possible worlds. We call such formulas *valid*. For instance, $\Box(p \wedge q) \to \Box p$ is valid. Some formulas one might expect to be valid on the basis of the alethic interpretation of $\Box$, such as $\Box p \to p$, are not valid, however. Part of the interest of relational models is that different interpretations of $\Box$ and $\Diamond$ can be captured by different kinds of accessibility relations. This suggests that we should define validity not just relative to *all* models, but relative to all models *of a certain kind*. It will turn out, e.g., that $\Box p \to p$ is true in all models where every world is accessible from itself, i.e., $R$ is reflexive. Defining validity relative to classes of models enables us to formulate this succinctly: $\Box p \to p$ is valid in the class of reflexive models.

**Definition 38.10.** A formula $\varphi$ is *valid* in a class $\mathcal{C}$ of models if it is true in every model in $\mathcal{C}$ (i.e., true at every world in every model in $\mathcal{C}$). If $\varphi$ is valid in $\mathcal{C}$, we write $\mathcal{C} \vDash \varphi$, and we write $\vDash \varphi$ if $\varphi$ is valid in the class of *all* models.

**Proposition 38.11.** *If $\varphi$ is valid in $\mathcal{C}$ it is also valid in each class $\mathcal{C}' \subseteq \mathcal{C}$.*

**Proposition 38.12.** *If $\varphi$ is valid, then so is $\Box\varphi$.*

*Proof.* Assume $\vDash \varphi$. To show $\vDash \Box\varphi$ let $\mathfrak{M} = \langle W, R, V\rangle$ be a model and $w \in W$. If $Rww'$ then $\mathfrak{M}, w' \Vdash \varphi$, since $\varphi$ is valid, and so also $\mathfrak{M}, w \Vdash \Box\varphi$. Since $\mathfrak{M}$ and $w$ were arbitrary, $\vDash \Box\varphi$. □

## 38.8 Tautological Instances

A modal-free formula is a tautology if it is true under every truth-value assignment. Clearly, every tautology is true at every world in every model. But for formulas involving $\Box$ and $\Diamond$, the notion of tautology is not defined. Is it the case, e.g., that $\Box p \vee \neg\Box p$—an instance of the principle of excluded middle—is valid? The notion of a *tautological instance* helps: a formula that is a substitution instance of a (non-modal) tautology. It is not surprising, but still requires proof, that every tautological instance is valid.

**Definition 38.13.** A modal formula $\psi$ is a *tautological instance* if and only if there is a modal-free tautology $\varphi$ with propositional variables $p_1, \ldots, p_n$ and formulas $\theta_1, \ldots, \theta_n$ such that $\psi \equiv \varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$.

**Lemma 38.14.** *Suppose $\varphi$ is a modal-free formula whose propositional variables are $p_1, \ldots, p_n$, and let $\theta_1, \ldots, \theta_n$ be modal formulas. Then for any assignment $\mathfrak{v}$, any model $\mathfrak{M} = \langle W, R, V\rangle$, and any $w \in W$ such that $\mathfrak{v}(p_i) = \mathbb{T}$ if and only if $\mathfrak{M}, w \Vdash \theta_i$ we have that $\mathfrak{v} \vDash \varphi$ if and only if $\mathfrak{M}, w \Vdash \varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$.*

*Proof.* By induction on $\varphi$.

1. $\varphi \equiv \bot$: Both $\mathfrak{v} \nvDash \bot$ and $\mathfrak{M}, w \nVdash \bot$.

2. $\varphi \equiv p_i$:

$$\begin{aligned}
\mathfrak{v} \vDash p_i &\Leftrightarrow \mathfrak{v}(p_i) = \mathbb{T} && \text{by definition of } \mathfrak{v} \vDash p_i;\\
&\Leftrightarrow \mathfrak{M}, w \Vdash \theta_i && \text{by assumption}\\
&\Leftrightarrow \mathfrak{M}, w \Vdash p_i[\theta_1/p_1, \ldots, \theta_n/p_n] && \text{since } p_i[\theta_1/p_1, \ldots, \theta_n/p_n] \equiv \theta_i.
\end{aligned}$$

3. $\varphi \equiv \neg\psi$:

$$\begin{aligned}
\mathfrak{v} \vDash \neg\psi &\Leftrightarrow \mathfrak{v} \nvDash \psi && \text{by definition of } \mathfrak{v} \vDash;\\
&\Leftrightarrow \mathfrak{M}, w \nVdash \psi[\theta_1/p_1, \ldots, \theta_n/p_n] && \text{by induction hypothesis;}\\
&\Leftrightarrow \mathfrak{M}, w \Vdash \neg\psi[\theta_1/p_1, \ldots, \theta_n/p_n] && \text{by definition of } \mathfrak{v} \vDash.
\end{aligned}$$

4. $\varphi \equiv (\psi \wedge \chi)$:

$$\begin{aligned}
\mathfrak{v} \vDash \psi \wedge \chi &\Leftrightarrow \mathfrak{v} \vDash \psi \text{ and } \mathfrak{v} \vDash \chi && \text{by definition of } \mathfrak{v} \vDash;\\
&\Leftrightarrow \mathfrak{M}, w \Vdash \psi[\theta_1/p_1, \ldots, \theta_n/p_n] \text{ and} &&\\
&\quad \mathfrak{M}, w \Vdash \chi[\theta_1/p_1, \ldots, \theta_n/p_n], && \text{by induction hypothesis;}\\
&\Leftrightarrow \mathfrak{M}, w \Vdash (\psi \wedge \chi)[\theta_1/p_1, \ldots, \theta_n/p_n] && \text{by definition of } \mathfrak{M}, w \Vdash.
\end{aligned}$$

5. $\varphi \equiv (\psi \vee \chi)$:

$$\begin{aligned}
\mathfrak{v} \vDash \psi \vee \chi &\Leftrightarrow \mathfrak{v} \vDash \psi \text{ or } \mathfrak{v} \vDash \chi && \text{by definition of } \mathfrak{v} \vDash;\\
&\Leftrightarrow \mathfrak{M}, w \Vdash \psi[\theta_1/p_1, \ldots, \theta_n/p_n] \text{ or} \\
&\quad \mathfrak{M}, w \Vdash \chi[\theta_1/p_1, \ldots, \theta_n/p_n], && \text{by induction hypothesis;}\\
&\Leftrightarrow \mathfrak{M}, w \Vdash (\psi \vee \chi)[\theta_1/p_1, \ldots, \theta_n/p_n] && \text{by definition of } \mathfrak{M}, w \Vdash.
\end{aligned}$$

6. $\varphi \equiv (\psi \to \chi)$:

$$\begin{aligned}
\mathfrak{v} \vDash \psi \to \chi &\Leftrightarrow \mathfrak{v} \nvDash \psi \text{ or } \mathfrak{v} \vDash \chi && \text{by definition of } \mathfrak{v} \vDash;\\
&\Leftrightarrow \mathfrak{M}, w \nVdash \psi[\theta_1/p_1, \ldots, \theta_n/p_n] \text{ or} \\
&\quad \mathfrak{M}, w \Vdash \chi[\theta_1/p_1, \ldots, \theta_n/p_n], && \text{by induction hypothesis;}\\
&\Leftrightarrow \mathfrak{M}, w \Vdash (\psi \to \chi)[\theta_1/p_1, \ldots, \theta_n/p_n] && \text{by definition of } \mathfrak{M}, w \Vdash.
\end{aligned}$$

□

**Proposition 38.15.** *All tautological instances are valid.*

*Proof.* Contrapositively, suppose $\varphi$ is such that $\mathfrak{M}, w \nVdash \varphi[\theta_1/p_1, \ldots, \theta_n/p_n]$, for some model $\mathfrak{M}$ and world $w$. Define an assignment $v$ such that $v(p_i) = \mathbb{T}$ if and only if $\mathfrak{M}, w \Vdash \theta_i$ (and $v$ assigns arbitrary values to $q \notin \{p_1, \ldots, p_n\}$). Then by Lemma 38.14, $\mathfrak{v} \nvDash \varphi$, so $\varphi$ is not a tautology. □

## 38.9 Schemas and Validity

**Definition 38.16.** A *schema* is a set of formulas comprising all and only the substitution instances of some modal formula $\chi$, i.e.,

$$\{\psi : \exists\theta_1, \ldots, \exists\theta_n\, (\psi = \chi[\theta_1/p_1, \ldots, \theta_n/p_n])\}.$$

The formula $\chi$ is called the *characteristic* formula of the schema, and it is unique up to a renaming of the propositional variables. A formula $\varphi$ is an *instance* of a schema if it is a member of the set.

It is convenient to denote a schema by the meta-linguistic expression obtained by substituting '$\varphi$', '$\psi$', ..., for the atomic components of $\chi$. So, for instance, the following denote schemas: '$\varphi$', '$\varphi \to \Box\varphi$', '$\varphi \to (\psi \to \varphi)$'. They correspond to the characteristic formulas $p$, $p \to \Box p$, $p \to (q \to p)$. The schema '$\varphi$' denotes the set of *all* formulas.

**Definition 38.17.** A schema is *true* in a model if and only if all of its instances are; and a schema is *valid* if and only if it is true in every model.

**Proposition 38.18.** *The following schema* K *is valid*

$$\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi). \tag{K}$$

*Proof.* We need to show that all instances of the schema are true at every world in every model. So let $\mathfrak{M} = \langle W, R, V\rangle$ and $w \in W$ be arbitrary. To show that a conditional is true at a world we assume the antecedent is true to show that consequent is true as well. In this case, let $\mathfrak{M}, w \Vdash \Box(\varphi \to \psi)$ and $\mathfrak{M}, w \Vdash \Box\varphi$. We need to show $\mathfrak{M} \Vdash \Box\psi$. So let $w'$ be arbitrary such that $Rww'$. Then by the first assumption $\mathfrak{M}, w' \Vdash \varphi \to \psi$ and by the second assumption $\mathfrak{M}, w' \Vdash \varphi$. It follows that $\mathfrak{M}, w' \Vdash \psi$. Since $w'$ was arbitrary, $\mathfrak{M}, w \Vdash \Box\psi$. □

**Proposition 38.19.** *The following schema* DUAL *is valid*

$$\Diamond\varphi \leftrightarrow \neg\Box\neg\varphi. \tag{DUAL}$$

*Proof.* Exercise. □

**Proposition 38.20.** *If $\varphi$ and $\varphi \to \psi$ are true at a world in a model then so is $\psi$. Hence, the valid formulas are closed under modus ponens.*

**Proposition 38.21.** *A formula $\varphi$ is valid iff all its substitution instances are. In other words, a schema is valid iff its characteristic formula is.*

*Proof.* The "if" direction is obvious, since $\varphi$ is a substitution instance of itself.

To prove the "only if" direction, we show the following: Suppose $\mathfrak{M} = \langle W, R, V\rangle$ is a modal model, and $\psi \equiv \varphi[D_1/p_1, \ldots, D_n/p_n]$ is a substitution instance of $\varphi$. Define $\mathfrak{M}' = \langle W, R, V'\rangle$ by $V(p_i) = \{w : \mathfrak{M} \Vdash \theta_i w\}$. Then $\mathfrak{M} \Vdash \psi w$ iff $\mathfrak{M}' \Vdash \varphi w$, for any $w \in W$. (We leave the proof as an exercise.) Now suppose that $\varphi$ was valid, but some substitution instance $\psi$ of $\varphi$ was not valid. Then for some $\mathfrak{M} = \langle W, R, V\rangle$ and some $w \in W$, $\mathfrak{M} \nVdash \psi w$. But then $\mathfrak{M}' \nVdash \varphi w$ by the claim, and $\varphi$ is not valid, a contradiction. □

Note, however, that it is not true that a schema is true in a model iff its characteristic formula is. Of course, the "only if" direction holds: if every instance of $\varphi$ is true in $\mathfrak{M}$, $\varphi$ itself is true in $\mathfrak{M}$. But it may happen that $\varphi$ is true in $\mathfrak{M}$ but some instance of $\varphi$ is false at some world in $\mathfrak{M}$. For a very simple counterexample consider $p$ in a model with only one world $w$ and $V(p) = \{w\}$, so that $p$ is true at $w$. But $\bot$ is an instance of $p$, and not true at $w$.

## 38.10 Entailment

With the definition of truth at a world, we can define an entailment relation between formulas. A formula $\psi$ entails $\varphi$ iff, whenever $\psi$ is true, $\varphi$ is true as well. Here, "whenever" means both "whichever model we consider" as well as "whichever world in that model we consider."

| *Valid Schemas* | *Invalid Schemas* |
|---|---|
| $\Box(\varphi \to \psi) \to (\Diamond\varphi \to \Diamond\psi)$ | $\Box(\varphi \vee \psi) \to (\Box\varphi \vee \Box\psi)$ |
| $\Diamond(\varphi \to \psi) \to (\Box\varphi \to \Diamond\psi)$ | $(\Diamond\varphi \wedge \Diamond\psi) \to \Diamond(\varphi \wedge \psi)$ |
| $\Box(\varphi \wedge \psi) \leftrightarrow (\Box\varphi \wedge \Box\psi)$ | $\varphi \to \Box\varphi$ |
| $\Box\varphi \to \Box(\psi \to \varphi)$ | $\Box\Diamond\varphi \to \psi$ |
| $\neg\Diamond\varphi \to \Box(\varphi \to \psi)$ | $\Box\Box\varphi \to \Box\varphi$ |
| $\Diamond(\varphi \vee \psi) \leftrightarrow (\Diamond\varphi \vee \Diamond\psi)$ | $\Box\Diamond\varphi \to \Diamond\Box\varphi$. |

Table 38.1: Valid and (or?) invalid schemas.

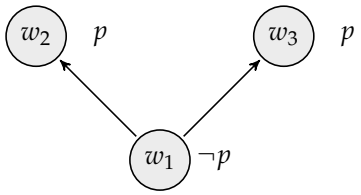



Figure 38.2: Counterexample to $p \to \Diamond p \vDash \Box p \to p$.

**Definition 38.22.** If $\Gamma$ is a set of formulas and $\varphi$ a formula, then $\Gamma$ *entails* $\varphi$, in symbols: $\Gamma \vDash \varphi$, if and only if for every model $\mathfrak{M} = \langle W, R, V\rangle$ and world $w \in W$, if $\mathfrak{M}, w \Vdash \psi$ for every $\psi \in \Gamma$, then $\mathfrak{M}, w \Vdash \varphi$. If $\Gamma$ contains a single formula $\psi$, then we write $\psi \vDash \varphi$.

**Example 38.23.** To show that a formula entails another, we have to reason about all models, using the definition of $\mathfrak{M}, w \Vdash$. For instance, to show $p \to \Diamond p \vDash \Box\neg p \to \neg p$, we might argue as follows: Consider a model $\mathfrak{M} = \langle W, R, V\rangle$ and $w \in W$, and suppose $\mathfrak{M}, w \Vdash p \to \Diamond p$. We have to show that $\mathfrak{M}, w \Vdash \Box\neg p \to \neg p$. Suppose not. Then $\mathfrak{M}, w \Vdash \Box\neg p$ and $\mathfrak{M}, w \nVdash \neg p$. Since $\mathfrak{M}, w \nVdash \neg p$, $\mathfrak{M}, w \Vdash p$. By assumption, $\mathfrak{M}, w \Vdash p \to \Diamond p$, hence $\mathfrak{M}, w \Vdash \Diamond p$. By definition of $\mathfrak{M}, w \Vdash \Diamond p$, there is some $w'$ with $Rww'$ such that $\mathfrak{M}, w' \Vdash p$. Since also $\mathfrak{M}, w \Vdash \Box\neg p$, $\mathfrak{M}, w' \Vdash \neg p$, a contradiction.

To show that a formula $\psi$ does not entail another $\varphi$, we have to give a counterexample, i.e., a model $\mathfrak{M} = \langle W, R, V\rangle$ where we show that at some world $w \in W$, $\mathfrak{M}, w \Vdash \psi$ but $\mathfrak{M}, w \nVdash \varphi$. Let's show that $p \to \Diamond p \nvDash \Box p \to p$. Consider the model in Figure 38.2. We have $\mathfrak{M}, w_1 \Vdash \Diamond p$ and hence $\mathfrak{M}, w_1 \Vdash p \to \Diamond p$. However, since $\mathfrak{M}, w_1 \Vdash \Box p$ but $\mathfrak{M}, w_1 \nVdash p$, we have $\mathfrak{M}, w_1 \nVdash \Box p \to p$.

Often very simple counterexamples suffice. The model $\mathfrak{M}' = \{W', R', V'\}$ with $W' = \{w\}$, $R' = \emptyset$, and $V'(p) = \emptyset$ is also a counterexample: Since $\mathfrak{M}', w \nVdash p$, $\mathfrak{M}', w \Vdash p \to \Diamond p$. As no worlds are accessible from $w$, we have $\mathfrak{M}', w \Vdash \Box p$, and so $\mathfrak{M}', w \nVdash \Box p \to p$.

## Problems

**Problem 38.1.** Consider the model of Figure 38.1. Which of the following hold?

1. $\mathfrak{M}, w_1 \Vdash q$;
2. $\mathfrak{M}, w_3 \Vdash \neg q$;
3. $\mathfrak{M}, w_1 \Vdash p \vee q$;
4. $\mathfrak{M}, w_1 \Vdash \Box(p \vee q)$;
5. $\mathfrak{M}, w_3 \Vdash \Box q$;
6. $\mathfrak{M}, w_3 \Vdash \Box\bot$;
7. $\mathfrak{M}, w_1 \Vdash \Diamond q$;
8. $\mathfrak{M}, w_1 \Vdash \Box q$;
9. $\mathfrak{M}, w_1 \Vdash \neg\Box\Box\neg q$.

**Problem 38.2.** Complete the proof of Proposition 38.7.

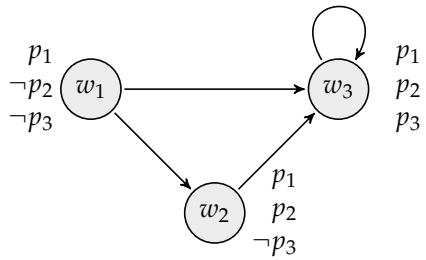
**Problem 38.3.** Let $\mathfrak{M} = \langle W, R, V\rangle$ be a model, and suppose $w_1, w_2 \in W$ are such that:

1. $w_1 \in V(p)$ if and only if $w_2 \in V(p)$; and
2. for all $w \in W$: $Rw_1w$ if and only if $Rw_2w$.

Using induction on formulas, show that for all formulas $\varphi$: $\mathfrak{M}, w_1 \Vdash \varphi$ if and only if $\mathfrak{M}, w_2 \Vdash \varphi$.

**Problem 38.4.** Let $\mathfrak{M} = \langle M, R, V\rangle$. Show that $\mathfrak{M}, w \Vdash \neg\Diamond\varphi$ if and only if $\mathfrak{M}, w \Vdash \Box\neg\varphi$.

**Problem 38.5.** Consider the following model $\mathfrak{M}$ for the language comprising $p_1$, $p_2$, $p_3$ as the only propositional variables:



Are the following formulas and schemas true in the model $\mathfrak{M}$, i.e., true at every world in $\mathfrak{M}$? Explain.

1. $p \to \Diamond p$ (for $p$ atomic);
2. $\varphi \to \Diamond \varphi$ (for $\varphi$ arbitrary);
3. $\Box p \to p$ (for $p$ atomic);
4. $\neg p \to \Diamond \Box p$ (for $p$ atomic);
5. $\Diamond \Box \varphi$ (for $\varphi$ arbitrary);
6. $\Box \Diamond p$ (for $p$ atomic).

**Problem 38.6.** Show that the following are valid:

1. $\vDash \Box p \to \Box(q \to p)$;
2. $\vDash \Box \neg \bot$;
3. $\vDash \Box p \to (\Box q \to \Box p)$.

**Problem 38.7.** Show that $\varphi \to \Box \varphi$ is valid in the class $\mathcal{C}$ of models $\mathfrak{M} = \langle W, R, V \rangle$ where $W = \{w\}$. Similarly, show that $\psi \to \Box \varphi$ and $\Diamond \varphi \to \psi$ are valid in the class of models $\mathfrak{M} = \langle W, R, V \rangle$ where $R = \emptyset$.

**Problem 38.8.** Prove Proposition 38.19.

**Problem 38.9.** Prove the claim in the "only if" part of the proof of Proposition 38.21. (Hint: use induction on $\varphi$.)

**Problem 38.10.** Show that none of the following formulas are valid:

D: $\Box p \to \Diamond p$;

T: $\Box p \to p$;

B: $p \to \Box \Diamond p$;

4: $\Box p \to \Box \Box p$;

5: $\Diamond p \to \Box \Diamond p$.

**Problem 38.11.** Prove that the schemas in the first column of table 38.1 are valid and those in the second column are not valid.

**Problem 38.12.** Decide whether the following schemas are valid or invalid:

1. $(\Diamond \varphi \to \Box \psi) \to (\Box \varphi \to \Box \psi)$;
2. $\Diamond(\varphi \to \psi) \vee \Box(\psi \to \varphi)$.

**Problem 38.13.** For each of the following schemas find a model $\mathfrak{M}$ such that every instance of the formula is true in $\mathfrak{M}$:

1. $p \to \Diamond\Diamond p$;
2. $\Diamond p \to \Box p$.

**Problem 38.14.** Show that $\Box(\varphi \wedge \psi) \vDash \Box\varphi$.

**Problem 38.15.** Show that $\Box(p \to q) \nvDash p \to \Box q$ and $p \to \Box q \nvDash \Box(p \to q)$.

# Chapter 39

# Frame Definability

## 39.1 Introduction

One question that interests modal logicians is the relationship between the accessibility relation and the truth of certain formulas in models with that accessibility relation. For instance, suppose the accessibility relation is reflexive, i.e., for every $w \in W$, $Rww$. In other words, every world is accessible from itself. That means that when $\Box\varphi$ is true at a world $w$, $w$ itself is among the accessible worlds at which $\varphi$ must therefore be true. So, if the accessibility relation $R$ of $\mathfrak{M}$ is reflexive, then whatever world $w$ and formula $\varphi$ we take, $\Box\varphi \to \varphi$ will be true there (in other words, the schema $\Box p \to p$ and all its substitution instances are true in $\mathfrak{M}$).

The converse, however, is false. It's not the case, e.g., that if $\Box p \to p$ is true in $\mathfrak{M}$, then $R$ is reflexive. For we can easily find a non-reflexive model $\mathfrak{M}$ where $\Box p \to p$ is true at all worlds: take the model with a single world $w$, not accessible from itself, but with $w \in V(p)$. By picking the truth value of $p$ suitably, we can make $\Box\varphi \to \varphi$ true in a model that is not reflexive.

The solution is to remove the variable assignment $V$ from the equation. If we require that $\Box p \to p$ is true at all worlds in $\mathfrak{M}$, regardless of which worlds are in $V(p)$, then it is necessary that $R$ is reflexive. For in any non-reflexive model, there will be at least one world $w$ such that not $Rww$. If we set $V(p) = W \setminus \{w\}$, then $p$ will be true at all worlds other than $w$, and so at all worlds accessible from $w$ (since $w$ is guaranteed not to be accessible from $w$, and $w$ is the only world where $p$ is false). On the other hand, $p$ is false at $w$, so $\Box p \to p$ is false at $w$.

This suggests that we should introduce a notation for model structures without a valuation: we call these *frames*. A frame $\mathfrak{F}$ is simply a pair $\langle W, R\rangle$ consisting of a set of worlds with an accessibility relation. Every model $\langle W, R, V\rangle$ is then, as we say, *based on* the frame $\langle W, R\rangle$. Conversely, a frame determines the class of models based on it; and a class of frames determines the class of models which are based on any frame in the class. And we can define $\mathfrak{F} \vDash \varphi$,

| *If R is ...* | *then ... is true in $\mathfrak{M}$:* | |
|---|---|---|
| *serial*: $\forall u \exists v Ruv$ | $\Box p \to \Diamond p$ | (D) |
| *reflexive*: $\forall w Rww$ | $\Box p \to p$ | (T) |
| *symmetric*: $\forall u \forall v (Ruv \to Rvu)$ | $p \to \Box \Diamond p$ | (B) |
| *transitive*: $\forall u \forall v \forall w ((Ruv \wedge Rvw) \to Ruw)$ | $\Box p \to \Box \Box p$ | (4) |
| *euclidean*: $\forall w \forall u \forall v ((Rwu \wedge Rwv) \to Ruv$ | $\Diamond p \to \Box \Diamond p$ | (5) |

Table 39.1: Five correspondence facts.

the notion of a formula being *valid* in a frame as: $\mathfrak{M} \Vdash \varphi$ for all $\mathfrak{M}$ based on $\mathfrak{F}$.

With this notation, we can establish correspondence relations between formulas and classes of frames: e.g., $\mathfrak{F} \vDash \Box p \to p$ if, and only if, $\mathfrak{F}$ is reflexive.

## 39.2 Properties of Accessibility Relations

Many modal formulas turn out to be characteristic of simple, and even familiar, properties of the accessibility relation. In one direction, that means that any model that has a given property makes a corresponding formula (and all its substitution instances) true. We begin with five classical examples of kinds of accessibility relations and the formulas the truth of which they guarantee.

**Theorem 39.1.** *Let $\mathfrak{M} = \langle W, R, V \rangle$ be a model. If $R$ has the property on the left side of table 39.1, every instance of the formula on the right side is true in $\mathfrak{M}$.*

*Proof.* Here is the case for B: to show that the schema is true in a model we need to show that all of its instances are true all worlds in the model. So let $\varphi \to \Box \Diamond \varphi$ be a given instance of B, and let $w \in W$ be an arbitrary world. Suppose the antecedent $\varphi$ is true at $w$, in order to show that $\Box \Diamond \varphi$ is true at $w$. So we need to show that $\Diamond \varphi$ is true at all $w'$ accessible from $w$. Now, for any $w'$ such that $Rww'$ we have, using the hypothesis of symmetry, that also $Rw'w$ (see Figure 39.1). Since $\mathfrak{M}, w \Vdash \varphi$, we have $\mathfrak{M}, w' \Vdash \Diamond \varphi$. Since $w'$ was an arbitrary world such that $Rww'$, we have $\mathfrak{M}, w \Vdash \Box \Diamond \varphi$.

We leave the other cases as exercises. □

Notice that the converse implications of Theorem 39.1 do not hold: it's not true that if a model verifies a schema, then the accessibility relation of that model has the corresponding property. In the case of T and reflexive models, it is easy to give an example of a model in which T itself fails: let $W = \{w\}$ and $V(p) = \emptyset$. Then $R$ is not reflexive, but $\mathfrak{M}, w \Vdash \Box p$ and $\mathfrak{M}, w \nVdash p$. But here we have just a single instance of T that fails in $\mathfrak{M}$, other instances, e.g., $\Box \neg p \to \neg p$

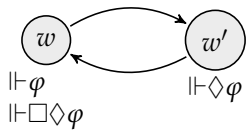



Figure 39.1: The argument from symmetry.

are true. It is harder to give examples where *every substitution instance* of T is true in $\mathfrak{M}$ and $\mathfrak{M}$ is not reflexive. But there are such models, too:

**Proposition 39.2.** *Let $\mathfrak{M} = \langle W, R, V\rangle$ be a model such that $W = \{u, v\}$, where worlds $u$ and $v$ are related by $R$: i.e., both $Ruv$ and $Rvu$. Suppose that for all $p$: $u \in V(p) \Leftrightarrow v \in V(p)$. Then:*

1. *For all $\varphi$: $\mathfrak{M}, u \Vdash \varphi$ if and only if $\mathfrak{M}, v \Vdash \varphi$ (use induction on $\varphi$).*

2. *Every instance of* T *is true in $\mathfrak{M}$.*

*Since $\mathfrak{M}$ is not reflexive (it is, in fact,* irreflexive*), the converse of Theorem 39.1 fails in the case of* T *(similar arguments can be given for some—though not all—the other schemas mentioned in Theorem 39.1).*

Although we will focus on the five classical formulas D, T, B, 4, and 5, we record in table 39.2 a few more properties of accessibility relations. The accessibility relation $R$ is partially functional, if from every world at most one world is accessible. If it is the case that from every world exactly one world is accessible, we call it functional. (Thus the functional relations are precisely those that are both serial and partially functional). They are called "functional" because the accessibility relation operates like a (partial) function. A relation is weakly dense if whenever $Ruv$, there is a $w$ "between" $u$ and $v$. So weakly dense relations are in a sense the opposite of transitive relations: in a transitive relation, whenever you can reach $v$ from $u$ by a detour via $w$, you can reach $v$ from $u$ directly; in a weakly dense relation, whenever you can reach $v$ from $u$ directly, you can also reach it by a detour via some $w$. A relation is weakly directed if whenever you can reach worlds $u$ and $v$ from some world $w$, you can reach a single world $t$ from both $u$ and $v$—this is sometimes called the "diamond property" or "confluence."

## 39.3 Frames

**Definition 39.3.** A *frame* is a pair $\mathfrak{F} = \langle W, R\rangle$ where $W$ is a non-empty set of worlds and $R$ a binary relation on $W$. A model $\mathfrak{M}$ is *based on* a frame $\mathfrak{F} = \langle W, R\rangle$ if and only if $\mathfrak{M} = \langle W, R, V\rangle$.

| *If R is ...* | *then ... is true in $\mathfrak{M}$:* |
|---|---|
| *partially functional*: $\forall w \forall u \forall v((Rwu \wedge Rwv) \rightarrow u = v)$ | $\Diamond p \rightarrow \Box p$ |
| *functional*: $\forall w \exists v \forall u(Rwu \leftrightarrow u = v)$ | $\Diamond p \leftrightarrow \Box p$ |
| *weakly dense*: $\forall u \forall v(Ruv \rightarrow \exists w(Ruw \wedge Rwv))$ | $\Box\Box p \rightarrow \Box p$ |
| *weakly connected*: $\forall w \forall u \forall v((Rwu \wedge Rwv) \rightarrow (Ruv \vee u = v \vee Rvu))$ | $\Box((p \wedge \Box p) \rightarrow q) \vee \Box((q \wedge \Box q) \rightarrow p)$ (L) |
| *weakly directed*: $\forall w \forall u \forall v((Rwu \wedge Rwv) \rightarrow \exists t(Rut \wedge Rvt))$ | $\Diamond\Box p \rightarrow \Box\Diamond p$ (G) |

Table 39.2: Five more correspondence facts.

**Definition 39.4.** If $\mathfrak{F}$ is a frame, we say that $\varphi$ is *valid in* $\mathfrak{F}$, $\mathfrak{F} \vDash \varphi$, if $\mathfrak{M} \Vdash \varphi$ for every model $\mathfrak{M}$ based on $\mathfrak{F}$.

If $\mathcal{F}$ is a class of frames, we say $\varphi$ is *valid in* $\mathcal{F}$, $\mathcal{F} \vDash \varphi$, iff $\mathfrak{F} \vDash \varphi$ for every frame $\mathfrak{F} \in \mathcal{F}$.

The reason frames are interesting is that correspondence between schemas and properties of the accessibility relation $R$ is at the level of frames, *not of models*. For instance, although T is true in all reflexive models, not every model in which T is true is reflexive. However, it *is* true that not only is T *valid* on all reflexive *frames*, also every frame in which T is valid is reflexive.

*Remark* 6. Validity in a class of frames is a special case of the notion of validity in a class of models: $\mathcal{F} \vDash \varphi$ iff $\mathcal{C} \vDash \varphi$ where $\mathcal{C}$ is the class of all models based on a frame in $\mathcal{F}$.

Obviously, if a formula or a schema is valid, i.e., valid with respect to the class of *all* models, it is also valid with respect to any class $\mathcal{F}$ of frames.

## 39.4 Frame Definability

Even though the converse implications of Theorem 39.1 fail, they hold if we replace "model" by "frame": for the properties considered in Theorem 39.1, it *is* true that if a formula is valid in a *frame* then the accessibility relation of that frame has the corresponding property. So, the formulas considered *define* the classes of frames that have the corresponding property.

**Definition 39.5.** If $\mathcal{C}$ is a class of frames, we say $\varphi$ *defines* $\mathcal{C}$ iff $\mathfrak{F} \vDash \varphi$ for all and only frames $\mathfrak{F} \in \mathcal{C}$.

We now proceed to establish the full definability results for frames.

**Theorem 39.6.** *If the formula on the right side of table 39.1 is valid in a frame $\mathfrak{F}$, then $\mathfrak{F}$ has the property on the left side.*

*Proof.* 1. Suppose D is valid in $\mathfrak{F} = \langle W, R\rangle$, i.e., $\mathfrak{F} \vDash \Box p \to \Diamond p$. Let $\mathfrak{M} = \langle W, R, V\rangle$ be a model based on $\mathfrak{F}$, and $w \in W$. We have to show that there is a $v$ such that $Rwv$. Suppose not: then both $\mathfrak{M} \Vdash \Box\varphi$ and $\mathfrak{M}, w \nVdash \Diamond\varphi$ for any $\varphi$, including $p$. But then $\mathfrak{M}, w \nVdash \Box p \to \Diamond p$, contradicting the assumption that $\mathfrak{F} \vDash \Box p \to \Diamond p$.

2. Suppose T is valid in $\mathfrak{F}$, i.e., $\mathfrak{F} \vDash \Box p \to p$. Let $w \in W$ be an arbitrary world; we need to show $Rww$. Let $u \in V(p)$ if and only if $Rwu$ (when $q$ is other than $p$, $V(q)$ is arbitrary, say $V(q) = \emptyset$). Let $\mathfrak{M} = \langle W, R, V\rangle$. By construction, for all $u$ such that $Rwu$: $\mathfrak{M}, u \Vdash p$, and hence $\mathfrak{M}, w \Vdash \Box p$. But by hypothesis $\Box p \to p$ is true at $w$, so that $\mathfrak{M}, w \Vdash p$, but by definition of $V$ this is possible only if $Rww$.

3. We prove the contrapositive: Suppose $\mathfrak{F}$ is not symmetric, we show that B, i.e., $p \to \Box\Diamond p$ is not valid in $\mathfrak{F} = \langle W, R\rangle$. If $\mathfrak{F}$ is not symmetric, there are $u, v \in W$ such that $Ruv$ but not $Rvu$. Define $V$ such that $w \in V(p)$ if and only if not $Rvw$ (and $V$ is arbitrary otherwise). Let $\mathfrak{M} = \langle W, R, V\rangle$. Now, by definition of $V$, $\mathfrak{M}, w \Vdash p$ for all $w$ such that not $Rvw$, in particular, $\mathfrak{M}, u \Vdash p$ since not $Rvu$. Also, since $Rvw$ iff $p \notin V(w)$, there is no $w$ such that $Rvw$ and $\mathfrak{M}, w \Vdash p$, and hence $\mathfrak{M}, v \nVdash \Diamond p$. Since $Ruv$, also $\mathfrak{M}, u \nVdash \Box\Diamond p$. It follows that $\mathfrak{M}, u \nVdash p \to \Box\Diamond p$, and so B is not valid in $\mathfrak{F}$.

4. Suppose 4 is valid in $\mathfrak{F} = \langle W, R\rangle$, i.e., $\mathfrak{F} \vDash \Box p \to \Box\Box p$, and let $u$, $v$, $w \in W$ be arbitrary worlds such that $Ruv$ and $Rvw$; we need to show that $Ruw$. Define $V$ such that $z \in V(p)$ if and only if $Ruz$ (and $V$ is arbitrary otherwise). Let $\mathfrak{M} = \langle W, R, V\rangle$. By definition of $V$, $\mathfrak{M}, z \Vdash p$ for all $z$ such that $Ruz$, and hence $\mathfrak{M}, u \Vdash \Box p$. But by hypothesis 4, $\Box p \to \Box\Box p$, is true at $u$, so that $\mathfrak{M}, u \Vdash \Box\Box p$. Since $Ruv$ and $Rvw$, we have $\mathfrak{M}, w \Vdash p$, but by definition of $V$ this is possible only if $Ruw$, as desired.

5. We proceed contrapositively, assuming that the frame $\mathfrak{F} = \langle W, R\rangle$ is not euclidean, and show that it falsifies 5, i.e., i.e., $\mathfrak{F} \nvDash \Diamond p \to \Box\Diamond p$. Suppose there are worlds $u$, $v$, $w \in W$ such that $Rwu$ and $Rwv$ but not $Ruv$. Define $V$ such that for all worlds $z$, $z \in V(p)$ if and only if it is *not* the case that $Ruz$. Let $\mathfrak{M} = \langle W, R, V\rangle$. Then by hypothesis $\mathfrak{M}, v \Vdash p$ and since $Rwv$ also $\mathfrak{M}, w \Vdash \Diamond p$. However, there is no world $y$ such that $Ruy$ and $\mathfrak{M}, y \Vdash p$ so $\mathfrak{M}, u \nVdash \Diamond p$. Since $Rwu$, it follows that $\mathfrak{M}, w \nVdash \Box\Diamond p$, so that 5, $\Diamond p \to \Box\Diamond p$, fails at $w$.

□

You'll notice a difference between the proof for D and the other cases: no mention was made of the valuation $V$. In effect, we proved that if $\mathfrak{M} \Vdash \mathrm{D}$ then $\mathfrak{M}$ is serial. So D defines the class of serial *models*, not just frames.

**Corollary 39.7.** *Any model where* D *is true is serial.*

**Corollary 39.8.** *Each formula on the right side of table 39.1 defines the class of frames which have the property on the left side.*

*Proof.* In Theorem 39.1, we proved that if a model has the property on the left, the formula on the right is true in it. Thus, if a frame $\mathfrak{F}$ has the property on the left, the formula on the left is valid in $\mathfrak{F}$. In Theorem 39.6, we proved the converse implications: if a formula on the right is valid in $\mathfrak{F}$, $\mathfrak{F}$ has the property on the left. □

Theorem 39.6 also shows that the properties can be combined: for instance if both B and 4 are valid in $\mathfrak{F}$ then the frame is both symmetric and transitive, etc. Many important modal logics are characterized as the set of formulas valid in all frames that combine some frame properties, and so we can characterize them as the set of formulas valid in all frames in which the corresponding defining formulas are valid. For instance, the classical system **S4** is the set of all formulas valid in all reflexive and transitive frames, i.e., in all those where both T and 4 are valid. **S5** is the set of all formulas valid in all reflexive, symmetric, and euclidean frames, i.e., all those where all of T, B, and 5 are valid.

Logical relationships between properties of $R$ in general correspond to relationships between the corresponding defining formulas. For instance, every reflexive relation is serial; hence, whenever T is valid in a frame, so is D. (Note that this relationship is *not* that of entailment. It is not the case that whenever $\mathfrak{M}, w \Vdash \mathrm{T}$ then $\mathfrak{M}, w \Vdash \mathrm{D}$.) We record some such relationships.

**Proposition 39.9.** *Let $R$ be a binary relation on a set $W$; then:*

1. *If $R$ is reflexive, then it is serial.*
2. *If $R$ is symmetric, then it is transitive if and only if it is euclidean.*
3. *If $R$ is symmetric or euclidean then it is weakly directed (it has the "diamond property").*
4. *If $R$ is euclidean then it is weakly connected.*
5. *If $R$ is functional then it is serial.*

## 39.5 First-order Definability

We've seen that a number of properties of accessibility relations of frames can be defined by modal formulas. For instance, symmetry of frames can be defined by the formula B, $p \to \Box\Diamond p$. The conditions we've encountered so far can all be expressed by first-order formulas in a language involving a single two-place predicate symbol. For instance, symmmetry is defined by $\forall x \forall y\,(Q(x,y) \to Q(y,x))$ in the sense that a first-order structure $\mathfrak{M}$ with $|\mathfrak{M}| = W$ and $Q^{\mathfrak{M}} = R$ satisfies the preceding formula iff $R$ is symmetric. This suggests the following definition:

**Definition 39.10.** A class $\mathcal{C}$ of frames is *first-order definable* if there is a sentence $\varphi$ in the first-order language with a single two-place predicate symbol $P$ such that $\mathfrak{F} = \langle W, R\rangle \in \mathcal{C}$ iff $\mathfrak{M} \vDash \varphi$ in the first-order structure $\mathfrak{M}$ with $|\mathfrak{M}| = W$ and $Q^{\mathfrak{M}} = R$.

It turns out that the properties and modal formulas that define them considered so far are exceptional. Not every formula defines a first-order definable class of frames, and not every first-order definable class of frames is definable by a modal formula.

A counterexample to the first is given by the Löb formula:

$$\Box(\Box p \to p) \to \Box p. \tag{W}$$

W defines the class of transitive and converse well-founded frames. A relation is well-founded if there is no infinite sequence $w_1, w_2, \ldots$ such that $Rw_2w_1$, $Rw_3w_2, \ldots$. For instance, the relation $<$ on $\mathbb{N}$ is well-founded, whereas the relation $<$ on $\mathbb{Z}$ is not. A relation is converse well-founded iff its converse is well-founded. So converse well-founded relations are those where there is no infinite sequence $w_1, w_2, \ldots$ such that $Rw_1w_2, Rw_2w_3, \ldots$.

There is, however, no first-order formula defining transitive converse well-founded relations. For suppose $\mathfrak{M} \vDash \beta$ iff $R = Q^{\mathfrak{M}}$ is transitive converse well-founded. Let $\varphi_n$ be the formula

$$(Q(a_1,a_2) \wedge \cdots \wedge Q(a_{n-1},a_n))$$

Now consider the set of formulas

$$\Gamma = \{\beta, \varphi_1, \varphi_2, \ldots\}.$$

Every finite subset of $\Gamma$ is satisfiable: Let $k$ be largest such that $\varphi_k$ is in the subset, $|\mathfrak{M}_k| = \{1,\ldots,k\}$, $a_i^{\mathfrak{M}_k} = i$, and $P^{\mathfrak{M}_k} = <$. Since $<$ on $\{1,\ldots,k\}$ is transitive and converse well-founded, $\mathfrak{M}_k \vDash \beta$. $\mathfrak{M}_k \vDash \varphi_i$ by construction, for all $i \leq k$. By the Compactness Theorem for first-order logic, $\Gamma$ is satisfiable in some structure $\mathfrak{M}$. By hypothesis, since $\mathfrak{M} \vDash \beta$, the relation $Q^{\mathfrak{M}}$ is converse well-founded. But clearly, $a_1^{\mathfrak{M}}, a_2^{\mathfrak{M}}, \ldots$ would form an infinite sequence of the kind ruled out by converse well-foundedness.

A counterexample to the second claim is given by the property of universality: for every $u$ and $v$, $Ruv$. Universal frames are first-order definable by the formula $\forall x \forall y\, Q(x, y)$. However, no modal formula is valid in all and only the universal frames. This is a consequence of a result that is independently interesting: the formulas valid in universal frames are exactly the same as those valid in reflexive, symmetric, and transitive frames. There are reflexive, symmetric, and transitive frames that are not universal, hence every formula valid in all universal frames is also valid in some non-universal frames.

## 39.6 Equivalence Relations and S5

The modal logic **S5** is characterized as the set of formulas valid on all universal frames, i.e., every world is accessible from every world, including itself. In such a scenarion, $\Box$ corresponds to necessity and $\Diamond$ to possibility: $\Box\varphi$ is true if $\varphi$ is true at *every* world, and $\Diamond\varphi$ is true if $\varphi$ is true at *some* world. It turns out that **S5** can also be characterized as the formulas valid on all reflexive, symmetric, and transitive frames, i.e., on all *equivalence relations*.

**Definition 39.11.** A binary relation $R$ on $W$ is an *equivalence relation* if and only if it is reflexive, symmetric and transitive. A relation $R$ on $W$ is *universal* if and only if $Ruv$ for all $u, v \in W$.

Since T, B, and 4 characterize the reflexive, symmetric, and transitive frames, the frames where the accessibility relation is an equivalence relation are exactly those in which all four formulas are valid. It turns out that the equivalence relations can also be characterized by other combinations of formulas, since the conditions with which we've defined equivalence relations are equivalent to combinations of other familiar conditions on $R$.

**Proposition 39.12.** *The following are equivalent:*

1. *$R$ is an equivalence relation;*
2. *$R$ is reflexive and euclidean;*
3. *$R$ is serial, symmetric, and euclidean;*
4. *$R$ is serial, symmetric, and transitive.*

*Proof.* Exercise. □

Proposition 39.12 is the semantic counterpart to Proposition 40.29, in that it gives equivalent characterization of the modal logic of frames over which $R$ is an equivalence (the logic traditionally referred to as **S5**).

What is the relationship between universal and equivalence relations? Although every universal relation is an equivalence relation, clearly not every equivalence relation is universal. However, the formulas valid on all universal relations are exactly the same as those valid on all equivalence relations.

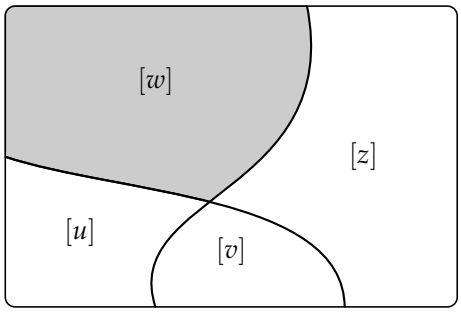



Figure 39.2: A partition of $W$ in equivalence classes.

**Proposition 39.13.** *Let $R$ be an equivalence relation, and for each $w \in W$ define the* equivalence class *of $w$ as the set $[w] = \{w' \in W : Rww'\}$. Then:*

1. *$w \in [w]$;*
2. *$R$ is universal on each equivalence class $[w]$;*
3. *The collection of equivalence classes partitions $W$ into mutually exclusive and jointly exhaustive subsets.*

**Proposition 39.14.** *A formula $\varphi$ is valid in all frames $\mathfrak{F} = \langle W, R\rangle$ where $R$ is an equivalence relation, if and only if it is valid in all frames $\mathfrak{F} = \langle W, R\rangle$ where $R$ is universal. Hence, the logic of universal frames is just* **S5**.

*Proof.* It's immediate to verify that a universal relation $R$ on $W$ is an equivalence. Hence, if $\varphi$ is valid in all frames where $R$ is an equivalence it is valid in all universal frames. For the other direction, we argue contrapositively: suppose $\psi$ is a formula that fails at a world $w$ in a model $\mathfrak{M} = \langle W, R, V\rangle$ based on a frame $\langle W, R\rangle$, where $R$ is an equivalence on $W$. So $\mathfrak{M}, w \nVdash \psi$. Define a model $\mathfrak{M}' = \langle W', R', V'\rangle$ as follows:

1. $W' = [w]$;
2. $R'$ is universal on $W'$;
3. $V'(p) = V(p) \cap W'$.

(So the set $W'$ of worlds in $\mathfrak{M}'$ is represented by the shaded area in Figure 39.2.) It is easy to see that $R$ and $R'$ agree on $W'$. Then one can show by induction on formulas that for all $w' \in W'$: $\mathfrak{M}', w' \Vdash \varphi$ if and only if $\mathfrak{M}, w' \Vdash \varphi$ for each $\varphi$ (this makes sense since $W' \subseteq W$). In particular, $\mathfrak{M}', w \nVdash \psi$, and $\psi$ fails in a model based on a universal frame. □

## 39.7 Second-order Definability

Not every frame property definable by modal formulas is first-order definable. However, if we allow quantification over one-place predicates (i.e., monadic second-order quantification), we define all modally definable frame properties. The trick is to exploit a systematic way in which the conditions under which a modal formula is true at a world are related to first-order formulas. This is the so-called standard translation of modal formulas into first-order formulas in a language containing not just a two-place predicate symbol $Q$ for the accessibility relation, but also a one-place predicate symbol $P_i$ for the propositional variables $p_i$ occurring in $\varphi$.

**Definition 39.15.** The *standard translation* $\mathrm{ST}_x(\varphi)$ is inductively defined as follows:

1. $\varphi \equiv \bot$: $\mathrm{ST}_x(\varphi) = \bot$.
2. $\varphi \equiv p_i$: $\mathrm{ST}_x(\varphi) = P_i(x)$.
3. $\varphi \equiv \neg\psi$: $\mathrm{ST}_x(\varphi) = \neg\mathrm{ST}_x(\psi)$.
4. $\varphi \equiv (\psi \wedge \chi)$: $\mathrm{ST}_x(\varphi) = (\mathrm{ST}_x(\psi) \wedge \mathrm{ST}_x(\chi))$.
5. $\varphi \equiv (\psi \vee \chi)$: $\mathrm{ST}_x(\varphi) = (\mathrm{ST}_x(\psi) \vee \mathrm{ST}_x(\chi))$.
6. $\varphi \equiv (\psi \to \chi)$: $\mathrm{ST}_x(\varphi) = (\mathrm{ST}_x(\psi) \to \mathrm{ST}_x(\chi))$.
7. $\varphi \equiv \Box\psi$: $\mathrm{ST}_x(\varphi) = \forall y\,(Q(x,y) \to \mathrm{ST}_y(\psi))$.
8. $\varphi \equiv \Diamond\psi$: $\mathrm{ST}_x(\varphi) = \exists y\,(Q(x,y) \wedge \mathrm{ST}_y(\psi))$.

For instance, $\mathrm{ST}_x(\Box p \to p)$ is $\forall y\,(Q(x,y) \to P(y)) \to P(x)$. Any structure for the language of $\mathrm{ST}_x(\varphi)$ requires a domain, a two-place relation assigned to $Q$, and subsets of the domain assigned to the one-place predicate symbols $P_i$. In other words, the components of such a structure are exactly those of a model for $\varphi$: the domain is the set of worlds, the two-place relation assigned to $Q$ is the accessibility relation, and the subsets assigned to $P_i$ are just the assignments $V(p_i)$. It won't surprise that satisfaction of $\varphi$ in a modal model and of $\mathrm{ST}_x(\varphi)$ in the corresponding structure agree:

**Proposition 39.16.** *Let $\mathfrak{M} = \langle W, R, V\rangle$, $\mathfrak{M}'$ be the first-order structure with $|\mathfrak{M}'| = W$, $Q^{\mathfrak{M}} = R$, and $P_i^{\mathfrak{M}'} = V(p_i)$, and $s(x) = w$. Then*

$$\mathfrak{M}, w \Vdash \varphi \textit{ iff } \mathfrak{M}', s \vDash \mathrm{ST}_x(\varphi)$$

*Proof.* By induction on $\varphi$. □

**Proposition 39.17.** *Suppose $\varphi$ is a modal formula and $\mathfrak{F} = \langle W, R\rangle$ is a frame. Let $\mathfrak{F}'$ be the first-order structure with $|\mathfrak{F}'| = W$ and $Q^{\mathfrak{F}'} = R$, and let $\varphi'$ be the second-order formula*

$$\forall X_1 \ldots \forall X_n \, \forall x \, \mathrm{ST}_x(\varphi)[X_1/P_1, \ldots, X_n/P_n],$$

*where $P_1, \ldots, P_n$ are all one-place predicate symbols in $\mathrm{ST}_x(\varphi)$. Then*

$$\mathfrak{F} \vDash \varphi \textit{ iff } \mathfrak{F}' \vDash \varphi'$$

*Proof.* $\mathfrak{F}' \vDash \varphi'$ iff for every structure $\mathfrak{M}'$ where $P_i^{\mathfrak{M}'} \subseteq W$ for $i = 1, \ldots, n$, and for every $s$ with $s(x) \in W$, $\mathfrak{M}', s \vDash \mathrm{ST}_x(\varphi)$. By Proposition 39.16, that is the case iff for all models $\mathfrak{M}$ based on $\mathfrak{F}$ and every world $w \in W$, $\mathfrak{M}, w \Vdash \varphi$, i.e., $\mathfrak{F} \vDash \varphi$. □

**Definition 39.18.** A class $\mathcal{C}$ of frames is *second-order definable* if there is a sentence $\varphi$ in the second-order language with a single two-place predicate symbol $P$ and quantifiers only over monadic set variables such that $\mathfrak{F} = \langle W, R\rangle \in \mathcal{C}$ iff $\mathfrak{M} \vDash \varphi$ in the structure $\mathfrak{M}$ with $|\mathfrak{M}| = W$ and $P^{\mathfrak{M}} = R$.

**Corollary 39.19.** *If a class of frames is definable by a formula $\varphi$, the corresponding class of accessibility relations is definable by a monadic second-order sentence.*

*Proof.* The monadic second-order sentence $\varphi'$ of the preceding proof has the required property. □

As an example, consider again the formula $\Box p \to p$. It defines reflexivity. Reflexivity is of course first-order definable by the sentence $Q(x, x)$. But it is also definable by the monadic second-order sentence

$$\forall X \, \forall x \, (\forall y \, (Q(x, y) \to X(y)) \to X(x)).$$

This means, of course, that the two sentences are equivalent. Here's how you might convince yourself of this directly: First suppose the second-order sentence is true in a structure $M$. Since $x$ and $X$ is universally quantified, the remainder must hold for any $x \in W$ and set $X \subseteq W$, e.g, the set $\{z : Rxz\}$ where $R = Q^{\mathfrak{M}}$. So, for any $s$ with $s(x) \in W$ and $s(X) = \{z : Rzx\}$ we have $\mathfrak{M} \vDash \forall y \, (Q(x, y) \to X(y)) \to X(x)$. But by the way we've picked $s(X)$ that means $\mathfrak{M}, s \vDash \forall y \, (Q(x, y) \to Q(x, y)) \to Q(x, x)$, which is equivalent to $Q(x, x)$ since the antecedent is valid. Since $s(x)$ is arbitrary, we have $\mathfrak{M} \vDash \forall x \, Q(x, x)$.

Now suppose that $\mathfrak{M} \vDash Q(x, x)$ and show that $\mathfrak{M} \vDash \forall X \, \forall x \, (\forall y \, (Q(x, y) \to X(y)) \to X(x))$. Pick any assignment $s$, and assume $\mathfrak{M}, s \vDash \forall y \, (Q(x, y) \to X(y))$. Let $s'$ be the $y$-variant of $s$ with $s'(y) = x$; we have $\mathfrak{M}, s' \vDash Q(x, y) \to X(y))$, i.e., $\mathfrak{M}, s \vDash Q(x, x) \to X(x))$. Since $\mathfrak{M} \vDash \forall x \, Q(x, x)$, the antecedent is true, and we have $\mathfrak{M}, s \vDash X(x)$, which is what we needed to show.

Since some definable classes of frames are not first-order definable, not every monadic-second order sentence of the form $\varphi'$ is equivalent to a first-order sentence. There is no effective method to decide which ones are.

## Problems

**Problem 39.1.** Complete the proof of Theorem 39.1

**Problem 39.2.** Prove the claims in Proposition 39.2.

**Problem 39.3.** Let $\mathfrak{M} = \langle W, R, V \rangle$ be a model. Show that if $R$ satisfies the left-hand properties of table 39.2, every instance of the corresponding right-hand formula is true in $\mathfrak{M}$.

**Problem 39.4.** Show that if the formula on the right side of table 39.2 is valid in a frame $\mathfrak{F}$, then $\mathfrak{F}$ has the property on the left side. To do this, consider a frame that does *not* satisfy the property on the left, and define a suitable $V$ such that the formula on the right is false at some world.

**Problem 39.5.** Prove Proposition 39.9.

**Problem 39.6.** Prove Proposition 39.12 by showing:

1. If $R$ is symmetric and transitive, it is euclidean.
2. If $R$ is reflexive, it is serial.
3. If $R$ is reflexive and euclidean, it is symmetric.
4. If $R$ is symmetric and euclidean, it is transitive.
5. If $R$ is serial, symmetric, and transitive, it is reflexive.

Explain why this suffices for the proof that the conditions are equivalent.

# Chapter 40

# Axiomatic Derivations

## 40.1 Introduction

We have a semantics for the basic modal language in terms of modal models, and a notion of a formula being valid—true at all worlds in all models—or valid with respect to some class of models or frames—true at all worlds in all models in the class, or based on the frame. Logic usually connects such semantic characterizations of validity with a proof-theoretic notion of derivability. The aim is to define a notion of derivability in some system such that a formula is derivable iff it is valid.

The simplest and historically oldest derivation systems are so-called Hilbert-type or axiomatic derivation systems. Hilbert-type derivation systems for many modal logics are relatively easy to construct: they are simple as objects of metatheoretical study (e.g., to prove soundness and completeness). However, they are much harder to use to prove formulas in than, say, natural deduction systems.

In Hilbert-type derivation systems, a derivation of a formula is a sequence of formulas leading from certain axioms, via a handful of inference rules, to the formula in question. Since we want the derivation system to match the semantics, we have to guarantee that the set of derivable formulas are true in all models (or true in all models in which all axioms are true). We'll first isolate some properties of modal logics that are necessary for this to work: the "normal" modal logics. For normal modal logics, there are only two inference rules that need to be assumed: modus ponens and necessitation. As axioms we take all (substitution instances) of tautologies, and, depending on the modal logic we deal with, a number of modal axioms. Even if we are just interested in the class of all models, we must also count all substitution instances of K and Dual as axioms. This alone generates the minimal normal modal logic **K**.

**Definition 40.1.** The rule of *modus ponens* is the inference schema

$$\frac{\varphi \qquad \varphi \to \psi}{\psi}\ \text{MP}$$

We say a formula $\psi$ *follows from* formulas $\varphi, \chi$ by modus ponens iff $\chi \equiv \varphi \to \psi$.

**Definition 40.2.** The rule of *necessitation* is the inference schema

$$\frac{\varphi}{\Box\varphi}\ \text{NEC}$$

We say the formula $\psi$ follows from the formulas $\varphi$ by necessitation iff $\psi \equiv \Box\varphi$.

**Definition 40.3.** A *derivation* from a set of axioms $\Sigma$ is a sequence of formulas $\psi_1, \psi_2, \ldots, \psi_n$, where each $\psi_i$ is either

1. a substitution instance of a tautology, or
2. a substitution instance of a formula in $\Sigma$, or
3. follows from two formulas $\psi_j, \psi_k$ with $j, k < i$ by modus ponens, or
4. follows from a formula $\psi_j$ with $j < i$ by necessitation.

If there is such a derivation with $\psi_n \equiv \varphi$, we say that $\varphi$ is *derivable from* $\Sigma$, in symbols $\Sigma \vdash \varphi$.

With this definition, it will turn out that the set of derivable formulas forms a normal modal logic, and that any derivable formula is true in every model in which every axiom is true. This property of derivations is called *soundness*. The converse, *completeness*, is harder to prove.

## 40.2 Normal Modal Logics

Not every set of modal formulas can easily be characterized as those formulas derivable from a set of axioms. We want modal logics to be well-behaved. First of all, everything we can derive in classical propositional logic should still be derivable, of course taking into account that the formulas may now contain also $\Box$ and $\Diamond$. To this end, we require that a modal logic contain all tautological instances and be closed under modus ponens.

**Definition 40.4.** A *modal logic* is a set $\Sigma$ of modal formulas which

1. contains all tautologies, and
2. is closed under substitution, i.e., if $\varphi \in \Sigma$, and $\theta_1, \ldots, \theta_n$ are formulas, then
$$\varphi[\theta_1/p_1, \ldots, \theta_n/p_n] \in \Sigma,$$
3. is closed under *modus ponens*, i.e., if $\varphi$ and $\varphi \to \psi \in \Sigma$, then $\psi \in \Sigma$.

In order to use the relational semantics for modal logics, we also have to require that all formulas valid in all modal models are included. It turns out that this requirement is met as soon as all instances of K and DUAL are derivable, and whenever a formula $\varphi$ is derivable, so is $\Box\varphi$. A modal logic that satisfies these conditions is called *normal*. (Of course, there are also non-normal modal logics, but the usual relational models are not adequate for them.)

**Definition 40.5.** A modal logic $\Sigma$ is *normal* if it contains

$$\Box(p \to q) \to (\Box p \to \Box q), \tag{K}$$
$$\Diamond p \leftrightarrow \neg\Box\neg p \tag{DUAL}$$

and is closed under *necessitation*, i.e., if $\varphi \in \Sigma$, then $\Box\varphi \in \Sigma$.

Observe that while tautological implication is "fine-grained" enough to preserve *truth at a world*, the rule NEC only preserves *truth in a model* (and hence also validity in a frame or in a class of frames).

**Proposition 40.6.** *Every normal modal logic is closed under rule* RK*,*

$$\frac{\varphi_1 \to (\varphi_2 \to \cdots (\varphi_{n-1} \to \varphi_n)\cdots)}{\Box\varphi_1 \to (\Box\varphi_2 \to \cdots (\Box\varphi_{n-1} \to \Box\varphi_n)\cdots).}\ \text{RK}$$

*Proof.* By induction on $n$: If $n = 1$, then the rule is just NEC, and every normal modal logic is closed under NEC.

Now suppose the result holds for $n-1$; we show it holds for $n$.

Assume

$$\varphi_1 \to (\varphi_2 \to \cdots (\varphi_{n-1} \to \varphi_n)\cdots) \in \Sigma$$

By the induction hypothesis, we have

$$\Box\varphi_1 \to (\Box\varphi_2 \to \cdots \Box(\varphi_{n-1} \to \varphi_n)\cdots) \in \Sigma$$

Since $\Sigma$ is a normal modal logic, it contains all instances of K, in particular

$$\Box(\varphi_{n-1} \to \varphi_n) \to (\Box\varphi_{n-1} \to \Box\varphi_n) \in \Sigma$$

Using modus ponens and suitable tautological instances we get

$$\Box\varphi_1 \to (\Box\varphi_2 \to \cdots (\Box\varphi_{n-1} \to \Box\varphi_n)\cdots) \in \Sigma. \qquad \Box$$

**Proposition 40.7.** *Every normal modal logic $\Sigma$ contains $\neg\Diamond\bot$.*

**Proposition 40.8.** *Let $\varphi_1, \ldots, \varphi_n$ be formulas. Then there is a smallest modal logic $\Sigma$ containing all instances of $\varphi_1, \ldots, \varphi_n$.*

*Proof.* Given $\varphi_1, \ldots, \varphi_n$, define $\Sigma$ as the intersection of all normal modal logics containing all instances of $\varphi_1, \ldots, \varphi_n$. The intersection is non-empty as $\mathrm{Frm}(\mathcal{L})$, the set of all formulas, is such a modal logic. □

**Definition 40.9.** The smallest normal modal logic containing $\varphi_1, \ldots, \varphi_n$ is called a *modal system* and denoted by $\mathbf{K}\varphi_1 \ldots \varphi_n$. The smallest normal modal logic is denoted by **K**.

## 40.3 Derivations and Modal Systems

We first define what a derivation is for normal modal logics. Roughly, a derivation is a sequence of formulas in which every element is either (a substitution instance of) one of a number of *axioms*, or follows from previous elements by one of a few inference rules. For normal modal logics, all instances of tautologies, K, and DUAL count as axioms. This results in the modal system **K**, the smallest normal modal logic. We may wish to add additional axioms to obtain other systems, however. The rules are always modus ponens MP and necessitation NEC.

**Definition 40.10.** Given a modal system $\mathbf{K}\varphi_1 \ldots \varphi_n$ and a formula $\psi$ we say that $\psi$ is *derivable* in $\mathbf{K}\varphi_1 \ldots \varphi_n$, written $\mathbf{K}\varphi_1 \ldots \varphi_n \vdash \psi$, if and only if there are formulas $\chi_1, \ldots, \chi_k$ such that $\chi_k = \psi$ and each $\chi_i$ is either a tautological instance, or an instance of one of K, DUAL, $\varphi_1, \ldots, \varphi_n$, or it follows from previous formulas by means of the rules MP or NEC.

The following proposition allows us to show that $\psi \in \Sigma$ by exhibiting a $\Sigma$-proof of $\psi$.

**Proposition 40.11.** $\mathbf{K}\varphi_1 \ldots \varphi_n = \{\psi : \mathbf{K}\varphi_1 \ldots \varphi_n \vdash \psi\}$.

*Proof.* We use induction on the length of derivations to show that $\{\psi : \mathbf{K}\varphi_1 \ldots \varphi_n \vdash \psi\} \subseteq \mathbf{K}\varphi_1 \ldots \varphi_n$.

If the derivation of $\psi$ has length 1, it contains a single formula. That formula cannot follow from previous formulas by MP or NEC, so must be a tautological instance, an instance of K, DUAL, or an instance of one of $\varphi_1, \ldots, \varphi_n$. But $\mathbf{K}\varphi_1 \ldots \varphi_n$ contains these as well, so $\psi \in \mathbf{K}\varphi_1 \ldots \varphi_n$.

If the derivation of $\psi$ has length $> 1$, then $\psi$ may in addition be obtained by MP or NEC from formulas not occurring as the last line in the derivation. If $\psi$ follows from $\chi$ and $\chi \to \psi$ (by MP), then $\chi$ and $\chi \to \psi \in \mathbf{K}\varphi_1 \ldots \varphi_n$ by induction hypothesis. But every modal logic is closed under modus ponens, so $\psi \in \mathbf{K}\varphi_1 \ldots \varphi_n$. If $\psi \equiv \Box\chi$ follows from $\chi$ by NEC, then $\chi \in \mathbf{K}\varphi_1 \ldots \varphi_n$ by induction hypothesis. But every normal modal logic is closed under NEC, so $\psi \in \mathbf{K}\varphi_1 \ldots \varphi_n$.

The converse inclusion follows by showing that $\Sigma = \{\psi : \mathbf{K}\varphi_1 \ldots \varphi_n \vdash \psi\}$ is a normal modal logic containing all the instances of $\varphi_1, \ldots, \varphi_n$, and the observation that $\mathbf{K}\varphi_1 \ldots \varphi_n$ is, by definition, the smallest such logic.

1. Every tautology $\psi$ is a tautological instance, so $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \psi$, so $\Sigma$ contains all tautologies.

2. If $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \chi$ and $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \chi \to \psi$, then $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \psi$: Combine the derivation of $\chi$ with that of $\chi \to \psi$, and add the line $\psi$. The last line is justified by MP. So $\Sigma$ is closed under modus ponens.

3. If $\psi$ has a derivation, then every substitution instance of $\psi$ also has a derivation: apply the substitution to every formula in the derivation. (Exercise: prove by induction on the length of derivations that the result is also a correct derivation). So $\Sigma$ is closed under uniform substitution. (We have now established that $\Sigma$ satisfies all conditions of a modal logic.)

4. We have $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \mathrm{K}$, so $K \in \Sigma$.

5. We have $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \mathrm{DUAL}$, so $\mathrm{DUAL} \in \Sigma$.

6. If $\mathbf{K}\varphi_1 \dots \varphi_n \vdash \chi$, the additional line $\Box\chi$ is justified by NEC. Consequently, $\Sigma$ is closed under NEC. Thus, $\Sigma$ is normal.

□

## 40.4 Proofs in K

In order to practice proofs in the smallest modal system, we show the valid formulas on the left-hand side of table 38.1 can all be given **K**-proofs.

**Proposition 40.12.** $\mathbf{K} \vdash \Box\varphi \to \Box(\psi \to \varphi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\varphi \to (\psi \to \varphi)$ | TAUT |
| 2. | $\Box(\varphi \to (\psi \to \varphi))$ | NEC, 1 |
| 3. | $\Box(\varphi \to (\psi \to \varphi)) \to (\Box\varphi \to \Box(\psi \to \varphi))$ | K |
| 4. | $\Box\varphi \to \Box(\psi \to \varphi)$ | MP, 2, 3 |

□

**Proposition 40.13.** $\mathbf{K} \vdash \Box(\varphi \wedge \psi) \to (\Box\varphi \wedge \Box\psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $(\varphi \wedge \psi) \to \varphi$ | TAUT |
| 2. | $\Box((\varphi \wedge \psi) \to \varphi)$ | NEC |
| 3. | $\Box((\varphi \wedge \psi) \to \varphi) \to (\Box(\varphi \wedge \psi) \to \Box\varphi)$ | K |
| 4. | $\Box(\varphi \wedge \psi) \to \Box\varphi$ | MP, 2, 3 |
| 5. | $(\varphi \wedge \psi) \to \psi$ | TAUT |
| 6. | $\Box((\varphi \wedge \psi) \to \psi)$ | NEC |
| 7. | $\Box((\varphi \wedge \psi) \to \psi) \to (\Box(\varphi \wedge \psi) \to \Box\psi)$ | K |
| 8. | $\Box(\varphi \wedge \psi) \to \Box\psi$ | MP, 6, 7 |
| 9. | $(\Box(\varphi \wedge \psi) \to \Box\varphi) \to$ $((\Box(\varphi \wedge \psi) \to \Box\varphi) \to$ $(\Box(\varphi \wedge \psi) \to (\Box\varphi \wedge \Box\psi)))$ | TAUT |
| 10. | $(\Box(\varphi \wedge \psi) \to \Box\varphi) \to$ $(\Box(\varphi \wedge \psi) \to (\Box\varphi \wedge \Box\psi))$ | MP, 4, 9 |
| 11. | $\Box(\varphi \wedge \psi) \to (\Box\varphi \wedge \Box\psi)$ | MP, 4, 10. |

Note that the formula on line 9 is an instance of the tautology

$$(p \to q) \to ((p \to r) \to (p \to (q \wedge r))).$$ □

**Proposition 40.14.** $\mathbf{K} \vdash (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\varphi \to (\psi \to (\varphi \wedge \psi))$ | TAUT |
| 2. | $\Box(\varphi \to (\psi \to (\varphi \wedge \psi)))$ | NEC, 1 |
| 3. | $\Box(\varphi \to (\psi \to (\varphi \wedge \psi))) \to (\Box\varphi \to \Box(\psi \to (\varphi \wedge \psi)))$ | K |
| 4. | $\Box\varphi \to \Box(\psi \to (\varphi \wedge \psi))$ | MP, 2, 3 |
| 5. | $\Box(\psi \to (\varphi \wedge \psi)) \to (\Box\psi \to \Box(\varphi \wedge \psi))$ | K |
| 6. | $(\Box\varphi \to \Box(\psi \to (\varphi \wedge \psi))) \to$ $(\Box(\psi \to (\varphi \wedge \psi)) \to (\Box\psi \to \Box(\varphi \wedge \psi))) \to$ $(\Box\varphi \to (\Box\psi \to \Box(\varphi \wedge \psi))))$ | TAUT |
| 7. | $(\Box(\psi \to (\varphi \wedge \psi)) \to (\Box\psi \to \Box(\varphi \wedge \psi))) \to$ $(\Box\varphi \to (\Box\psi \to \Box(\varphi \wedge \psi)))$ | MP, 4, 6 |
| 8. | $\Box\varphi \to (\Box\psi \to \Box(\varphi \wedge \psi)))$ | MP, 5, 7 |
| 9. | $(\Box\varphi \to (\Box\psi \to \Box(\varphi \wedge \psi)))) \to$ $((\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi))$ | TAUT |
| 10. | $(\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi)$ | MP, 8, 9 |

The formulas on lines 6 and 9 are instances of the tautologies

$$(p \to q) \to ((q \to r) \to (p \to r))$$
$$(p \to (q \to r)) \to ((p \wedge q) \to r)$$

□

**Proposition 40.15.** $\mathbf{K} \vdash \neg\Box p \to \Diamond\neg p$

*Proof.*

| | | |
|---|---|---|
| 1. | $\Diamond\neg p \leftrightarrow \neg\Box\neg\neg p$ | DUAL |
| 2. | $(\Diamond\neg p \leftrightarrow \neg\Box\neg\neg p) \rightarrow$ $(\neg\Box\neg\neg p \rightarrow \Diamond\neg p)$ | TAUT |
| 3. | $\neg\Box\neg\neg p \rightarrow \Diamond\neg p$ | MP, 1, 2 |
| 4. | $\neg\neg p \rightarrow p$ | TAUT |
| 5. | $\Box(\neg\neg p \rightarrow p)$ | NEC, 4 |
| 6. | $\Box(\neg\neg p \rightarrow p) \rightarrow (\Box\neg\neg p \rightarrow \Box p)$ | K |
| 7. | $(\Box\neg\neg p \rightarrow \Box p)$ | MP, 5, 6 |
| 8. | $(\Box\neg\neg p \rightarrow \Box p) \rightarrow (\neg\Box p \rightarrow \neg\Box\neg\neg p)$ | TAUT |
| 9. | $\neg\Box p \rightarrow \neg\Box\neg\neg p$ | MP, 7, 8 |
| 10. | $(\neg\Box p \rightarrow \neg\Box\neg\neg p) \rightarrow$ $((\neg\Box\neg\neg p \rightarrow \Diamond\neg p) \rightarrow (\neg\Box p \rightarrow \Diamond\neg p))$ | TAUT |
| 11. | $(\neg\Box\neg\neg p \rightarrow \Diamond\neg p) \rightarrow (\neg\Box p \rightarrow \Diamond\neg p)$ | MP, 9, 10 |
| 12. | $\neg\Box p \rightarrow \Diamond\neg p$ | MP, 3, 11 |

The formulas on lines 8 and 10 are instances of the tautologies

$$(p \rightarrow q) \rightarrow (\neg q \rightarrow \neg p)$$
$$(p \rightarrow q) \rightarrow ((q \rightarrow r) \rightarrow (p \rightarrow r)).$$

□

## 40.5 Derived Rules

Finding and writing derivations is obviously difficult, cumbersome, and repetitive. For instance, very often we want to pass from $\varphi \rightarrow \psi$ to $\Box\varphi \rightarrow \Box\psi$, i.e., apply rule RK. That requires an application of NEC, then recording the proper instance of K, then applying MP. Passing from $\varphi \rightarrow \psi$ and $\psi \rightarrow \chi$ to $\varphi \rightarrow \chi$ requires recording the (long) tautological instance

$$(\varphi \rightarrow \psi) \rightarrow ((\psi \rightarrow \chi) \rightarrow (\varphi \rightarrow \chi))$$

and applying MP twice. Often we want to replace a sub-formula by a formula we know to be equivalent, e.g., $\Diamond\varphi$ by $\neg\Box\neg\varphi$, or $\neg\neg\varphi$ by $\varphi$. So rather than write out the actual derivation, it is more convenient to simply record why the intermediate steps are derivable. For this purpose, let us collect some facts about derivability.

**Proposition 40.16.** *If* $\mathbf{K} \vdash \varphi_1, \ldots, \mathbf{K} \vdash \varphi_n$, *and* $\psi$ *follows from* $\varphi_1, \ldots, \varphi_n$ *by propositional logic, then* $\mathbf{K} \vdash \psi$.

*Proof.* If $\psi$ follows from $\varphi_1, \ldots, \varphi_n$ by propositional logic, then

$$\varphi_1 \rightarrow (\varphi_2 \rightarrow \cdots (\varphi_n \rightarrow \psi) \ldots)$$

is a tautological instance. Applying MP $n$ times gives a derivation of $\psi$. □

We will indicate use of this proposition by PL.

**Proposition 40.17.** *If* $\mathbf{K} \vdash \varphi_1 \to (\varphi_2 \to \cdots (\varphi_{n-1} \to \varphi_n) \dots)$ *then* $\mathbf{K} \vdash \Box\varphi_1 \to (\Box\varphi_2 \to \cdots (\Box\varphi_{n-1} \to \Box\varphi_n) \dots)$.

*Proof.* By induction on $n$, just as in the proof of Proposition 40.6. □

We will indicate use of this proposition by RK. Let's illustrate how these results help establishing derivability results more easily.

**Proposition 40.18.** $\mathbf{K} \vdash (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\mathbf{K} \vdash \varphi \to (\psi \to (\varphi \wedge \psi))$ | TAUT |
| 2. | $\mathbf{K} \vdash \Box\varphi \to (\Box\psi \to \Box(\varphi \wedge \psi)))$ | RK, 1 |
| 3. | $\mathbf{K} \vdash (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi)$ | PL, 2 |

□

**Proposition 40.19.** *If* $\mathbf{K} \vdash \varphi \leftrightarrow \psi$ *and* $\mathbf{K} \vdash \chi[\varphi/p]$ *then* $\mathbf{K} \vdash \chi[B/p]$

*Proof.* Exercise. □

This proposition comes in handy especially when we want to convert $\Diamond$ into $\Box$ (or vice versa), or remove double negations inside a formula. For instance:

**Proposition 40.20.** $\mathbf{K} \vdash \neg\Box p \to \Diamond\neg p$

*Proof.*

| | | |
|---|---|---|
| 1. | $\mathbf{K} \vdash \Diamond\neg p \leftrightarrow \neg\Box\neg\neg p$ | DUAL |
| 2. | $\mathbf{K} \vdash \neg\Box\neg\neg p \to \Diamond\neg p$ | PL, 1 |
| 3. | $\mathbf{K} \vdash \neg\Box p \to \Diamond\neg p$ | re-write $p$ for $\neg\neg p$ |

□

The following proposition justifies that we can establish derivability results schematically. E.g., the previous proposition does not just establish that $\mathbf{K} \vdash \neg\Box p \to \Diamond\neg p$, but $\mathbf{K} \vdash \neg\Box\varphi \to \Diamond\neg\varphi$ for arbitrary $\varphi$.

**Proposition 40.21.** *If* $\varphi$ *is a substitution instance of* $\psi$ *and* $\mathbf{K} \vdash \psi$, *then* $\mathbf{K} \vdash \varphi$.

*Proof.* It is tedious but routine to verify (by induction on the length of the derivation of $\psi$) that applying a substitution to an entire derivation also results in a correct derivation. Specifically, substitution instances of tautological instances are themselves tautological instances, substitution instances of instances of DUAL and K are themselves instances of DUAL and K, and applications of MP and NEC remain correct when substituting formulas for propositional variables in both premise(s) and conclusion. □

## 40.6 More Proofs in K

Let's see some more examples of derivability in **K**, now using the simplified method introduced in section 40.5.

**Proposition 40.22.** $\mathbf{K} \vdash \Box(\varphi \to \psi) \to (\Diamond\varphi \to \Diamond\psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\mathbf{K} \vdash (\varphi \to \psi) \to (\neg\psi \to \neg\varphi)$ | PL |
| 2. | $\mathbf{K} \vdash \Box(\varphi \to \psi) \to (\Box\neg\psi \to \Box\neg\varphi)$ | RK, 1 |
| 3. | $\mathbf{K} \vdash (\Box\neg\psi \to \Box\neg\varphi) \to (\neg\Box\neg\varphi \to \neg\Box\neg\psi)$ | TAUT |
| 4. | $\mathbf{K} \vdash (\Box\neg\psi \to \Box\neg\varphi) \to (\neg\Box\neg\varphi \to \neg\Box\neg\psi)$ | PL, 2, 3 |
| 5. | $\mathbf{K} \vdash \Box(\varphi \to \psi) \to (\Diamond\varphi \to \Diamond\psi)$ | re-writing $\Diamond$ for $\neg\Box\neg$. |

□

**Proposition 40.23.** $\mathbf{K} \vdash \Box\varphi \to (\Diamond(\varphi \to \psi) \to \Diamond\psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\mathbf{K} \vdash \varphi \to (\neg\psi \to \neg(\varphi \to \psi))$ | TAUT |
| 2. | $\mathbf{K} \vdash \Box\varphi \to (\Box\neg\psi \to \Box\neg(\varphi \to \psi))$ | RK, 1 |
| 3. | $\mathbf{K} \vdash \Box\varphi \to (\neg\Box\neg(\varphi \to \psi) \to \neg\Box\neg\psi)$ | PL, 2 |
| 4. | $\mathbf{K} \vdash \Box\varphi \to (\Diamond(\varphi \to \psi) \to \Diamond\psi)$ | re-writing $\Diamond$ for $\neg\Box\neg$. |

□

**Proposition 40.24.** $\mathbf{K} \vdash (\Diamond\varphi \vee \Diamond\psi) \to \Diamond(\varphi \vee \psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\mathbf{K} \vdash \neg(\varphi \vee \psi) \to \neg\varphi$ | TAUT |
| 2. | $\mathbf{K} \vdash \Box\neg(\varphi \vee \psi) \to \Box\neg\varphi$ | RK, 1 |
| 3. | $\mathbf{K} \vdash \neg\Box\neg\varphi \to \neg\Box\neg(\varphi \vee \psi)$ | PL, 2 |
| 4. | $\mathbf{K} \vdash \Diamond\varphi \to \Diamond(\varphi \vee \psi)$ | re-writing |
| 5. | $\mathbf{K} \vdash \Diamond\psi \to \Diamond(\varphi \vee \psi)$ | similarly |
| 6. | $\mathbf{K} \vdash (\Diamond\varphi \vee \Diamond\psi) \to \Diamond(\varphi \vee \psi)$ | PL, 4, 5. |

□

**Proposition 40.25.** $\mathbf{K} \vdash \Diamond(\varphi \vee \psi) \to (\Diamond\varphi \vee \Diamond\psi)$

*Proof.*

| | | |
|---|---|---|
| 1. | $\mathbf{K} \vdash \neg\varphi \to (\neg\psi \to \neg(\varphi \vee \psi)$ | TAUT |
| 2. | $\mathbf{K} \vdash \Box\neg\varphi \to (\Box\neg\psi \to \Box\neg(\varphi \vee \psi)$ | RK |
| 3. | $\mathbf{K} \vdash \Box\neg\varphi \to (\neg\Box\neg(\varphi \vee \psi) \to \neg\Box\neg\psi))$ | PL, 2 |
| 4. | $\mathbf{K} \vdash \neg\Box\neg(\varphi \vee \psi) \to (\Box\neg\varphi \to \neg\Box\neg\psi)$ | PL, 3 |
| 5. | $\mathbf{K} \vdash \neg\Box\neg(\varphi \vee \psi) \to (\neg\neg\Box\neg\psi \to \neg\Box\neg\varphi)$ | PL, 4 |
| 6. | $\mathbf{K} \vdash \Diamond(\varphi \vee \psi) \to (\neg\Diamond\psi \to \Diamond\varphi)$ | re-writing $\Diamond$ for $\neg\Box\neg$ |
| 7. | $\mathbf{K} \vdash \Diamond(\varphi \vee \psi) \to (\Diamond\psi \vee \Diamond\varphi)$ | PL, 6. |

□

## 40.7 Dual Formulas

**Definition 40.26.** Each of the formulas T, B, 4, and 5 has a *dual*, denoted by a subscripted diamond, as follows:

$$p \to \Diamond p \tag{$T_\Diamond$}$$
$$\Diamond\Box p \to p \tag{$B_\Diamond$}$$
$$\Diamond\Diamond p \to \Diamond p \tag{$4_\Diamond$}$$
$$\Diamond\Box p \to \Box p \tag{$5_\Diamond$}$$

Each of the above dual formulas is obtained from the corresponding formula by substituting $\neg p$ for $p$, contraposing, replacing $\neg\Box\neg$ by $\Diamond$, and replacing $\neg\Diamond\neg$ by $\Box$. D, i.e., $\Box\varphi \to \Diamond\varphi$ is its own dual in that sense.

## 40.8 Proofs in Modal Systems

We now come to proofs in systems of modal logic other than **K**.

**Proposition 40.27.** *The following provability results obtain:*

1. **KT5** $\vdash$ B*;*
2. **KT5** $\vdash$ 4*;*
3. **KDB4** $\vdash$ T*;*
4. **KB4** $\vdash$ 5*;*
5. **KB5** $\vdash$ 4*;*
6. **KT** $\vdash$ D*.*

*Proof.* We exhibit proofs for each.

1. **KT5** $\vdash$ B:

1. **KT5** $\vdash \Diamond\varphi \to \Box\Diamond\varphi$ 5
2. **KT5** $\vdash \varphi \to \Diamond\varphi$ $T_\Diamond$
3. **KT5** $\vdash \varphi \to \Box\Diamond\varphi$ PL.

2. **KT5** $\vdash$ 4:

1. **KT5** $\vdash \Diamond\Box\varphi \to \Box\Diamond\Box\varphi$ 5 with $\Box\varphi$ for $p$
2. **KT5** $\vdash \Box\varphi \to \Diamond\Box\varphi$ $T_\Diamond$ with $\Box\varphi$ for $p$
3. **KT5** $\vdash \Box\varphi \to \Box\Diamond\Box\varphi$ PL, 1, 2
4. **KT5** $\vdash \Diamond\Box\varphi \to \Box\varphi$ $5_\Diamond$
5. **KT5** $\vdash \Box\Diamond\Box\varphi \to \Box\Box\varphi$ RK, 4
6. **KT5** $\vdash \Box\varphi \to \Box\Box\varphi$ PL, 3, 5.

3. **KDB4** $\vdash$ T:

1. **KDB4** $\vdash \Diamond\Box\varphi \to \varphi$ $B_\Diamond$
2. **KDB4** $\vdash \Box\Box\varphi \to \Diamond\Box\varphi$ D with $\Box\varphi$ for $p$
3. **KDB4** $\vdash \Box\Box\varphi \to \varphi$ PL1, 2
4. **KDB4** $\vdash \Box\varphi \to \Box\Box\varphi$ 4
5. **KDB4** $\vdash \Box\varphi \to \varphi$ PL, 1, 4.

4. **KB4** $\vdash$ 5:

1. **KB4** $\vdash \Diamond\varphi \to \Box\Diamond\Diamond\varphi$ B with $\Diamond\varphi$ for $p$
2. **KB4** $\vdash \Diamond\Diamond\varphi \to \Diamond\varphi$ $4_\Diamond$
3. **KB4** $\vdash \Box\Diamond\Diamond\varphi \to \Box\Diamond\varphi$ RK, 2
4. **KB4** $\vdash \Diamond\varphi \to \Box\Diamond\varphi$ PL, 1, 3.

5. **KB5** $\vdash$ 4:

1. **KB5** $\vdash \Box\varphi \to \Box\Diamond\Box\varphi$ B with $\Box\varphi$ for $p$
2. **KB5** $\vdash \Diamond\Box\varphi \to \Box\varphi$ $5_\Diamond$
3. **KB5** $\vdash \Box\Diamond\Box\varphi \to \Box\Box\varphi$ RK, 2
4. **KB5** $\vdash \Box\varphi \to \Box\Box\varphi$ PL, 1, 3.

6. **KT** $\vdash$ D:

1. **KT** $\vdash \Box\varphi \to \varphi$ T
2. **KT** $\vdash \varphi \to \Diamond\varphi$ $T_\Diamond$
3. **KT** $\vdash \Box\varphi \to \Diamond\varphi$ PL, 1, 2

□

**Definition 40.28.** Following tradition, we define **S4** to be the system **KT4**, and **S5** the system **KTB4**.

The following proposition shows that the classical system **S5** has several equivalent axiomatizations. This should not surprise, as the various combinations of axioms all characterize equivalence relations (see Proposition 39.12).

**Proposition 40.29.** $\mathbf{KTB4} = \mathbf{KT5} = \mathbf{KDB4} = \mathbf{KDB5}$.

*Proof.* Exercise. □

## 40.9 Soundness

A derivation system is called sound if everything that can be derived is valid. When considering modal systems, i.e., derivations where in addition to K we can use instances of some formulas $\varphi_1, \ldots, \varphi_n$, we want every derivable formula to be true in any model in which $\varphi_1, \ldots, \varphi_n$ are true.

**Theorem 40.30** (Soundness Theorem)**.** *If every instance of $\varphi_1, \ldots, \varphi_n$ is valid in the classes of models $\mathcal{C}_1, \ldots, \mathcal{C}_n$, respectively, then $\mathbf{K}\varphi_1 \ldots \varphi_n \vdash \psi$ implies that $\psi$ is valid in the class of models $\mathcal{C}_1 \cap \cdots \cap \mathcal{C}_n$.*

*Proof.* By induction on length of proofs. For brevity, put $\mathcal{C} = \mathcal{C}_n \cap \cdots \cap \mathcal{C}_n$.

1. Induction Basis: If $\psi$ has a proof of length 1, then it is either a tautological instance, an instance of K, or of DUAL, or an instance of one of $\varphi_1, \ldots, \varphi_n$. In the first case, $\psi$ is valid in $\mathcal{C}$, since tautological instance are valid in *any* class of models, by Proposition 38.15. Similarly in the second case, by Proposition 38.18 and Proposition 38.19. Finally in the third case, since $\psi$ is valid in $\mathcal{C}_i$ and $\mathcal{C} \subseteq \mathcal{C}_i$, we have that $\psi$ is valid in $\mathcal{C}$ as well.

2. Inductive step: Suppose $\psi$ has a proof of length $k > 1$. If $\psi$ is a tautological instance or an instance of one of $\varphi_1, \ldots, \varphi_n$, we proceed as in the previous step. So suppose $\psi$ is obtained by MP from previous formulas $\chi \to \psi$ and $\chi$. Then $\chi \to \psi$ and $\chi$ have proofs of length $< k$, and by inductive hypothesis they are valid in $\mathcal{C}$. By Proposition 38.20, $\psi$ is valid in $\mathcal{C}$ as well. Finally suppose $\psi$ is obtained by NEC from $\chi$ (so that $\psi = \Box\chi$). By inductive hypothesis, $\chi$ is valid in $\mathcal{C}$, and by Proposition 38.12 so is $\psi$. □

## 40.10 Showing Systems are Distinct

In section 40.8 we saw how to prove that two systems of modal logic are in fact the same system. Theorem 40.30 allows us to show that two modal systems $\Sigma$ and $\Sigma'$ are distinct, by finding a formula $\varphi$ such that $\Sigma' \vdash \varphi$ that fails in a model of $\Sigma$.

**Proposition 40.31.** $\mathbf{KD} \subsetneq \mathbf{KT}$

*Proof.* This is the syntactic counterpart to the semantic fact that all reflexive relations are serial. To show **KD** $\subseteq$ **KT** we need to see that **KD** $\vdash \psi$ implies **KT** $\vdash \psi$, which follows from **KT** $\vdash$ **D**, as shown in Proposition 40.27(6). To show that the inclusion is proper, by Soundness (Theorem 40.30), it suffices to exhibit a model of **KD** where T, i.e., $\Box p \to p$, fails (an easy task left as an exercise), for then by Soundness **KD** $\nvdash \Box p \to p$. □

**Proposition 40.32.** **KB** $\neq$ **K4**.

*Proof.* We construct a symmetric model where some instance of 4 fails; since obviously the instance is derivable for **K4** but not in **KB**, it will follow **K4** $\nsubseteq$ **KB**. Consider the symmetric model $\mathfrak{M}$ of Figure 40.1. Since the model is symmetric, K and B are true in $\mathfrak{M}$ (by Proposition 38.18 and Theorem 39.1, respectively). However, $\mathfrak{M}, w_1 \nVdash \Box p \to \Box\Box p$. □

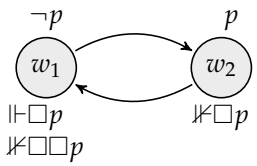



Figure 40.1: A symmetric model falsifying an instance of 4.

**Theorem 40.33.** **KTB** $\nvdash$ **4** *and* **KTB** $\nvdash$ **5**.

*Proof.* By Theorem 39.1 we know that all instances of T and B are true in every reflexive symmetric model (respectively). So by soundness, it suffices to find a reflexive symmetric model containing a world at which some instance of 4 fails, and similarly for 5. We use the same model for both claims. Consider the symmetric, reflexive model in Figure 40.2. Then $\mathfrak{M}, w_1 \nVdash \Box p \to \Box\Box p$, so 4 fails at $w_1$. Similarly, $\mathfrak{M}, w_2 \nVdash \Diamond\neg p \to \Box\Diamond\neg p$, so the instance of 5 with $\varphi = \neg p$ fails at $w_2$. □

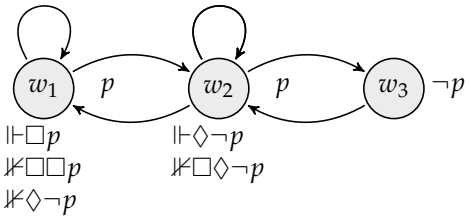



Figure 40.2: The model for Theorem 40.33.

**Theorem 40.34.** **KD5** $\neq$ **KT4** $=$ **S4**.

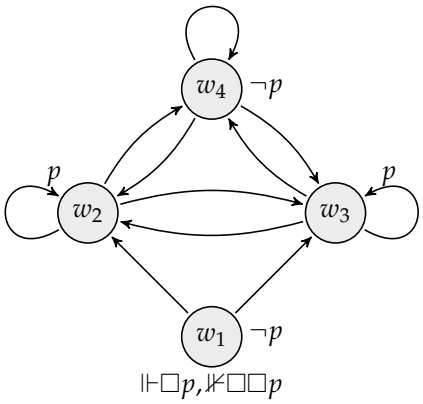



Figure 40.3: The model for Theorem 40.34.

*Proof.* By Theorem 39.1 we know that all instances of D and 5 are true in all serial euclidean models. So it suffices to find a serial euclidean model containing a world at which some instance of 4 fails. Consider the model of Figure 40.3, and notice that $\mathfrak{M}, w_1 \nVdash \Box p \to \Box\Box p$. □

## 40.11 Derivability from a Set of Formulas

In section 40.8 we defined a notion of provability of a formula in a system $\Sigma$. We now extend this notion to provability in $\Sigma$ from formulas in a set $\Gamma$.

**Definition 40.35.** A formula $\varphi$ is derivable in a system $\Sigma$ from a set of formulas $\Gamma$, written $\Gamma \vdash_\Sigma \varphi$ if and only if there are $\psi_1, \ldots, \psi_n \in \Gamma$ such that $\Sigma \vdash \psi_1 \to (\psi_2 \to \cdots (\psi_n \to \varphi) \cdots)$.

## 40.12 Properties of Derivability

**Proposition 40.36.** *Let $\Sigma$ be a modal system and $\Gamma$ a set of modal formulas. The following properties hold:*

1. Monotony*: If $\Gamma \vdash_\Sigma \varphi$ and $\Gamma \subseteq \Delta$ then $\Delta \vdash_\Sigma \varphi$;*
2. Reflexivity*: If $\varphi \in \Gamma$ then $\Gamma \vdash_\Sigma \varphi$;*
3. Cut*: If $\Gamma \vdash_\Sigma \varphi$ and $\Delta \cup \{\varphi\} \vdash_\Sigma \psi$ then $\Gamma \cup \Delta \vdash_\Sigma \psi$;*
4. Deduction theorem*: $\Gamma \cup \{\psi\} \vdash_\Sigma \varphi$ if and only if $\Gamma \vdash_\Sigma \psi \to \varphi$;*
5. Rule T*: If $\Gamma \vdash_\Sigma \varphi_1$ and ... and $\Gamma \vdash_\Sigma \varphi_n$ and $\varphi_1 \to (\varphi_2 \to \cdots (\varphi_n \to \psi) \cdots)$ is a tautological instance, then $\Gamma \vdash_\Sigma \psi$.*

The proof is an easy exercise. Part (5) of Proposition 40.36 gives us that, for instance, if $\Gamma \vdash_\Sigma \varphi \vee \psi$ and $\Gamma \vdash_\Sigma \neg\varphi$, then $\Gamma \vdash_\Sigma \psi$. Also, in what follows, we write $\Gamma, \varphi \vdash_\Sigma \psi$ instead of $\Gamma \cup \{\varphi\} \vdash_\Sigma \psi$.

**Definition 40.37.** A set $\Gamma$ is *deductively closed* relatively to a system $\Sigma$ if and only if $\Gamma \vdash_\Sigma \varphi$ implies $\varphi \in \Gamma$.

## 40.13 Consistency

Consistency is an important property of sets of formulas. A set of formulas is inconsistent if a contradiction, such as $\bot$, is derivable from it; and otherwise consistent. If a set is inconsistent, its formulas cannot all be true in a model at a world. For the completeness theorem we prove the converse: every consistent set is true at a world in a model, namely in the "canonical model."

**Definition 40.38.** A set $\Gamma$ is *consistent* relatively to a system $\Sigma$ or, as we will say, $\Sigma$-consistent, if and only if $\Gamma \nvdash_\Sigma \bot$.

So for instance, the set $\{\Box(p \to q), \Box p, \neg\Box q\}$ is consistent relatively to propositional logic, but not **K**-consistent. Similarly, the set $\{\Diamond p, \Box\Diamond p \to q, \neg q\}$ is not **K5**-consistent.

**Proposition 40.39.** *Let $\Gamma$ be a set of formulas. Then:*

1. *A set $\Gamma$ is $\Sigma$-consistent if and only if there is some formula $\varphi$ such that $\Gamma \nvdash_\Sigma \varphi$.*
2. *$\Gamma \vdash_\Sigma \varphi$ if and only if $\Gamma \cup \{\neg\varphi\}$ is not $\Sigma$-consistent.*
3. *If $\Gamma$ is $\Sigma$-consistent, then for any formula $\varphi$, either $\Gamma \cup \{\varphi\}$ is $\Sigma$-consistent or $\Gamma \cup \{\neg\varphi\}$ is $\Sigma$-consistent.*

*Proof.* These facts follow easily using classical propositional logic. We give the argument for (3). Proceed contrapositively and suppose neither $\Gamma \cup \{\varphi\}$ nor $\Gamma \cup \{\neg\varphi\}$ is $\Sigma$-consistent. Then by (2), both $\Gamma, \varphi \vdash_\Sigma \bot$ and $\Gamma, \neg\varphi \vdash_\Sigma \bot$. By the deduction theorem $\Gamma \vdash_\Sigma \varphi \to \bot$ and $\Gamma \vdash_\Sigma \neg\varphi \to \bot$. But $(\varphi \to \bot) \to ((\neg\varphi \to \bot) \to \bot)$ is a tautological instance, hence by Proposition 40.36(5), $\Gamma \vdash_\Sigma \bot$. □

## Problems

**Problem 40.1.** Prove Proposition 40.7.

**Problem 40.2.** Find derivations in **K** for the following formulas:

1. $\Box\neg p \to \Box(p \to q)$
2. $(\Box p \vee \Box q) \to \Box(p \vee q)$
3. $\Diamond p \to \Diamond(p \vee q)$

**Problem 40.3.** Prove Proposition 40.19 by proving, by induction on the complexity of $\chi$, that if $\mathbf{K} \vdash \varphi \leftrightarrow \psi$ then $\mathbf{K} \vdash \chi[\varphi/p] \leftrightarrow \chi[\psi/p]$.

**Problem 40.4.** Show that the following derivability claims hold:

1. $\mathbf{K} \vdash \Diamond\neg\bot \to (\Box\varphi \to \Diamond\varphi)$;
2. $\mathbf{K} \vdash \Box(\varphi \vee \psi) \to (\Diamond\varphi \vee \Box\psi)$;
3. $\mathbf{K} \vdash (\Diamond\varphi \to \Box\psi) \to \Box(\varphi \to \psi)$.

**Problem 40.5.** Show that for each formula $\varphi$ in Definition 40.26: $\mathbf{K} \vdash \varphi \leftrightarrow \varphi_\Diamond$.

**Problem 40.6.** Prove Proposition 40.29.

**Problem 40.7.** Give an alternative proof of Theorem 40.34 using a model with 3 worlds.

**Problem 40.8.** Provide a single reflexive transitive model showing that both $\mathbf{KT4} \nvdash \mathrm{B}$ and $\mathbf{KT4} \nvdash 5$.

# Chapter 41

# Completeness and Canonical Models

## 41.1 Introduction

If $\Sigma$ is a modal system, then the soundness theorem establishes that if $\Sigma \vdash \varphi$, then $\varphi$ is valid in any class $\mathcal{C}$ of models in which all instances of all formulas in $\Sigma$ are valid. In particular that means that if $\mathbf{K} \vdash \varphi$ then $\varphi$ is true in all models; if $\mathbf{KT} \vdash \varphi$ then $\varphi$ is true in all reflexive models; if $\mathbf{KD} \vdash \varphi$ then $\varphi$ is true in all serial models, etc.

Completeness is the converse of soundness: that **K** is complete means that if a formula $\varphi$ is valid, $\vdash \varphi$, for instance. Proving completeness is a lot harder to do than proving soundness. It is useful, first, to consider the contrapositive: **K** is complete iff whenever $\nvdash \varphi$, there is a countermodel, i.e., a model $\mathfrak{M}$ such that $\mathfrak{M} \nVdash \varphi$. Equivalently (negating $\varphi$), we could prove that whenever $\nvdash \neg\varphi$, there is a model of $\varphi$. In the construction of such a model, we can use information contained in $\varphi$. When we find models for specific formulas we often do the same: E.g., if we want to find a countermodel to $p \to \Box q$, we know that it has to contain a world where $p$ is true and $\Box q$ is false. And a world where $\Box q$ is false means there has to be a world accessible from it where $q$ is false. And that's all we need to know: which worlds make the propositional variables true, and which worlds are accessible from which worlds.

In the case of proving completeness, however, we don't have a specific formula $\varphi$ for which we are constructing a model. We want to establish that a model exists for every $\varphi$ such that $\nvdash_\Sigma \neg\varphi$. This is a minimal requirement, since *if* $\vdash_\Sigma \neg\varphi$, by soundness, there is no model for $\varphi$ (in which $\Sigma$ is true). Now note that $\nvdash_\Sigma \neg\varphi$ iff $\varphi$ is $\Sigma$-consistent. (Recall that $\Sigma \nvdash_\Sigma \neg\varphi$ and $\varphi \nvdash_\Sigma \bot$ are equivalent.) So our task is to construct a model for every $\Sigma$-consistent formula.

The trick we'll use is to find a $\Sigma$-consistent set of formulas that contains $\varphi$, but also other formulas which tell us what the world that makes $\varphi$ true has to

look like. Such sets are *complete* $\Sigma$-consistent sets. It's not enough to construct a model with a single world to make $\varphi$ true, it will have to contain multiple worlds and an accessibility relation. The complete $\Sigma$-consistent set containing $\varphi$ will also contain other formulas of the form $\Box\psi$ and $\Diamond\chi$. In all accessible worlds, $\psi$ has to be true; in at least one, $\chi$ has to be true. In order to accomplish this, we'll simply take *all* possible complete $\Sigma$-consistent sets as the basis for the set of worlds. A tricky part will be to figure out when a complete $\Sigma$-consistent set should count as being accessible from another in our model.

We'll show that in the model so defined, $\varphi$ is true at a world—which is also a complete $\Sigma$-consistent set—iff $\varphi$ is an element of that set. If $\varphi$ is $\Sigma$-consistent, it will be an element of at least one complete $\Sigma$-consistent set (a fact we'll prove), and so there will be a world where $\varphi$ is true. So we will have a single model where every $\Sigma$-consistent formula $\varphi$ is true at some world. This single model is the *canonical* model for $\Sigma$.

## 41.2 Complete $\Sigma$-Consistent Sets

Suppose $\Sigma$ is a set of modal formulas—think of them as the axioms or defining principles of a normal modal logic. A set $\Gamma$ is $\Sigma$-consistent iff $\Gamma \nvdash_\Sigma \bot$, i.e., if there is no derivation of $\varphi_1 \to (\varphi_2 \to \cdots (\varphi_n \to \bot)\dots)$ from $\Sigma$, where each $\varphi_i \in \Gamma$. We will construct a "canonical" model in which each world is taken to be a special kind of $\Sigma$-consistent set: one which is not just $\Sigma$-consistent, but maximally so, in the sense that it settles the truth value of every modal formula: for every $\varphi$, either $\varphi \in \Gamma$ or $\neg\varphi \in \Gamma$:

**Definition 41.1.** A set $\Gamma$ is *complete $\Sigma$-consistent* if and only if it is $\Sigma$-consistent and for every $\varphi$, either $\varphi \in \Gamma$ or $\neg\varphi \in \Gamma$.

Complete $\Sigma$-consistent sets $\Gamma$ have a number of useful properties. For one, they are deductively closed, i.e., if $\Gamma \vdash_\Sigma \varphi$ then $\varphi \in \Gamma$. This means in particular that every instance of a formula $\varphi \in \Sigma$ is also $\in \Gamma$. Moreover, membership in $\Gamma$ mirrors the truth conditions for the propositional connectives. This will be important when we define the "canonical model."

**Proposition 41.2.** *Suppose $\Gamma$ is complete $\Sigma$-consistent. Then:*

1. *$\Gamma$ is deductively closed in $\Sigma$.*
2. *$\Sigma \subseteq \Gamma$.*
3. *$\bot \notin \Gamma$*
4. *$\neg\varphi \in \Gamma$ if and only if $\varphi \notin \Gamma$.*
5. *$\varphi \wedge \psi \in \Gamma$ iff $\varphi \in \Gamma$ and $\psi \in \Gamma$*
6. *$\varphi \vee \psi \in \Gamma$ iff $\varphi \in \Gamma$ or $\psi \in \Gamma$*

7. $\varphi \to \psi \in \Gamma$ *iff* $\varphi \notin \Gamma$ *or* $\psi \in \Gamma$

*Proof.* 1. Suppose $\Gamma \vdash_\Sigma \varphi$ but $\varphi \notin \Gamma$. Then since $\Gamma$ is complete $\Sigma$-consistent, $\neg\varphi \in \Gamma$. This would make $\Gamma$ inconsistent, since $\varphi, \neg\varphi \vdash_\Sigma \bot$.

2. If $\varphi \in \Sigma$ then $\Gamma \vdash_\Sigma \varphi$, and $\varphi \in \Gamma$ by deductive closure, i.e., case (1).

3. If $\bot \in \Gamma$, then $\Gamma \vdash_\Sigma \bot$, so $\Gamma$ would be $\Sigma$-inconsistent.

4. If $\neg\varphi \in \Gamma$, then by consistency $\varphi \notin \Gamma$; and if $\varphi \notin \Gamma$ then $\varphi \in \Gamma$ since $\Gamma$ is complete $\Sigma$-consistent.

5. Exercise.

6. Suppose $\varphi \vee \psi \in \Gamma$, and $\varphi \notin \Gamma$ and $\psi \notin \Gamma$. Since $\Gamma$ is complete $\Sigma$-consistent, $\neg\varphi \in \Gamma$ and $\neg\psi \in \Gamma$. Then $\neg(\varphi \vee \psi) \in \Gamma$ since $\neg\varphi \to (\neg\psi \to \neg(\varphi \vee \psi))$ is a tautological instance. This would mean that $\Gamma$ is $\Sigma$-inconsistent, a contradiction.

7. Exercise.

□

## 41.3 Lindenbaum's Lemma

Lindenbaum's Lemma establishes that every $\Sigma$-consistent set of formulas is contained in at least one *complete* $\Sigma$-consistent set. Our construction of the canonical model will show that for each complete $\Sigma$-consistent set $\Delta$, there is a world in the canonical model where all and only the formulas in $\Delta$ are true. So Lindenbaum's Lemma guarantees that every $\Sigma$-consistent set is true at some world in the canonical model.

**Theorem 41.3** (Lindenbaum's Lemma)**.** *If $\Gamma$ is $\Sigma$-consistent then there is a complete $\Sigma$-consistent set $\Delta$ extending $\Gamma$.*

*Proof.* Let $\varphi_0, \varphi_1, \ldots$ be an exhaustive listing of all formulas of the language (repetitions are allowed). For instance, start by listing $p_0$, and at each stage $n \geq 1$ list the finitely many formulas of length $n$ using only variables among $p_0, \ldots, p_n$. We define sets of formulas $\Delta_n$ by induction on $n$, and we then set $\Delta = \bigcup_n \Delta_n$. We first put $\Delta_0 = \Gamma$. Supposing that $\Delta_n$ has been defined, we define $\Delta_{n+1}$ by:

$$\Delta_{n+1} = \begin{cases} \Delta_n \cup \{\varphi_n\}, & \text{if } \Delta_n \cup \{\varphi_n\} \text{ is consistent;} \\ \Delta_n \cup \{\neg\varphi_n\}, & \text{otherwise.} \end{cases}$$

If we now let $\Delta = \bigcup_{n=0}^{\infty} \Delta_n$.

We have to show that this definition actually yields a set $\Delta$ with the required properties, i.e., $\Gamma \subseteq \Delta$ and $\Delta$ is complete $\Sigma$-consistent.

It's obvious that $\Gamma \subseteq \Delta$, since $\Delta_0 \subseteq \Delta$ by construction, and $\Delta_0 = \Gamma$. In fact, $\Delta_n \subseteq \Delta$ for all $n$, since $\Delta$ is the union of all $\Delta_n$. (Since in each step of the construction, we add a formula to the set already constructed, $\Delta_n \subseteq \Delta_{n+1}$, so since $\subseteq$ is transitive, $\Delta_n \subseteq \Delta_m$ whenever $n \leq m$.) At each stage of the construction, we either add $\varphi_n$ or $\neg\varphi_n$, and every formula appears (at least once) in the list of all $\varphi_n$. So, for every $\varphi$ either $\varphi \in \Delta$ or $\neg\varphi \in \Delta$, so $\Delta$ is complete by definition.

Finally, we have to show, that $\Delta$ is $\Sigma$-consistent. To do this, we show that (a) if $\Delta$ were $\Sigma$-inconsistent, then some $\Delta_n$ would be $\Sigma$-inconsistent, and (b) all $\Delta_n$ are $\Sigma$-consistent.

So suppose $\Delta$ were $\Sigma$-inconsistent. Then $\Delta \vdash_\Sigma \bot$, i.e., there are $\varphi_1, \ldots, \varphi_k \in \Delta$ such that $\Sigma \vdash \varphi_1 \to (\varphi_2 \to \cdots (\varphi_k \to \bot) \ldots)$. Since $\Delta = \bigcap_{n=0}^{\infty}$, each $\varphi_i \in \Delta_{n_i}$ for some $n_i$. Let $n$ be the largest of these. Since $n_i \leq n$, $\Delta_{n_i} \subseteq \Delta_n$. So, all $\varphi_i$ are in some $\Delta_n$. This would mean $\Delta_n \vdash_\Sigma \bot$, i.e., $\Delta_n$ is $\Sigma$-inconsistent.

To show that each $\Delta_n$ is $\Sigma$-consistent, we use a simple induction on $n$. $\Delta_0 = \Gamma$, and we assumed $\Gamma$ was $\Sigma$-consistent. So the claim holds for $n = 0$. Now suppose it holds for $n$, i.e., $\Delta_n$ is $\Sigma$-consistent. $\Delta_{n+1}$ is either $\Delta_n \cup \{\varphi_n\}$ is that is $\Sigma$-consistent, otherwise it is $\Delta_n \cup \{\neg\varphi_n\}$. In the first case, $\Delta_{n+1}$ is clearly $\Sigma$-consistent. However, by Proposition 40.39(3), either $\Delta_n \cup \{\varphi_n\}$ or $\Delta_n \cup \{\neg\varphi_n\}$ is consistent, so $\Delta_{n+1}$ is consistent in the other case as well. □

**Corollary 41.4.** *$\Gamma \vdash_\Sigma \varphi$ if and only if $\varphi \in \Delta$ for each complete $\Sigma$-consistent set $\Delta$ extending $\Gamma$ (including when $\Gamma = \emptyset$, in which case we get another characterization of the modal system $\Sigma$.)*

*Proof.* Suppose $\Gamma \vdash_\Sigma \varphi$, and let $\Delta$ be any complete $\Sigma$-consistent set extending $\Gamma$. If $\varphi \notin \Delta$ then by maximality $\neg\varphi \in \Delta$ and so $\Delta \vdash_\Sigma \varphi$ (by monotony) and $\Delta \vdash_\Sigma \neg\varphi$ (by reflexivity), and so $\Delta$ is inconsistent. Conversely if $\Gamma \nvdash_\Sigma \varphi$, then $\Gamma \cup \{\neg\varphi\}$ is $\Sigma$-consistent, and by Lindenbaum's Lemma there is a complete consistent set $\Delta$ extending $\Gamma \cup \{\neg\varphi\}$. By consistency, $\varphi \notin \Delta$. □

## 41.4 Modalities and Complete Consistent Sets

When we construct a model $\mathfrak{M}^\Sigma$ whose set of worlds is given by the complete $\Sigma$-consistent sets $\Delta$ in some normal modal logic $\Sigma$, we will also need to define an accessibility relation $R^\Sigma$ between such "worlds." We want it to be the case that the accessibility relation (and the assignment $V^\Sigma$) are defined in such a way that $\mathfrak{M}^\Sigma, \Delta \Vdash \varphi$ iff $\varphi \in \Delta$. How should we do this?

Once the accessibility relation is defined, the definition of truth at a world ensures that $\mathfrak{M}^\Sigma, \Delta \Vdash \Box\varphi$ iff $\mathfrak{M}^\Sigma, \Delta' \Vdash \varphi$ for all $\Delta'$ such that $R^\Sigma\Delta\Delta'$. The proof that $\mathfrak{M}^\Sigma, \Delta \Vdash \varphi$ iff $\varphi \in \Delta$ requires that this is true in particular for formulas starting with a modal operator, i.e., $\mathfrak{M}^\Sigma, \Delta \Vdash \Box\varphi$ iff $\Box\varphi \in \Delta$. Combining this requirement with the definition of truth at a world for $\Box\varphi$ yields:

$$\Box\varphi \in \Delta \text{ iff } \varphi \in \Delta' \text{ for all } \Delta' \text{ with } R^\Sigma\Delta\Delta'$$

Consider the left-to-right direction: it says that if $\Box\varphi \in \Delta$, then $\varphi \in \Delta'$ for any $\varphi$ and any $\Delta'$ with $R^\Sigma\Delta\Delta'$. If we stipulate that $R^\Sigma\Delta\Delta'$ iff $\varphi \in \Delta'$ for all $\Box\varphi \in \Delta$, then this holds. We can write the condition on the right of the "iff" more compactly as: $\{\varphi : \Box\varphi \in \Delta\} \subseteq \Delta'$.

So the question is: does this definition of $R^\Sigma$ in fact guarantee that $\Box\varphi \in \Delta$ iff $\mathfrak{M}^\Sigma, \Delta \Vdash \Box\varphi$? Does it also guarantee that $\Diamond\varphi \in \Delta$ iff $\mathfrak{M}^\Sigma, \Delta \Vdash \Diamond\varphi$? The next few results will establish this.

**Definition 41.5.** If $\Gamma$ is a set of formulas, let

$$\Box\Gamma = \{\Box\psi : \psi \in \Gamma\}$$
$$\Diamond\Gamma = \{\Diamond\psi : \psi \in \Gamma\}$$

and

$$\Box^{-1}\Gamma = \{\psi : \Box\psi \in \Gamma\}$$
$$\Diamond^{-1}\Gamma = \{\psi : \Diamond\psi \in \Gamma\}$$

In other words, $\Box\Gamma$ is $\Gamma$ with $\Box$ in front of every formula in $\Gamma$; $\Box^{-1}\Gamma$ is all the $\Box$'ed formulas of $\Gamma$ with the initial $\Box$'s removed. This definition is not terribly important on its own, but will simplify the notation considerably.

Note that $\Box\Box^{-1}\Gamma \subseteq \Gamma$:

$$\Box\Box^{-1}\Gamma = \{\Box\psi : \Box\psi \in \Gamma\}$$

i.e., it's just the set of all those formulas of $\Gamma$ that start with $\Box$.

**Lemma 41.6.** *If $\Gamma \vdash_\Sigma \varphi$ then $\Box\Gamma \vdash_\Sigma \Box\varphi$.*

*Proof.* If $\Gamma \vdash_\Sigma \varphi$ then there are $\psi_1, \ldots, \psi_k \in \Gamma$ such that $\Sigma \vdash \psi_1 \to (\psi_2 \to \cdots(\psi_n \to \varphi)\cdots)$. Since $\Sigma$ is normal, by rule RK, $\Sigma \vdash \Box\psi_1 \to (\Box\psi_2 \to \cdots(\Box\psi_n \to \Box\varphi)\cdots)$, where obviously $\Box\psi_1, \ldots, \Box\psi_k \in \Box\Gamma$. Hence, by definition, $\Box\Gamma \vdash_\Sigma \Box\varphi$. □

**Lemma 41.7.** *If $\Box^{-1}\Gamma \vdash_\Sigma \varphi$ then $\Gamma \vdash_\Sigma \Box\varphi$.*

*Proof.* Suppose $\Box^{-1}\Gamma \vdash_\Sigma \varphi$; then by Lemma 41.6, $\Box\Box^{-1}\Gamma \vdash \Box\varphi$. But since $\Box\Box^{-1}\Gamma \subseteq \Gamma$, also $\Gamma \vdash_\Sigma \Box\varphi$ by Monotony. □

**Proposition 41.8.** *If $\Gamma$ is complete $\Sigma$-consistent, then $\Box\varphi \in \Gamma$ if and only if for every complete $\Sigma$-consistent $\Delta$ such that $\Box^{-1}\Gamma \subseteq \Delta$, it holds that $\varphi \in \Delta$.*

*Proof.* Suppose $\Gamma$ is complete $\Sigma$-consistent. The "only if" direction is easy: Suppose $\Box\varphi \in \Gamma$ and that $\Box^{-1}\Gamma \subseteq \Delta$. Since $\Box\varphi \in \Gamma$, $\varphi \in \Box^{-1}\Gamma \subseteq \Delta$, so $\varphi \in \Delta$.

For the "if" direction, we prove the contrapositive: Suppose $\Box\varphi \notin \Gamma$. Since $\Gamma$ is complete $\Sigma$-consistent, it is deductively closed, and hence $\Gamma \nvdash_\Sigma \Box\varphi$. By Lemma 41.7, $\Box^{-1}\Gamma \nvdash_\Sigma \varphi$. By Proposition 40.39(2), $\Box^{-1}\Gamma \cup \{\neg\varphi\}$ is $\Sigma$-consistent. By Lindenbaum's Lemma, there is a complete $\Sigma$-consistent set $\Delta$ such that $\Box^{-1}\Gamma \cup \{\neg\varphi\} \subseteq \Delta$. By consistency, $\varphi \notin \Delta$. □

**Lemma 41.9.** *Suppose $\Gamma$ and $\Delta$ are complete $\Sigma$-consistent. Then: $\Box^{-1}\Gamma \subseteq \Delta$ if and only if $\Diamond\Delta \subseteq \Gamma$.*

*Proof.* "Only if" direction: Assume $\Box^{-1}\Gamma \subseteq \Delta$ and suppose $\Diamond\varphi \in \Diamond\Delta$ (i.e., $\varphi \in \Delta$). In order to show $\Diamond\varphi \in \Gamma$ it suffices to show $\Box\neg\varphi \notin \Gamma$ for then by maximality $\neg\Box\neg\varphi \in \Gamma$. Now, if $\Box\neg\varphi \in \Gamma$ then by hypothesis $\neg\varphi \in \Delta$, against the consistency of $\Delta$ (since $\varphi \in \Delta$). Hence $\Box\neg\varphi \notin \Gamma$, as required.

"If" direction: Assume $\Diamond\Delta \subseteq \Gamma$. We argue contrapositively: suppose $\varphi \notin \Delta$ in order to show $\Box\varphi \notin \Gamma$. If $\varphi \notin \Delta$ then by maximality $\neg\varphi \in \Delta$ and so by hypothesis $\Diamond\neg\varphi \in \Gamma$. But in a normal modal logic $\Diamond\neg\varphi$ is equivalent to $\neg\Box\varphi$, and if the latter is in $\Gamma$, by consistency $\Box\varphi \notin \Gamma$, as required. □

**Proposition 41.10.** *If $\Gamma$ is complete $\Sigma$-consistent, then $\Diamond\varphi \in \Gamma$ if and only if for some complete $\Sigma$-consistent $\Delta$ such that $\Diamond\Delta \subseteq \Gamma$, it holds that $\varphi \in \Delta$.*

*Proof.* Suppose $\Gamma$ is complete $\Sigma$-consistent. $\Diamond\varphi \in \Gamma$ iff $\neg\Box\neg\varphi \in \Gamma$ by DUAL and closure. $\neg\Box\neg\varphi \in \Gamma$ iff $\Box\neg\varphi \notin \Gamma$ by Proposition 41.2(4) since $\Gamma$ is complete $\Sigma$-consistent. By Proposition 41.8, $\Box\neg\varphi \notin \Gamma$ iff, for some complete $\Sigma$-consistent $\Delta$ with $\Box^{-1}\Gamma \subseteq \Delta$, $\neg\varphi \notin \Delta$. Now consider any such $\Delta$. By Lemma 41.9, $\Box^{-1}\Gamma \subseteq \Delta$ iff $\Diamond\Delta \subseteq \Gamma$. Also, $\neg\varphi \notin \Delta$ iff $\varphi \in \Delta$ by Proposition 41.2(4). So $\Diamond\varphi \in \Gamma$ iff, for some complete $\Sigma$-consistent $\Delta$ with $\Diamond\Delta \subseteq \Gamma$, $\varphi \in \Delta$. □

## 41.5 Canonical Models

The *canonical model* for a modal system $\Sigma$ is a specific model $\mathfrak{M}^\Sigma$ in which the worlds are all complete $\Sigma$-consistent sets. Its accessibility relation $R^\Sigma$ and valuation $V^\Sigma$ are defined so as to guarantee that the formulas true at a world $\Delta$ are exactly the formulas making up $\Delta$.

**Definition 41.11.** Let $\Sigma$ be a normal modal logic. The *canonical model* for $\Sigma$ is $\mathfrak{M}^\Sigma = \langle W^\Sigma, R^\Sigma, V^\Sigma\rangle$, where:

1. $\mathfrak{M}^\Sigma = \{\Delta : \Delta \text{ is complete } \Sigma\text{-consistent}\}$.
2. $R^\Sigma\Delta\Delta'$ holds if and only if $\Box^{-1}\Delta \subseteq \Delta'$.
3. $V^\Sigma(p) = \{\Delta : p \in \Delta\}$.

## 41.6 The Truth Lemma

The canonical model $\mathfrak{M}^\Sigma$ is defined in such a way that $\mathfrak{M}^\Sigma, \Delta \Vdash \varphi$ iff $\varphi \in \Delta$. For propositional variables, the definition of $V^\Sigma$ yields this directly. We have to verify that the equivalence holds for all formulas, however. We do this by induction. The inductive step involves proving the equivalence for formulas involving propositional operators (where we have to use Proposition 41.2) and the modal operators (where we invoke the results of section 41.4).

**Proposition 41.12** (Truth Lemma)**.** *For every formula $\varphi$, $\mathfrak{M}^\Sigma, \Delta \Vdash \varphi$ if and only if $\varphi \in \Delta$.*

*Proof.* By induction on $\varphi$.

1. $\varphi \equiv \bot$: $\mathfrak{M}^\Sigma, \Delta \nVdash \bot$ by Definition 38.6, and $\bot \notin \Delta$ by Proposition 41.2(3).

2. $\varphi \equiv p$: $\mathfrak{M}^\Sigma, \Delta \Vdash p$ iff $\Delta \in V^\Sigma(p)$ by Definition 38.6. Also, $\Delta \in V^\Sigma(p)$ iff $p \in \Delta$ by definition of $V^\Sigma$.

3. $\varphi \equiv \neg\psi$: $\mathfrak{M}^\Sigma, \Delta \Vdash \neg\psi$ iff $\mathfrak{M}^\Sigma, \Delta \nVdash \psi$ (Definition 38.6) iff $\psi \notin \Delta$ (by inductive hypothesis) iff $\neg\psi \in \Delta$ (by Proposition 41.2(4)).

4. $\varphi \equiv \psi \wedge \chi$: Exercise.

5. $\varphi \equiv \psi \vee \chi$: $\mathfrak{M}^\Sigma, \Delta \Vdash \psi \vee \chi$ iff $\mathfrak{M}^\Sigma, \Delta \Vdash \psi$ or $\mathfrak{M}^\Sigma, \Delta \Vdash \chi$ (by Definition 38.6) iff $\psi \in \Delta$ or $\chi \in \Delta$ (by inductive hypothesis) iff $\psi \vee \chi \in \Delta$ (by Proposition 41.2(6)).

6. $\varphi \equiv \psi \to \chi$: Exercise.

7. $\varphi \equiv \Box\psi$: First suppose that $\mathfrak{M}^\Sigma, \Delta \Vdash \Box\psi$. By Definition 38.6, for every $\Delta'$ such that $R^\Sigma\Delta\Delta'$, $\mathfrak{M}^\Sigma, \Delta' \Vdash \psi$. By inductive hypothesis, for every $\Delta'$ such that $R^\Sigma\Delta\Delta'$, $\psi \in \Delta'$. By definition of $R^\Sigma$, for every $\Delta'$ such that $\Box^{-1}\Delta \subseteq \Delta'$, $\psi \in \Delta'$. By Proposition 41.8, $\Box\psi \in \Delta$.

   Now assume $\Box\psi \in \Delta$. Let $\Delta' \in W^\Sigma$ be such that $R^\Sigma\Delta\Delta'$, i.e., $\Box^{-1}\Delta \subseteq \Delta'$. Since $\Box\psi \in \Delta$, $\psi \in \Box^{-1}\Delta$. Consequently, $\psi \in \Delta'$. By inductive hypothesis, $\mathfrak{M}^\Sigma, \Delta' \Vdash \psi$. Since $\Delta'$ is arbitrary with $R^\Sigma\Delta\Delta'$, for all $\Delta' \in W^\Sigma$ such that $R^\Sigma\Delta\Delta'$, $\mathfrak{M}^\Sigma, \Delta' \Vdash \psi$. By Definition 38.6, $\mathfrak{M}^\Sigma, \Delta \Vdash \Box\psi$.

8. $\varphi \equiv \Diamond\psi$: Exercise.

□

## 41.7 Determination and Completeness for K

We are now prepared to use the canonical model to establish completeness. Completeness follows from the fact that the formulas true in the canonical for $\Sigma$ are exactly the $\Sigma$-derivable ones. Models with this property are said to *determine* $\Sigma$.

**Definition 41.13.** A model $\mathfrak{M}$ *determines* a normal modal logic $\Sigma$ precisely when $\mathfrak{M} \Vdash \varphi$ if and only if $\Sigma \vdash \varphi$, for all formulas $\varphi$.

**Theorem 41.14** (Determination)**.** $\mathfrak{M}^\Sigma \Vdash \varphi$ *if and only if* $\Sigma \vdash \varphi$.

*Proof.* If $\mathfrak{M}^\Sigma \Vdash \varphi$, then for every complete $\Sigma$-consistent $\Delta$, we have $\mathfrak{M}^\Sigma, \Delta \Vdash \varphi$. Hence, by the Truth Lemma, $\varphi \in \Delta$ for every complete $\Sigma$-consistent $\Delta$, whence by Corollary 41.4 (with $\Gamma = \emptyset$), $\Sigma \vdash \varphi$.

Conversely, if $\Sigma \vdash \varphi$ then by Proposition 41.2(1), every complete $\Sigma$-consistent $\Delta$ contains $\varphi$, and hence by the Truth Lemma, $\mathfrak{M}^\Sigma, \Delta \Vdash \varphi$ for every $\Delta \in W^\Sigma$, i.e., $\mathfrak{M}^\Sigma \Vdash \varphi$. □

Since the canonical model for **K** determines **K**, we immediately have completeness of **K** as a corollary:

**Corollary 41.15.** *The basic modal logic* **K** *is complete with respect to the class of all models, i.e., if* $\vDash \varphi$ *then* $\mathbf{K} \vdash \varphi$.

*Proof.* Contrapositively, if $\mathbf{K} \nvdash \varphi$ then by Determination $\mathfrak{M}^\mathbf{K} \nVdash \varphi$ and hence $\varphi$ is not valid. □

For the general case of completeness of a system $\Sigma$ with respect to a class of models, e.g., of **KTB4** with respect to the class of reflexive, symmetric, transitive models, determination alone is not enough. We must also show that the canonical model for the system $\Sigma$ is a member of the class, which does not follow obviously from the canoncial model construction—nor is it always true!

## 41.8 Frame Completeness

The completeness theorem for **K** can be extended to other modal systems, once we show that the canonical model for a given logic has the corresponding frame property.

**Theorem 41.16.** *If a normal modal logic* $\Sigma$ *contains one of the formulas on the left-hand side of table 41.1, then the canonical model for* $\Sigma$ *has the corresponding property on the right-hand side.*

| *If $\Sigma$ contains ...* | *... the canonical model for $\Sigma$ is:* |
|---|---|
| D: $\Box\varphi \to \Diamond\varphi$ | serial; |
| T: $\Box\varphi \to \varphi$ | reflexive; |
| B: $\varphi \to \Box\Diamond\varphi$ | symmetric; |
| 4: $\Box\varphi \to \Box\Box\varphi$ | transitive; |
| 5: $\Diamond\varphi \to \Box\Diamond\varphi$ | euclidean. |

Table 41.1: Basic correspondence facts.

*Proof.* We take each of these up in turn.

Suppose $\Sigma$ contains D, and let $\Delta \in W^\Sigma$; we need to show that there is a $\Delta'$ such that $R^\Sigma\Delta\Delta'$. It suffices to show that $\Box^{-1}\Delta$ is $\Sigma$-consistent, for then by Lindenbaum's Lemma, there is a complete $\Sigma$-consistent set $\Delta' \supseteq \Box^{-1}\Delta$, and by definition of $R^\Sigma$ we have $R^\Sigma\Delta\Delta'$. So, suppose for contradiction that $\Box^{-1}\Delta$ is *not* $\Sigma$-consistent, i.e., $\Box^{-1}\Delta \vdash_\Sigma \bot$. By Lemma 41.7, $\Delta \vdash_\Sigma \Box\bot$, and since $\Sigma$ contains D, also $\Delta \vdash_\Sigma \Diamond\bot$. But $\Sigma$ is normal, so $\Sigma \vdash \neg\Diamond\bot$ (Proposition 40.7), whence also $\Delta \vdash_\Sigma \neg\Diamond\bot$, against the consistency of $\Delta$.

Now suppose $\Sigma$ contains T, and let $\Delta \in W^\Sigma$. We want to show $R^\Sigma\Delta\Delta$, i.e., $\Box^{-1}\Delta \subseteq \Delta$. But if $\Box\varphi \in \Delta$ then by T also $\varphi \in \Delta$, as desired.

Now suppose $\Sigma$ contains B, and suppose $R^\Sigma\Delta\Delta'$ for $\Delta, \Delta' \in W^\Sigma$. We need to show that $R^\Sigma\Delta'\Delta$, i.e., $\Box^{-1}\Delta' \subseteq \Delta$. By Lemma 41.9, this is equivalent to $\Diamond\Delta \subseteq \Delta'$. So suppose $\varphi \in \Delta$. By B, also $\Box\Diamond\varphi \in \Delta$. By the hypothesis that $R^\Sigma\Delta\Delta'$, we have that $\Box^{-1}\Delta \subseteq \Delta'$, and hence $\Diamond\varphi \in \Delta'$, as required.

Now suppose $\Sigma$ contains 4, and suppose $R^\Sigma\Delta_1\Delta_2$ and $R^\Sigma\Delta_2\Delta_3$. We need to show $R^\Sigma\Delta_1\Delta_3$. From the hypothesis we have both $\Box^{-1}\Delta_1 \subseteq \Delta_2$ and $\Box^{-1}\Delta_2 \subseteq \Delta_3$. In order to show $R^\Sigma\Delta_1\Delta_3$ it suffices to show $\Box^{-1}\Delta_1 \subseteq \Delta_3$. So let $\psi \in \Box^{-1}\Delta_1$, i.e., $\Box\psi \in \Delta_1$. By 4, also $\Box\Box\psi \in \Delta_1$ and by hypothesis we get, first, that $\Box\psi \in \Delta_2$ and, second, that $\psi \in \Delta_3$, as desired.

Now suppose $\Sigma$ contains 5, suppose $R^\Sigma\Delta_1\Delta_2$ and $R^\Sigma\Delta_1\Delta_3$. We need to show $R^\Sigma\Delta_2\Delta_3$. The first hypothesis gives $\Box^{-1}\Delta_1 \subseteq \Delta_2$, and the second hypothesis is equivalent to $\Diamond\Delta_3 \subseteq \Delta_2$, by Lemma 41.9. To show $R^\Sigma\Delta_2\Delta_3$, by Lemma 41.9, it suffices to show $\Diamond\Delta_3 \subseteq \Delta_2$. So let $\Diamond\varphi \in \Diamond\Delta_3$, i.e., $\varphi \in \Delta_3$. By the second hypothesis $\Diamond\varphi \in \Delta_1$ and by 5, $\Box\Diamond\varphi \in \Delta_1$ as well. But now the first hypothesis gives $\Diamond\varphi \in \Delta_2$, as desired. □

As a corollary we obtain completeness results for a number of systems. For instance, we know that **S5** = **KT5** = **KTB4** is complete with respect to the class of all reflexive euclidean models, which is the same as the class of all reflexive, symmetric and transitive models.

**Theorem 41.17.** *Let $\mathcal{C}_\mathrm{D}$, $\mathcal{C}_\mathrm{T}$, $\mathcal{C}_\mathrm{B}$, $\mathcal{C}_4$, and $\mathcal{C}_5$ be the class of all serial, reflexive, symmetric, transitive, and euclidean models (respectively). Then for any schemas $\varphi_1, \ldots, \varphi_n$ among* D, T, B, 4, *and* 5, *the system $\mathbf{K}\varphi_1 \ldots \varphi_n$ is determined by the class of models $\mathcal{C} = \mathcal{C}_{\varphi_1} \cap \cdots \cap \mathcal{C}_{\varphi_n}$.*

**Proposition 41.18.** *Let $\Sigma$ be a normal modal logic; then:*

1. *If $\Sigma$ contains the schema $\Diamond\varphi \to \Box\varphi$ then the canonical model for $\Sigma$ is partially functional.*

2. *If $\Sigma$ contains the schema $\Diamond\varphi \leftrightarrow \Box\varphi$ then the canonical model for $\Sigma$ is functional.*

3. *If $\Sigma$ contains the schema $\Box\Box\varphi \to \Box\varphi$ then the canonical model for $\Sigma$ is weakly dense.*

*(see table 39.2 for definitions of these frame properties).*

*Proof.* 1. suppose that $\Sigma$ contains the schema $\Diamond\varphi \to \Box\varphi$, to show that $R^\Sigma$ is partially functional we need to prove that for any $\Delta_1$, $\Delta_2$, $\Delta_3 \in W^\Sigma$, if $R^\Sigma\Delta_1\Delta_2$ and $R^\Sigma\Delta_1\Delta_3$ then $\Delta_2 = \Delta_3$. Since $R^\Sigma\Delta_1\Delta_2$ we have $\Box^{-1}\Delta_1 \subseteq \Delta_2$ and since $R^\Sigma\Delta_1\Delta_3$ also $\Box^{-1}\Delta_1 \subseteq \Delta_3$. The identity $\Delta_2 = \Delta_3$ will follow if we can establish the two inclusions $\Delta_2 \subseteq \Delta_3$ and $\Delta_3 \subseteq \Delta_2$. For the first inclusion, let $\varphi \in \Delta_2$; then $\Diamond\varphi \in \Delta_1$, and by the schema and deductive closure of $\Delta_1$ also $\Box\varphi \in \Delta_1$, whence by the hypothesis that $R^\Sigma\Delta_1\Delta_3$, $\varphi \in \Delta_3$. The second inclusion is similar.

2. This follows immediately from part (1) and the seriality proof in Theorem 41.16.

3. Suppose $\Sigma$ contains the schema $\Box\Box\varphi \to \Box\varphi$ and to show that $R^\Sigma$ is weakly dense, let $R^\Sigma\Delta_1\Delta_2$. We need to show that there is a complete $\Sigma$-consistent set $\Delta_3$ such that $R^\Sigma\Delta_1\Delta_3$ and $R^\Sigma\Delta_3\Delta_2$. Let:

$$\Gamma = \Box^{-1}\Delta_1 \cup \Diamond\Delta_2.$$

It suffices to show that $\Gamma$ is $\Sigma$-consistent, for then by Lindenbaum's Lemma it can be extended to a complete $\Sigma$-consistent set $\Delta_3$ such that $\Box^{-1}\Delta_1 \subseteq \Delta_3$ and $\Diamond\Delta_2 \subseteq \Delta_3$, i.e., $R^\Sigma\Delta_1\Delta_3$ and $R^\Sigma\Delta_3\Delta_2$ (by Lemma 41.9).

Suppose for contradiction that $\Gamma$ is not consistent. Then there are formulas $\Box\varphi_1, \ldots, \Box\varphi_n \in \Delta_1$ and $\psi_1, \ldots, \psi_m \in \Delta_2$ such that

$$\varphi_1, \ldots, \varphi_n, \Diamond\psi_1, \ldots, \Diamond\psi_m \vdash_\Sigma \bot.$$

Since $\Diamond(\psi_1 \wedge \cdots \wedge \psi_m) \to (\Diamond\psi_1 \wedge \cdots \wedge \Diamond\psi_m)$ is derivable in every normal

modal logic, we argue as follows, contradicting the consistency of $\Delta_2$:

$$
\begin{array}{ll}
\varphi_1, \ldots, \varphi_n, \Diamond\psi_1, \ldots, \Diamond\psi_m \vdash_\Sigma \bot & \\
\quad \Rightarrow \varphi_1, \ldots, \varphi_n \vdash_\Sigma (\Diamond\psi_1 \wedge \cdots \wedge \Diamond\psi_m) \to \bot, & \text{deduction theorem;} \\
\quad \Rightarrow \varphi_1, \ldots, \varphi_n \vdash_\Sigma \Diamond(\psi_1 \wedge \cdots \wedge \psi_m) \to \bot, & \Sigma \text{ is normal;} \\
\quad \Rightarrow \varphi_1, \ldots, \varphi_n \vdash_\Sigma \Box\neg(\psi_1 \wedge \cdots \wedge \psi_m), & \text{PL;} \\
\quad \Rightarrow \Box\varphi_1, \ldots, \Box\varphi_n \vdash_\Sigma \Box\Box\neg(\psi_1 \wedge \cdots \wedge \psi_m), & \text{Lemma 41.6;} \\
\quad \Rightarrow \Box\varphi_1, \ldots, \Box\varphi_n \vdash_\Sigma \Box\neg(\psi_1 \wedge \cdots \wedge \psi_m), & \text{by the schema;} \\
\quad \Rightarrow \Delta_1 \vdash_\Sigma \Box\neg(\psi_1 \wedge \cdots \wedge \psi_m), & \text{Monotony;} \\
\quad \Rightarrow \Box\neg(\psi_1 \wedge \cdots \wedge \psi_m) \in \Delta_1, & \text{deductive closure;} \\
\quad \Rightarrow \neg(\psi_1 \wedge \cdots \wedge \psi_m) \in \Delta_2, & \text{since } R^\Sigma \Delta_1 \Delta_2.
\end{array}
$$

□

On the strength of these examples, one might think that every system $\Sigma$ of modal logic is *complete*, in the sense that it proves every formula which is valid in every frame in which every theorem of $\Sigma$ is valid. Unfortunately, there are many systems that are not complete in this sense.

## Problems

**Problem 41.1.** Complete the proof of Proposition 41.2.

**Problem 41.2.** Show that if $\Gamma$ is complete $\Sigma$-consistent, then $\Diamond\varphi \in \Gamma$ if and only if there is a complete $\Sigma$-consistent $\Delta$ such that $\Box^{-1}\Gamma \subseteq \Delta$ and $\varphi \in \Delta$. *Do this without using Lemma 41.9.*

**Problem 41.3.** Complete the proof of Proposition 41.12.

# Chapter 42

# Filtrations and Decidability

## 42.1 Introduction

One important question about a logic is always whether it is decidable, i.e., if there is an effective procedure which will answer the question "is this formula valid." Propositional logic is decidable: we can effectively test if a formula is a tautology by constructing a truth table, and for a given formula, the truth table is finite. But we can't obviously test if a modal formula is true in all models, for there are infinitely many of them. We can list all the finite models relevant to a given formula, since only the assignment of subsets of worlds to propositional variables which actually occur in the formula are relevant. If the accessibility relation is fixed, the possible different assignments $V(p)$ are just all the subsets of $W$, and if $|W| = n$ there are $2^n$ of those. If our formula $\varphi$ contains $m$ propositional variables there are then $2^{nm}$ different models with $n$ worlds. For each one, we can test if $\varphi$ is true at all worlds, simply by computing the truth value of $\varphi$ in each. Of course, we also have to check all possible accessibility relations, but there are only finitely many relations on $n$ worlds as well (specifically, the number of subsets of $W \times W$, i.e., $2^{n^2}$.

If we are not interested in the logic **K**, but a logic defined by some class of models (e.g., the reflexive transitive models), we also have to be able to test if the accessibility relation is of the right kind. We can do that whenever the frames we are interested in are definable by modal formulas (e.g., by testing if T and 4 valid in the frame). So, the idea would be to run through all the finite frames, test each one if it is a frame in the class we're interested in, then list all the possible models on that frame and test if $\varphi$ is true in each. If not, stop: $\varphi$ is not valid in the class of models of interest.

There is a problem with this idea: we don't know when, if ever, we can stop looking. If the formula has a finite countermodel, our procedure will find it. But if it has no finite countermodel, we won't get an answer. The formula may be valid (no countermodels at all), or it have only an infinite countermodel, which we'll never look at. This problem can be overcome if we can show that

every formula that has a countermodel has a finite countermodel. If this is the case we say the logic has the *finite model property*.

But how would we show that a logic has the finite model property? One way of doing this would be to find a way to turn an infinite (counter)model of $\varphi$ into a finite one. If that can be done, then whenever there is a model in which $\varphi$ is not true, then the resulting finite model also makes $\varphi$ not true. That finite model will show up on our list of all finite models, and we will eventually determine, for every formula that is not valid, that it isn't. Our procedure won't terminate if the formula is valid. If we can show in addition that there is some maximum size that the finite model our procedure provides can have, and that this maximum size depends only on the formula $\varphi$, we will have a size up to which we have to test finite models in our search for countermodels. If we haven't found a countermodel by then, there are none. Then our procedure will, in fact, decide the question "is $\varphi$ valid?" for any formula $\varphi$.

A strategy that often works for turning infinite structures into finite structures is that of "identifying" elements of the structure which behave the same way in relevant respects. If there are infinitely many worlds in $\mathfrak{M}$ that behave the same in relevant respects, then we might hope that there are only finitely many "classes" of such worlds. In other words, we partition the set of worlds in the right way. Each partition contains infinitely many worlds, but there are only finitely many partitions. Then we define a new model $\mathfrak{M}^*$ where the worlds are the partitions. Finitely many partitions in the old model give us finitely many worlds in the new model, i.e., a finite model. Let's call the partition a world $w$ is in $[w]$. We'll want it to be the case that $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}^*, [w] \Vdash \varphi$, since we want the new model to be a countermodel to $\varphi$ if the old one was. This requires that we define the partition, as well as the accessibility relation of $\mathfrak{M}^*$ in the right way.

To see how this would go, first imagine we have no accessibility relation. $\mathfrak{M}, w \Vdash \Box\psi$ iff for some $v \in W$, $\mathfrak{M}, v \Vdash \Box\psi$, and the same for $\mathfrak{M}^*$, except with $[w]$ and $[v]$. As a first idea, let's say that two worlds $u$ and $v$ are equivalent (belong to the same partition) if they agree on all propositional variables in $\mathfrak{M}$, i.e., $\mathfrak{M}, u \Vdash p$ iff $\mathfrak{M}, v \Vdash p$. Let $V^*(p) = \{[w] : \mathfrak{M}, w \Vdash p\}$. Our aim is to show that $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}^*, [w] \Vdash \varphi$. Obviously, we'd prove this by induction: The base case would be $\varphi \equiv p$. First suppose $\mathfrak{M}, w \Vdash p$. Then $[w] \in V^*$ by definition, so $\mathfrak{M}^*, [w] \Vdash p$. Now suppose that $\mathfrak{M}^*, [w] \Vdash p$. That means that $[w] \in V^*(p)$, i.e., for some $v$ equivalent to $w$, $\mathfrak{M}, v \Vdash p$. But "$w$ equivalent to $v$" means "$w$ and $v$ make all the same propositional variables true," so $\mathfrak{M}, w \Vdash p$. Now for the inductive step, e.g., $\varphi \equiv \neg\psi$. Then $\mathfrak{M}, w \Vdash \neg\psi$ iff $\mathfrak{M}, w \nVdash \psi$ iff $\mathfrak{M}^*, [w] \nVdash \psi$ (by inductive hypothesis) iff $\mathfrak{M}^*, [w] \Vdash \neg\psi$. Similarly for the other non-modal operators. It also works for $\Box$: suppose $\mathfrak{M}^*, [w] \Vdash \Box\psi$. That means that for every $[u]$, $\mathfrak{M}^*, [u] \Vdash \psi$. By inductive hypothesis, for every $u$, $\mathfrak{M}, u \Vdash \psi$. Consequently, $\mathfrak{M}, w \Vdash \Box\psi$.

In the general case, where we have to also define the accessibility relation for $\mathfrak{M}^*$, things are more complicated. We'll call a model $\mathfrak{M}^*$ a *filtration* if its accessibility relation $R^*$ satisfies the conditions required to make the inductive proof above go through. Then any filtration $\mathfrak{M}^*$ will make $\varphi$ true at $[w]$ iff $\mathfrak{M}$ makes $\varphi$ true at $w$. However, now we also have to show that there *are* filtrations, i.e., we can define $R^*$ so that it satisfies the required conditions. In order for this to work, however, we have to require that worlds $u$, $v$ count as equivalent not just when they agree on all propositional variables, but on all sub-formulas of $\varphi$. Since $\varphi$ has only finitely many sub-formulas, this will still guarantee that the filtration is finite. There is not just one way to define a filtration, and in order to make sure that the accessibility relation of the filtration satisfies the required properties (e.g., reflexive, transitive, etc.) we have to be inventive with the definition of $R^*$.

## 42.2 Preliminaries

Filtrations allow us to establish the decidability of our systems of modal logic by showing that they have the *finite model property*, i.e., that any formula that is true (false) in a model is also true (false) in a *finite* model. Filtrations are defined relative to sets of formulas which are closed under subformulas.

**Definition 42.1.** A set $\Gamma$ of formulas is *closed under subformulas* if it contains every subformula of a formula in $\Gamma$. Further, $\Gamma$ is *modally closed* if it is closed under subformulas and moreover $\varphi \in \Gamma$ implies $\Box\varphi, \Diamond\varphi \in \Gamma$.

For instance, given a formula $\varphi$, the set of all its sub-formulas is closed under sub-formulas. When we're defining a filtration of a model through the set of sub-formulas of $\varphi$, it will have the property we're after: it makes $\varphi$ true (false) iff the original model does.

The set of worlds of a filtration of $\mathfrak{M}$ through $\Gamma$ is defined as the set of all equivalence classes of the following equivalence relation.

**Definition 42.2.** Let $\mathfrak{M} = \langle W, R, V\rangle$ and suppose $\Gamma$ is closed under subformulas. Define a relation $\equiv$ on $W$ to hold of any two worlds that make the same formulas from $\Gamma$ true, i.e.:

$$u \equiv v \quad \text{if and only if} \quad \forall \varphi \in \Gamma : \mathfrak{M}, u \Vdash \varphi \Leftrightarrow \mathfrak{N}, v \Vdash \varphi.$$

The equivalence class $[w]_\equiv$ of a world $w$, or $[w]$ for short, is the set of all worlds $\equiv$-equivalent to $w$:

$$[w] = \{v : v \equiv w\}.$$

**Proposition 42.3.** *Given $\mathfrak{M}$ and $\Gamma$, $\equiv$ as defined above is an equivalence relation, i.e., it is reflexive, symmetric, and transitive.*

*Proof.* The relation $\equiv$ is reflexive, since $w$ makes exactly the same formulas from $\Gamma$ true as itself. It is symmetric since if $u$ makes the same formulas from $\Gamma$ true as $v$, the same holds for $v$ and $u$. It is also transitive, since if $u$ makes the same formulas from $\Gamma$ true as $v$, and $v$ as $w$, then $u$ makes the same formulas from $\Gamma$ true as $w$. □

The relation $\equiv$, like any equivalence relation, divides $W$ into *partitions*, i.e., subsets of $W$ which are pairwise disjoint, and together cover all of $W$. Every $w \in W$ is an element of one of the partitions, namely of $[w]$, since $w \equiv w$. So the partitions $[w]$ cover all of $W$. They are pairwise disjoint, for if $u \in [w]$ and $u \in [v]$, then $u \equiv w$ and $u \equiv v$, and by symmetry and transitivity, $w \equiv v$, and so $[w] = [v]$.

## 42.3 Filtrations

Rather than define "the" filtration of $\mathfrak{M}$ through $\Gamma$, we define when a model $\mathfrak{M}^*$ counts as a filtration of $\mathfrak{M}$. All filtrations have the same set of worlds $W^*$ and the same valuation $V^*$. But different filtrations may have different accessibility relations $R^*$. To count as a filtration, $R^*$ has to satisfy a number of conditions, however. These conditions are exactly what we'll require to prove the main result, namely that $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}^*, [w] \Vdash \varphi$, provided $\varphi \in \Gamma$.

**Definition 42.4.** Let $\Gamma$ be closed under subformulas and $\mathfrak{M} = \langle W, R, V\rangle$. A *filtration of $\mathfrak{M}$ through $\Gamma$* is any model $\mathfrak{M}^* = \langle W^*, R^*, V^*\rangle$, where:

1. $W^* = \{[w] : w \in W\}$;
2. For any $u, v \in W$:
   a) If $Ruv$ then $R^*[u][v]$;
   b) If $R^*[u][v]$ then for any $\Box\varphi \in \Gamma$, if $\mathfrak{M}, u \Vdash \Box\varphi$ then $\mathfrak{M}, v \Vdash \varphi$;
   c) If $R^*[u][v]$ then for any $\Diamond\varphi \in \Gamma$, if $\mathfrak{M}, v \Vdash \varphi$ then $\mathfrak{M}, u \Vdash \Diamond\varphi$.
3. $V^*(p) = \{[u] : u \in V(p)\}$.

It's worthwhile thinking about what $V^*(p)$ is: the set consisting of the equivalence classes $[w]$ of all worlds $w$ where $p$ is true in $\mathfrak{M}$. On the one hand, if $w \in V(p)$, then $[w] \in V^*(p)$ by that definition. However, it is not necessarily the case that if $[w] \in V^*(p)$, then $w \in V(p)$. If $[w] \in V^*(p)$ we are only guaranteed that $[w] = [u]$ for *some* $u \in V(p)$. Of course, $[w] = [u]$ means that $w \equiv u$. So, when $[w] \in V^*(p)$ we can (only) conclude that $w \equiv u$ for some $u \in V(p)$.

**Theorem 42.5.** *If $\mathfrak{M}^*$ is a filtration of $\mathfrak{M}$ through $\Gamma$, then for every $\varphi \in \Gamma$ and $w \in W$, we have $\mathfrak{M}, w \Vdash \varphi$ if and only if $\mathfrak{M}^*, [w] \Vdash \varphi$.*

*Proof.* By induction on $\varphi$, using the fact that $\Gamma$ is closed under subformulas. Since $\varphi \in \Gamma$ and $\Gamma$ is closed under sub-formulas, all sub-formulas of $\varphi$ are also $\in \Gamma$. Hence in each inductive step, the induction hypothesis applies to the sub-formulas of $\varphi$.

1. $\varphi \equiv \bot$: Neither $\mathfrak{M}, w \Vdash \varphi$ nor $\mathfrak{M}^*, w \Vdash \varphi$.

2. $\varphi \equiv p$: The left-to-right direction is immediate, as $\mathfrak{M}, w \Vdash \varphi$ only if $w \in V(p)$, which implies $[w] \in V^*(p)$, i.e., $\mathfrak{M}^*, [w] \Vdash \varphi$. Conversely, suppose $\mathfrak{M}^*, [w] \Vdash \varphi$, i.e., $[w] \in V^*(p)$. Then for some $v \in V(p)$, $w \equiv v$. Of course then also $\mathfrak{M}, v \Vdash p$. Since $w \equiv v$, $w$ and $v$ make the same formulas from $\Gamma$ true. Since by assumption $p \in \Gamma$ and $\mathfrak{M}, v \Vdash p$, $\mathfrak{M}, w \Vdash \varphi$.

3. $\varphi \equiv \neg\psi$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \nVdash \psi$. By induction hypothesis, $\mathfrak{M}, w \nVdash \psi$ iff $\mathfrak{M}^*, [w] \nVdash \psi$. Finally, $\mathfrak{M}^*, [w] \nVdash \psi$ iff $\mathfrak{M}^*, [w] \Vdash \varphi$.

4. Exercise.

5. $\varphi \equiv (\psi \vee \chi)$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \Vdash \psi$ or $\mathfrak{M}, w \Vdash \chi$. By induction hypothesis, $\mathfrak{M}, w \Vdash \psi$ iff $\mathfrak{M}^*, [w] \Vdash \psi$, and $\mathfrak{M}, w \Vdash \chi$ iff $\mathfrak{M}^*, [w] \Vdash \chi$. And $\mathfrak{M}^*, [w] \Vdash \varphi$ iff $\mathfrak{M}^*, [w] \Vdash \psi$ or $\mathfrak{M}^*, [w] \Vdash \chi$.

6. Exercise.

7. $\varphi \equiv \Box\psi$: Suppose $\mathfrak{M}, w \Vdash \varphi$; to show that $\mathfrak{M}^*, [w] \Vdash \varphi$, let $v$ be such that $R^*[w][v]$. From Definition 42.4(2b), we have that $\mathfrak{M}, v \Vdash \psi$, and by inductive hypothesis $\mathfrak{M}^*, [v] \Vdash \psi$. Since $v$ was arbitrary, $\mathfrak{M}^*, [w] \Vdash \varphi$ follows.

   Conversely, suppose $\mathfrak{M}^*, [w] \Vdash \varphi$ and let $v$ be arbitrary such that $Rwv$. From Definition 42.4(2a), we have $R^*[w][v]$, so that $\mathfrak{M}^*, [v] \Vdash \psi$; by inductive hypothesis $\mathfrak{M}, v \Vdash \psi$, and since $v$ was arbitrary, $\mathfrak{M}, u \Vdash \varphi$.

8. Exercise.

□

What holds for truth at worlds in a model also holds for truth in a model and validity in a class of models.

**Corollary 42.6.** *Let $\Gamma$ be closed under subformulas. Then:*

1. *If $\mathfrak{M}^*$ is a filtration of $\mathfrak{M}$ through $\Gamma$ then for any $\varphi \in \Gamma$: $\mathfrak{M} \Vdash \varphi$ if and only if $\mathfrak{M}^* \Vdash \varphi$.*

2. *If $\mathcal{C}$ is a class of models and $\Gamma(\mathcal{C})$ is the class of $\Gamma$-filtrations of models in $\mathcal{C}$, then any formula $\varphi \in \Gamma$ is valid in $\mathcal{C}$ if and only if it is valid in $\Gamma(\mathcal{C})$.*

## 42.4 Examples of Filtrations

We have not yet shown that there are any filtrations. But indeed, for any model $\mathfrak{M}$, there are many filtrations of $\mathfrak{M}$ through $\Gamma$. We identify two, in particular: the finest and coarsest filtrations. Filtrations of the same models will differ in their accessibility relation (as Definition 42.4 stipulates directly what $W^*$ and $V^*$ should be). The finest filtration will have as few related worlds as possible, whereas the coarsest will have as many as possible.

**Definition 42.7.** Where $\Gamma$ is closed under subformulas, the *finest* filtration $\mathfrak{M}^*$ of a model $\mathfrak{M}$ is defined by putting:

$$R^*[u][v] \quad \text{if and only if} \quad \exists u' \in [u]\ \exists v' \in [v] : Ru'v'.$$

**Proposition 42.8.** *The finest filtration $\mathfrak{M}^*$ is indeed a filtration.*

*Proof.* We need to check that $R^*$, so defined, satisfies Definition 42.4(2). We check the three conditions in turn.

If $Ruv$ then since $u \in [u]$ and $v \in [v]$, also $R^*[u][v]$, so (2a) is satisfied.

For (2b), suppose $\Box\varphi \in \Gamma$, $R^*[u][v]$, and $\mathfrak{M}, u \Vdash \Box\varphi$. By definition of $R^*$, there are $u' \equiv u$ and $v' \equiv v$ such that $Ru'v'$. Since $u$ and $u'$ agree on $\Gamma$, also $\mathfrak{M}, u' \Vdash \Box\varphi$, so that $\mathfrak{M}, v' \Vdash \varphi$. By closure of $\Gamma$ under sub-formulas, $v$ and $v'$ agree on $\varphi$, so $\mathfrak{M}, v \Vdash \varphi$, as desired.

We leave the verification of (2c) as an exercise. □

**Definition 42.9.** Where $\Gamma$ is closed under subformulas, the *coarsest* filtration $\mathfrak{M}^*$ of a model $\mathfrak{M}$ is defined by putting $R^*[u][v]$ if and only if *both* of the following conditions are met:

1. If $\Box\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \Box\varphi$ then $\mathfrak{M}, v \Vdash \varphi$;
2. If $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \varphi$ then $\mathfrak{M}, u \Vdash \Diamond\varphi$.

**Proposition 42.10.** *The coarsest filtration $\mathfrak{M}^*$ is indeed a filtration.*

*Proof.* Given the definition of $R^*$, the only condition that is left to verify is the implication from $Ruv$ to $R^*[u][v]$. So assume $Ruv$. Suppose $\Box\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \Box\varphi$; then obviously $\mathfrak{M}, v \Vdash \varphi$, and (1) is satisfied. Suppose $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \varphi$. Then $\mathfrak{M}, u \Vdash \Diamond\varphi$ since $Ruv$, and (2) is satisfied. □

**Example 42.11.** Let $W = \mathbb{Z}^+$, $Rnm$ iff $m = n + 1$, and $V(p) = \{2n : n \in \mathbb{N}\}$. The model $\mathfrak{M} = \langle W, R, V\rangle$ is depicted in Figure 42.1. The worlds are 1, 2, etc.; each world can access exactly one other world—its successor, and $p$ is true at all and only the even numbers.

Now let $\Gamma$ be the set of sub-formulas of $\Box p \to p$, i.e., $\{p, \Box p, \Box p \to p\}$. $p$ is true at all and only the even numbers, $\Box p$ is true at all and only the odd numbers, so $\Box p \to p$ is true at all and only the even numbers. In other words,

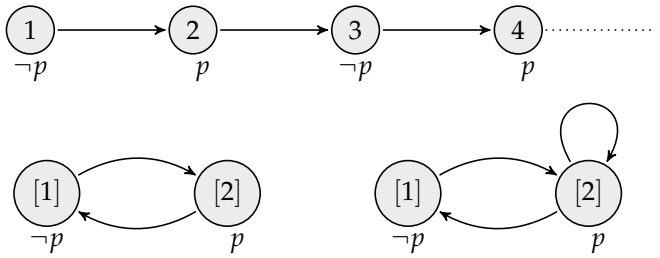



Figure 42.1: An infinite model and its filtrations.

every odd number makes $\Box p$ true and $p$ and $\Box p \to p$ false; every even number makes $p$ and $\Box p \to p$ true, but $\Box p$ false. So $W^* = \{[1], [2]\}$, where $[1] = \{1, 3, 5, \dots\}$ and $[2] = \{2, 4, 6, \dots\}$. Since $2 \in V(p)$, $[2] \in V^*(p)$; since $1 \notin V(p)$, $[1] \notin V^*(p)$. So $V^*(p) = \{[2]\}$.

Any filtration based on $W^*$ must have an accessibility relation that includes $\langle [1], [2] \rangle, \langle [2], [1] \rangle$: since $R12$, we must have $R^*[1][2]$ by Definition 42.4(2a), and since $R23$ we must have $R^*[2][3]$, and $[3] = [1]$. It cannot include $\langle [1], [1] \rangle$: if it did, we'd have $R^*[1][1]$, $\mathfrak{M}, 1 \Vdash \Box p$ but $\mathfrak{M}, 1 \nVdash p$, contradicting (2a). Nothing requires or rules out that $R^*[2][2]$. So, there are two possible filtrations of $\mathfrak{M}$, corresponding to the two accessibility relations

$$\{\langle [1], [2] \rangle, \langle [2], [1] \rangle\} \text{ and } \{\langle [1], [2] \rangle, \langle [2], [1] \rangle, \langle [2], [2] \rangle\}.$$

In either case, $p$ and $\Box p \to p$ are false and $\Box p$ is true at $[1]$; $p$ and $\Box p \to p$ are true and $\Box p$ is false at $[2]$.

## 42.5 Filtrations are Finite

We've defined filtrations for any set $\Gamma$ that is closed under sub-formulas. Nothing in the definition itself guarantees that filtrations are finite. In fact, when $\Gamma$ is infinite (e.g., is the set of all formulas), it may well be infinite. However, if $\Gamma$ is finite (e.g., when it is the set of sub-formulas of a given formula $\varphi$), so is any filtration through $\Gamma$.

**Proposition 42.12.** *If $\Gamma$ is finite then any filtration $\mathfrak{M}^*$ of a model $\mathfrak{M}$ through $\Gamma$ is also finite.*

*Proof.* The size of $W^*$ is the number of different classes $[w]$ under the equivalence relation $\equiv$. Any two worlds $u$, $v$ in such class—that is, any $u$ and $v$ such that $u \equiv v$—agree on all formulas $\varphi$ in $\Gamma$, $\varphi \in \Gamma$ either $\varphi$ is true at both $u$ and $v$, or at neither. So each class $[w]$ corresponds to subset of $\Gamma$, namely the set of all $\varphi \in \Gamma$ such that $\varphi$ is true at the worlds in $[w]$. No two different classes $[u]$ and $[v]$ correspond to the same subset of $\Gamma$. For if the set of formulas true at $u$ and that of formulas true at $v$ are the same, then $u$ and $v$ agree on all formulas

in $\Gamma$, i.e., $u \equiv v$. But then $[u] = [v]$. So, there is an injective function from $W^*$ to $\wp(\Gamma)$, and hence $|W^*| \leq |\wp(\Gamma)|$. Hence if $\Gamma$ contains $n$ sentences, the cardinality of $W^*$ is no greater than $2^n$. □

## 42.6 K and S5 have the Finite Model Property

**Definition 42.13.** A system $\Sigma$ of modal logic is said to have the *finite model property* if whenever a formula $\varphi$ is true at a world in a model of $\Sigma$ then $\varphi$ is true at a world in a *finite* model of $\Sigma$.

**Proposition 42.14.** **K** *has the finite model property.*

*Proof.* **K** is the set of valid formulas, i.e., any model is a model of **K**. By Theorem 42.5, if $\mathfrak{M}\varphi[w]$, then $\mathfrak{M}^*\varphi[w]$ for any filtration of $\mathfrak{M}$ through the set $\Gamma$ of sub-formulas of $\varphi$. Any formula only has finitely many sub-formulas, so $\Gamma$ is finite. By Proposition 42.12, $|W^*| \leq 2^n$, where $n$ is the number of formulas in $\Gamma$. And since **K** imposes no restriction on models, $\mathfrak{M}^*$ is a **K**-model. □

To show that a logic **L** has the finite model property via filtrations it is essential that the filtration of an **L**-model is itself a **L**-model. Often this requires a fair bit of work, and not any filtration yields a **L**-model. However, for universal models, this still holds.

**Proposition 42.15.** *Let $\mathcal{U}$ be the class of universal models (see Proposition 39.14) and $\mathcal{U}_{\text{Fin}}$ the class of all finite universal models. Then any formula $\varphi$ is valid in $\mathcal{U}$ if and only if it is valid in $\mathcal{U}_{\text{Fin}}$.*

*Proof.* Finite universal models are universal models, so the left-to-right direction is trivial. For the right-to left direction, suppose that $\varphi$ is false at some world $w$ in a universal model $\mathfrak{M}$. Let $\Gamma$ contain $\varphi$ as well as all of its sub-formulas; clearly $\Gamma$ is finite. Take a filtration $\mathfrak{M}^*$ of $\mathfrak{M}$; then $\mathfrak{M}^*$ is finite by Proposition 42.12, and by Theorem 42.5, $\varphi$ is false at $[w]$ in $\mathfrak{M}^*$. It remains to observe that $\mathfrak{M}^*$ is also universal: given $u$ and $v$, by hypothesis $Ruv$ and by Definition Definition 42.4(2), also $R^*[u][v]$. □

**Corollary 42.16.** **S5** *has the finite model property.*

*Proof.* By Proposition 39.14, if $\varphi$ is true at a world in some reflexive and euclidean model then it is true at a world in a universal model. By Proposition 42.15, it is true at a world in a finite universal model (namely the filtration of the model through the set of sub-formulas of $\varphi$). Every universal model is also reflexive and euclidean; so $\varphi$ is true at a world in a finite reflexive euclidean model. □

## 42.7 S5 is Decidable

The finite model property gives us an easy way to show that systems of modal logic given by schemas are *decidable* (i.e., that there is a computable procedure to determine whether a formulas is derivable in the system or not).

**Theorem 42.17. S5** *is decidable.*

*Proof.* Let $\varphi$ be given, and suppose the propositional variables occurring in $\varphi$ are among $p_1, \ldots, p_k$. Since for each $n$ there are only finitely many models with $n$ worlds assigning a value to $p_1, \ldots, p_k$, we can enumerate, *in parallel*, all the theorems of **S5** by generating proofs in some systematic way; and all the models containing $1, 2, \ldots$ worlds and checking whether $\varphi$ fails at a world in some such model. Eventually one of the two parallel processes will give an answer, as by Theorem 41.17 and Corollary 42.16, either $\varphi$ is derivable or it fails in a finite universal model. □

The above proof works for **S5** because filtrations of universal models are automatically universal. The same holds for reflexivity and seriality, but more work is needed for other properties.

## 42.8 Filtrations and Properties of Accessibility

As noted, filtrations of universal, serial, and reflexive models are always also universal, serial, or reflexive. But not every filtration of a symmetric or transitive model is symmetric or transitive, respectively. In some cases, however, it is possible to define filtrations so that this does hold. In order to do so, we proceed as in the definition of the coarsest filtration, but add additional conditions to the definition of $R^*$. Let $\Gamma$ be closed under sub-formulas. Consider the relations $C_i(u, v)$ in table 42.1 between worlds $u, v$ in a model $\mathfrak{M} = \langle W, R, V \rangle$. We can define $R * [u][v]$ on the basis of combinations of these conditions. For instance, if we stipulate that $R^*[u][v]$ iff the condition $C_1(u, v)$ holds, we get exactly the coarsest filtration. If we stipulate $R^*[u][v]$ iff both $C_1(u, v)$ and $C_2(u, v)$ hold, we get a different filtration. It is "finer" than the coarsest since fewer pairs of worlds satisfy $C_1(u, v)$ and $C_2(u, v)$ than $C_1(u, v)$ alone.

| | |
|---|---|
| $C_1(u, v)$: | if $\Box\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \Box\varphi$ then $\mathfrak{M}, v \Vdash \varphi$; and<br>if $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \varphi$ then $\mathfrak{M}, u \Vdash \Diamond\varphi$; |
| $C_2(u, v)$: | if $\Box\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \Box\varphi$ then $\mathfrak{M}, u \Vdash \varphi$; and<br>if $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \varphi$ then $\mathfrak{M}, v \Vdash \Diamond\varphi$; |
| $C_3(u, v)$: | if $\Box\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \Box\varphi$ then $\mathfrak{M}, v \Vdash \Box\varphi$; and<br>if $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \Diamond\varphi$ then $\mathfrak{M}, u \Vdash \Diamond\varphi$; |
| $C_4(u, v)$: | if $\Box\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \Box\varphi$ then $\mathfrak{M}, u \Vdash \Box\varphi$; and<br>if $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \Diamond\varphi$ then $\mathfrak{M}, v \Vdash \Diamond\varphi$; |

Table 42.1: Conditions on possible worlds for defining filtrations.

**Theorem 42.18.** *Let $\mathfrak{M} = \langle W, R, P\rangle$ be a model, $\Gamma$ closed under sub-formulas. Let $W^*$ and $V^*$ be defined as in Definition 42.4. Then:*

1. *Suppose $R^*[u][v]$ if and only if $C_1(u,v) \wedge C_2(u,v)$. Then $R^*$ is symmetric, and $\mathfrak{M}^* = \langle W^*, R^*, V^*\rangle$ is a filtration if $\mathfrak{M}$ is symmetric.*

2. *Suppose $R^*[u][v]$ if and only if $C_1(u,v) \wedge C_3(u,v)$. Then $R^*$ is transitive, and $\mathfrak{M}^* = \langle W^*, R^*, V^*\rangle$ is a filtration if $\mathfrak{M}$ is transitive.*

3. *Suppose $R^*[u][v]$ if and only if $C_1(u,v) \wedge C_2(u,v) \wedge C_3(u,v) \wedge C_4(u,v)$. Then $R^*$ is symmetric and transitive, and $\mathfrak{M}^* = \langle W^*, R^*, V^*\rangle$ is a filtration if $\mathfrak{M}$ is symmetric and transitive.*

4. *Suppose $R^*$ is defined as $R^*[u][v]$ if and only if $C_1(u,v) \wedge C_3(u,v) \wedge C_4(u,v)$. Then $R^*$ is transitive and euclidean, and $\mathfrak{M}^* = \langle W^*, R^*, V^*\rangle$ is a filtration if $\mathfrak{M}$ is transitive and euclidean.*

*Proof.* 1. It's immediate that $R^*$ is symmetric, since $C_1(u,v) \Leftrightarrow C_2(v,u)$ and $C_2(u,v) \Leftrightarrow C_1(v,u)$. So it's left to show that if $\mathfrak{M}$ is symmetric then $\mathfrak{M}^*$ is a filtration through $\Gamma$. Condition $C_1(u,v)$ guarantees that (2b) and (2c) of Definition 42.4 are satisfied. So we just have to verify Definition 42.4(2a), i.e., that $Ruv$ implies $R^*[u][v]$.

   So suppose $Ruv$. To show $R^*[u][v]$ we need to establish that $C_1(u,v)$ and $C_2(u,v)$. For $C_1$: if $\Box\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \Box\varphi$ then also $\mathfrak{M}, v \Vdash \varphi$ (since $Ruv$). Similarly, if $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \varphi$ then $\mathfrak{M}, u \Vdash \Diamond\varphi$ since $Ruv$. For $C_2$: if $\Box\varphi \in \Gamma$ and $\mathfrak{M}, v \Vdash \Box\varphi$ then $Ruv$ implies $Rvu$ by symmetry, so that $\mathfrak{M}, u \Vdash \varphi$. Similarly, if $\Diamond\varphi \in \Gamma$ and $\mathfrak{M}, u \Vdash \varphi$ then $\mathfrak{M}, v \Vdash \Diamond\varphi$ (since $Rvu$ by symmetry).

2. Exercise.

3. Exercise.

4. Exercise.

□

## 42.9 Filtrations of Euclidean Models

The approach of section 42.8 does not work in the case of models that are euclidean or serial and euclidean. Consider the model at the top of Figure 42.2, which is both euclidean and serial. Let $\Gamma = \{p, \Box p\}$. When taking a filtration through $\Gamma$, then $[w_1] = [w_3]$ since $w_1$ and $w_3$ are the only worlds that agree on $\Gamma$. Any filtration will also have the arrow inherited from $\mathfrak{M}$, as depicted in Figure 42.3. That model isn't euclidean. Moreover, we cannot add arrows to that model in order to make it euclidean. We would have to add double

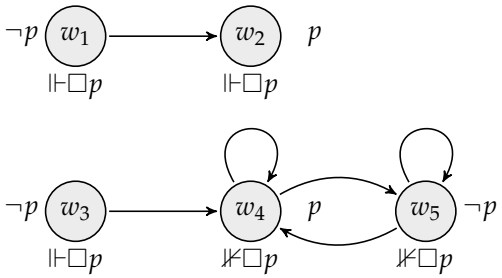



Figure 42.2: A serial and euclidean model.

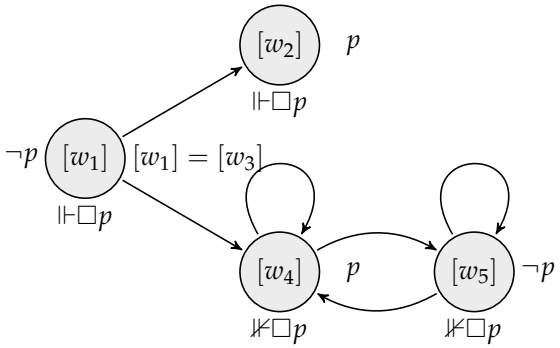



Figure 42.3: The filtration of the model in Figure 42.2.

arrows between $[w_2]$ and $[w_4]$, and then also between $w_2$ and $w_5$. But $\Box p$ is supposed to be true at $w_2$, while $p$ is false at $w_5$.

In particular, to obtain a euclidean flitration it is not enough to consider filtrations through arbitrary $\Gamma$'s closed under sub-formulas. Instead we need to consider sets $\Gamma$ that are *modally closed* (see Definition 42.1). Such sets of sentences are infinite, and therefore do not immediately yield a finite model property or the decidability of the corresponding system.

**Theorem 42.19.** *Let $\Gamma$ be modally closed, $\mathfrak{M} = \langle W, R, V\rangle$, and $\mathfrak{M}^* = \langle W^*, R^*, V^*\rangle$ be a coarsest filtration of $\mathfrak{M}$.*

1. *If $\mathfrak{M}$ is symmetric, so is $\mathfrak{M}^*$.*
2. *If $\mathfrak{M}$ is transitive, so is $\mathfrak{M}^*$.*
3. *If $\mathfrak{M}$ is euclidean, so is $\mathfrak{M}^*$.*

*Proof.* 1. If $\mathfrak{M}^*$ is a coarsest filtration, then by definition $R^*[u][v]$ holds if and only if $C_1(u, v)$. For transitivity, suppose $C_1(u, v)$ and $C_1(v, w)$; we have to show show $C_1(u, w)$. Suppose $\mathfrak{M}, u \Vdash \Box\varphi$; then $\mathfrak{M}, u \Vdash \Box\Box\varphi$

since 4 is valid in all transitive models; since $\Box\Box\varphi \in \Gamma$ by closure, also by $C_1(u,v)$, $\mathfrak{M}, v \Vdash \Box\varphi$ and by $C_1(v,w)$, also $\mathfrak{M}, w \Vdash \varphi$. Suppose $\mathfrak{M}, w \Vdash \varphi$; then $\mathfrak{M}, v \Vdash \Diamond\varphi$ by $C_1(v,w)$, since $\Diamond\varphi \in \Gamma$ by modal closure. By $C_1(u,v)$, we get $\mathfrak{M}, u \Vdash \Diamond\Diamond\varphi$ since $\Diamond\Diamond\varphi \in \Gamma$ by modal closure. Since $4_\Diamond$ is valid in all transitive models, $\mathfrak{M}, u \Vdash \Diamond\varphi$.

2. Exercise. Use the fact that both 5 and $5_\Diamond$ are valid in all euclidean models.

3. Exercise. Use the fact that B and $B_\Diamond$ are valid in all symmetric models.

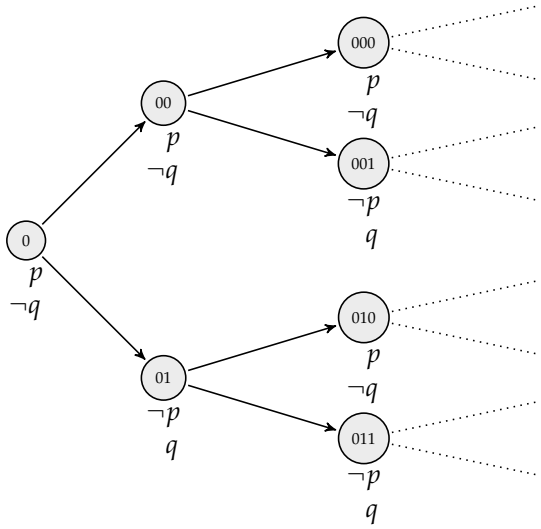
□

## Problems

**Problem 42.1.** Complete the proof of Theorem 42.5

**Problem 42.2.** Complete the proof of Proposition 42.8.

**Problem 42.3.** Consider the following model $\mathfrak{M} = \langle W, R, V\rangle$ where $W = \{0\sigma : \sigma \in \mathbb{B}^*\}$, the set of sequences of 0s and 1s starting with 0, with $R\sigma\sigma'$ iff $\sigma' = \sigma 0$ or $\sigma' = \sigma 1$, and $V(p) = \{\sigma 0 : \sigma \in \mathbb{B}^*\}$ and $V(q) = \{\sigma 1 : \sigma \in \mathbb{B}^* \setminus \{1\}\}$. Here's a picture:




We have $\mathfrak{M}, w \nVdash \Box(p \vee q) \to (\Box p \vee \Box q)$ for every $w$.

Let $\Gamma$ be the set of sub-formulas of $\Box(p \vee q) \to (\Box p \vee \Box q)$. What are $W^*$ and $V^*$? What is the accessibility relation of the finest filtration of $\mathfrak{M}$? Of the coarsest?

**Problem 42.4.** Show that any filtration of a serial or reflexive model is also serial or reflexive (respectively).

**Problem 42.5.** Find a non-symmetric (non-transitive, non-euclidean) filtration of a symmetric (transitive, euclidean) model.

**Problem 42.6.** Show that any filtration of a serial or reflexive model is also serial or reflexive (respectively).

**Problem 42.7.** Find a non-symmetric (non-transitive, non-euclidean) filtration of a symmetric (transitive, euclidean) model.

**Problem 42.8.** Complete the proof of Theorem 42.18.

**Problem 42.9.** Complete the proof of Theorem 42.19.

# Chapter 43

# Modal Tableaux

Draft chapter on prefixed tableaux for modal logic. Needs more examples, completeness proofs, and discussion of how one can find countermodels from unsuccessful searches for closed tableaux.

## 43.1 Introduction

Tableaux are certain (downward-branching) trees of signed formulas, i.e., pairs consisting of a truth value sign ($\mathbb{T}$ or $\mathbb{F}$) and a sentence

$$\mathbb{T}\varphi \text{ or } \mathbb{F}\varphi.$$

A tableau begins with a number of *assumptions*. Each further signed formula is generated by applying one of the inference rules. Some inference rules add one or more signed formulas to a tip of the tree; others add two new tips, resulting in two branches. Rules result in signed formulas where the formula is less complex than that of the signed formula to which it was applied. When a branch contains both $\mathbb{T}\varphi$ and $\mathbb{F}\varphi$, we say the branch is *closed*. If every branch in a tableau is closed, the entire tableau is closed. A closed tableau consititues a derivation that shows that the set of signed formulas which were used to begin the tableau are unsatisfiable. This can be used to define a $\vdash$ relation: $\Gamma \vdash \varphi$ iff there is some finite set $\Gamma_0 = \{\psi_1, \ldots, \psi_n\} \subseteq \Gamma$ such that there is a closed tableau for the assumptions

$$\{\mathbb{F}\varphi, \mathbb{T}\psi_1, \ldots, \mathbb{T}\psi_n\}.$$

For modal logics, we have to both extend the notion of signed formula and add rules that cover $\Box$ and $\Diamond$ In addition to a sign($\mathbb{T}$ or $\mathbb{F}$), formulas in modal tableaux also have *prefixes* $\sigma$. The prefixes are non-empty sequences of positive integers, i.e., $\sigma \in (\mathbb{Z}^+)^* \setminus \{\Lambda\}$. When we write such prefixes without

| | |
|---|---|
| $\dfrac{\sigma\,\mathbb{T}\neg\varphi}{\sigma\,\mathbb{F}\varphi}\ \neg\mathbb{T}$ | $\dfrac{\sigma\,\mathbb{F}\neg\varphi}{\sigma\,\mathbb{T}\varphi}\ \neg\mathbb{F}$ |
| $\dfrac{\sigma\,\mathbb{T}\varphi\wedge\psi}{\begin{array}{c}\sigma\,\mathbb{T}\varphi\\ \sigma\,\mathbb{T}\psi\end{array}}\ \wedge\mathbb{T}$ | $\dfrac{\sigma\,\mathbb{F}\varphi\wedge\psi}{\sigma\,\mathbb{F}\varphi \quad\mid\quad \sigma\,\mathbb{F}\psi}\ \wedge\mathbb{F}$ |
| $\dfrac{\sigma\,\mathbb{T}\varphi\vee\psi}{\sigma\,\mathbb{T}\varphi \quad\mid\quad \sigma\,\mathbb{T}\psi}\ \vee\mathbb{T}$ | $\dfrac{\sigma\,\mathbb{F}\varphi\vee\psi}{\begin{array}{c}\sigma\,\mathbb{F}\varphi\\ \sigma\,\mathbb{F}\psi\end{array}}\ \vee\mathbb{F}$ |
| $\dfrac{\sigma\,\mathbb{T}\varphi\to\psi}{\sigma\,\mathbb{F}\varphi \quad\mid\quad \sigma\,\mathbb{T}\psi}\ \to\mathbb{T}$ | $\dfrac{\sigma\,\mathbb{F}\varphi\to\psi}{\begin{array}{c}\sigma\,\mathbb{T}\varphi\\ \sigma\,\mathbb{F}\psi\end{array}}\ \to\mathbb{F}$ |

Table 43.1: Prefixed tableau rules for the propositional connectives

the surrounding $\langle\ \rangle$, and separate the individual elements by .'s instead of ,'s. If $\sigma$ is a prefix, then $\sigma.n$ is $\sigma \frown \langle n\rangle$; e.g., if $\sigma = 1.2.1$, then $\sigma.3$ is 1.2.1.3. So for instance,

$$1.2\,\mathbb{T}\Box\varphi\to\varphi$$

is a *prefixed signed formula* (or just a *prefixed formula* for short).

Intuitively, the prefix names a world in a model that might satisfy the formulas on a branch of a tableau, and if $\sigma$ names some world, then $\sigma.n$ names a world accessible from (the world named by) $\sigma$.

## 43.2 Rules for K

The rules for the regular propositional connectives are the same as for regular propositional signed tableaux, just with prefixes added. In each case, the rule applied to a signed formula $\sigma\,S\,\varphi$ produces new formulas that are also prefixed by $\sigma$. This should be intuitively clear: e.g., if $\varphi\wedge\psi$ is true at (a world named by) $\sigma$, then $\varphi$ and $\psi$ are true at $\sigma$ (and not at any other world). We collect the propositional rules in section 43.2.

The closure condition is the same as for ordinary tableaux, although we require that not just the formulas but also the prefixes must match. So a branch is closed if it contains both

$$\sigma\,\mathbb{T}\varphi \quad\text{and}\quad \sigma\,\mathbb{F}\varphi$$

for some prefix $\sigma$ and formula $\varphi$.

| | |
|---|---|
| $\dfrac{\sigma\,\mathbb{T}\Box\varphi}{\sigma.n\,\mathbb{T}\varphi}\,\Box\mathbb{T}$ <br> $\sigma.n$ is used | $\dfrac{\sigma\,\mathbb{F}\Box\varphi}{\sigma.n\,\mathbb{F}\varphi}\,\Box\mathbb{F}$ <br> $\sigma.n$ is new |
| $\dfrac{\sigma\,\mathbb{T}\Diamond\varphi}{\sigma.n\,\mathbb{T}\varphi}\,\Diamond\mathbb{T}$ <br> $\sigma.n$ is new | $\dfrac{\sigma\,\mathbb{F}\Diamond\varphi}{\sigma.n\,\mathbb{F}\varphi}\,\Diamond\mathbb{F}$ <br> $\sigma.n$ is used |

Table 43.2: The modal rules for K.

The rules for setting up assumptions is also as for ordinary tableaux, except that for asusmptions we always use the prefix 1. (It does not matter which prefix we use, as long as it's the same for all assumptions.) So, e.g., we say that

$$\psi_1, \ldots, \psi_n \vdash \varphi$$

iff there is a closed tableau for the assumptions

$$1\,\mathbb{T}\psi_1, \ldots, 1\,\mathbb{T}\psi_n, 1\,\mathbb{F}\varphi.$$

For the modal operators $\Box$ and $\Diamond$, the prefix of the conclusion of the rule applied to a formula with prefix $\sigma$ is $\sigma.n$. However, which $n$ is allowed depends on whether the sign is $\mathbb{T}$ or $\mathbb{F}$.

The $\mathbb{T}\Box$ rule extends a branch containing $\sigma\,\mathbb{T}\Box\varphi$ by $\sigma.n\,\mathbb{T}\varphi$. Similarly, the $\mathbb{F}\Diamond$ rule extends a branch containing $\sigma\,\mathbb{F}\Diamond\varphi$ by $\sigma.n\,\mathbb{F}\varphi$. They can only be applied for a prefix $\sigma.n$ which *already* occurs on the branch in which it is applied. Let's call such a prefix "used" (on the branch).

The $\mathbb{F}\Box$ rule extends a branch containing $\sigma\,\mathbb{F}\Box\varphi$ by $\sigma.n\,\mathbb{F}\varphi$. Similarly, the $\mathbb{T}\Diamond$ rule extends a branch containing $\sigma\,\mathbb{T}\Diamond\varphi$ by $\sigma.n\,\mathbb{T}\varphi$. These rules, however, can only be applied for a prefix $\sigma.n$ which *does not* already occur on the branch in which it is applied. We call such prefixes "new" (to the branch).

The rules are given in table 43.2.

The requirements that the restriction that the prefix for $\Box\mathbb{T}$must be used is necessary as otherwise we would count the following as a closed tableau:

| | | |
|---|---|---|
| 1. | $1\,\mathbb{T}\ \Box\varphi$ | Assumption |
| 2. | $1\,\mathbb{F}\ \Diamond\varphi$ | Assumption |
| 3. | $1.1\,\mathbb{T}\ \varphi$ | $\Box\mathbb{T}\,1$ |
| 4. | $1.1\,\mathbb{F}\ \varphi$ | $\Diamond\mathbb{F}\,2$ |
| | $\otimes$ | |

But $\Box\varphi \not\models \Diamond\varphi$, so our proof system would be unsound. Likewise, $\Diamond\varphi \not\models \Box\varphi$, but without the restriction that the prefix for $\Box\mathbb{F}$must be new, this would be a closed tableau:

| | | |
|---|---|---|
| 1. | $1\,\mathbb{T}\ \Diamond\varphi$ | Assumption |
| 2. | $1\,\mathbb{F}\ \Box\varphi$ | Assumption |
| 3. | $1.1\,\mathbb{T}\ \varphi$ | $\Diamond\mathbb{T}\,1$ |
| 4. | $1.1\,\mathbb{F}\ \varphi$ | $\Box\mathbb{F}\,2$ |
| | $\otimes$ | |

## 43.3 Tableaux for K

**Example 43.1.** We give a closed tableau that shows $\vdash (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi)$.

| | | | |
|---|---|---|---|
| 1. | $1\,\mathbb{F}\ (\Box\varphi \wedge \Box\psi) \to \Box(\varphi \wedge \psi)$ | | Assumption |
| 2. | $1\,\mathbb{T}\ \Box\varphi \wedge \Box\psi$ | | $\to\mathbb{T}\,1$ |
| 3. | $1\,\mathbb{F}\ \Box(\varphi \wedge \psi)$ | | $\to\mathbb{T}\,1$ |
| 4. | $1\,\mathbb{T}\ \Box\varphi$ | | $\wedge\mathbb{T}\,2$ |
| 5. | $1\,\mathbb{T}\ \Box\psi$ | | $\wedge\mathbb{T}\,2$ |
| 6. | $1.1\,\mathbb{F}\ \varphi \wedge \psi$ | | $\Box\mathbb{F}\,3$ |
| 7. | $1.1\,\mathbb{F}\ \varphi$ | $1.1\,\mathbb{F}\ \psi$ | $\wedge\mathbb{F}\,6$ |
| 8. | $1.1\,\mathbb{T}\ \varphi$ | $1.1\,\mathbb{T}\ \psi$ | $\Box\mathbb{T}\,4$; $\Box\mathbb{T}\,5$ |
| | $\otimes$ | $\otimes$ | |

**Example 43.2.** We give a closed tableau that shows $\vdash \Diamond(\varphi \vee \psi) \to (\Diamond\varphi \vee \Diamond\psi)$:

| | | | |
|---|---|---|---|
| 1. | $1\,\mathbb{F}\ \Diamond(\varphi \vee \psi) \to (\Diamond\varphi \vee \Diamond\psi)$ | | Assumption |
| 2. | $1\,\mathbb{T}\ \Diamond(\varphi \vee \psi)$ | | $\to\mathbb{T}\,1$ |
| 3. | $1\,\mathbb{F}\ \Diamond\varphi \vee \Diamond\psi$ | | $\to\mathbb{T}\,1$ |
| 4. | $1\,\mathbb{F}\ \Diamond\varphi$ | | $\vee\mathbb{F}\,3$ |
| 5. | $1\,\mathbb{F}\ \Diamond\psi$ | | $\vee\mathbb{F}\,3$ |
| 6. | $1.1\,\mathbb{T}\ \varphi \vee \psi$ | | $\Diamond\mathbb{T}\,2$ |
| 7. | $1.1\,\mathbb{T}\ \varphi$ | $1.1\,\mathbb{T}\ \psi$ | $\vee\mathbb{T}\,6$ |
| 8. | $1.1\,\mathbb{F}\ \varphi$ | $1.1\,\mathbb{F}\ \psi$ | $\Diamond\mathbb{F}\,4$; $\Diamond\mathbb{F}\,5$ |
| | $\otimes$ | $\otimes$ | |

## 43.4 Soundness

This soundness proof reuses the soundness proof for classical propositional logic, i.e., it proves everything from scratch. That's ok if you want a self-contained soundness proof. If you already have seen soundness for ordinary tableau this will be repetitive. It's planned to make it possible to switch between self-contained version and a version building on the non-modal case.

In order to show that prefixed tableau are sound, we have to show that if

$$1\,\mathbb{T}\,\psi_1, \ldots, 1\,\mathbb{T}\,\psi_n, 1\,\mathbb{F}\,\varphi$$

has a closed tableau then $\psi_1, \ldots, \psi_n \vDash \varphi$. It is easier to prove the contrapositive: if for some $\mathfrak{M}$ and world $w$, $\mathfrak{M}, w \Vdash \psi_i$ for all $i = 1, \ldots, n$ but $\mathfrak{M}, w \nVdash \varphi$, then no tableau can close. Such a countermodel shows that the initial assumptions of the tableau are satisfiable. The strategy of the proof is to show that whenever all the prefixed formulas on a tableau branch are satisfiable, any application of a rule results in at least one extended branch that is also satisfiable. Since closed branches are unsatisfiable, any tableau for a satisfiable set of prefixed formulas must have at least one open branch.

In order to apply this strategy in the modal case, we have to extend our definition of "satisfiable" to modal modals and prefixes. With that in hand, however, the proof is straightforward.

**Definition 43.3.** Let $P$ be some set of prefixes, i.e., $P \subseteq (\mathbb{Z}^+)^* \setminus \{\Lambda\}$ and let $\mathfrak{M}$ be a model. A function $f\colon P \to W$ is an *interpretation of* $P$ in $\mathfrak{M}$ if, whenever $\sigma$ and $\sigma.n$ are both in $P$, then $Rf(\sigma)f(\sigma.n)$.

Relative to an interpretation of prefixes $P$ we can define:

1. $\mathfrak{M}$ satisfies $\sigma\,\mathbb{T}\,\varphi$ iff $\mathfrak{M}, f(\sigma) \Vdash \varphi$.
2. $\mathfrak{M}$ satisfies $\sigma\,\mathbb{F}\,\varphi$ iff $\mathfrak{M}, f(\sigma) \nVdash \varphi$.

**Definition 43.4.** Let $\Gamma$ be a set of prefixed formulas, and let $P(\Gamma)$ be the set of prefixes that occur in it. If $f$ is an interpretation of $P(\Gamma)$ in $\mathfrak{M}$, we say that $\mathfrak{M}$ satisfies $\Gamma$ with respect to $f$, $\mathfrak{M}, f \Vdash \Gamma$, if $\mathfrak{M}$ satisfies every prefixed formula in $\Gamma$ with respect to $f$. $\Gamma$ is *satisfiable* iff there is a model $\mathfrak{M}$ and interpretation $f$ of $P(\Gamma)$ such that $\mathfrak{M}, f \Vdash \Gamma$.

**Proposition 43.5.** *If $\Gamma$ contains both $\sigma\,\mathbb{T}\,\varphi$ and $\sigma\,\mathbb{F}\,\varphi$, for some formula $\varphi$ and prefix $\sigma$, then $\Gamma$ is unsatisfiable.*

*Proof.* There cannot be a model $\mathfrak{M}$ and interpretation $f$ of $P(\Gamma)$ such that both $\mathfrak{M}, f(\sigma) \Vdash \varphi$ and $\mathfrak{M}, f(\sigma) \nVdash \varphi$. □

**Theorem 43.6** (Soundness)**.** *If $\Gamma$ has a closed tableau, $\Gamma$ is unsatisfiable.*

*Proof.* We call a branch of a tableau satisfiable iff the set of signed formulas on it is satisfiable, and let's call a tableau satisfiable if it contains at least one satisfiable branch.

We show the following: Extending a satisfiable tableau by one of the rules of inference always results in a satisfiable tableau. This will prove the theorem: any closed tableau results by applying rules of inference to the tableau consisting only of assumptions from $\Gamma$. So if $\Gamma$ were satisfiable, any tableau for it would be satisfiable. A closed tableau, however, is clearly not satisfiable, since all its branches are closed and closed branches are unsatisfiable.

Suppose we have a satisfiable tableau, i.e., a tableau with at least one satisfiable branch. Applying a rule of inference either adds signed formulas to a branch, or splits a branch in two. If the tableau has a satisfiable branch which is not extended by the rule application in question, it remains a satisfiable branch in the extended tableau, so the extended tableau is satisfiable. So we only have to consider the case where a rule is applied to a satisfiable branch.

Let $\Gamma$ be the set of signed formulas on that branch, and let $\sigma\, S\, \varphi \in \Gamma$ be the signed formula to which the rule is applied. If the rule does not result in a split branch, we have to show that the extended branch, i.e., $\Gamma$ together with the conclusions of the rule, is still satisfiable. If the rule results in split branch, we have to show that at least one of the two resulting branches is satisfiable. First, we consider the possible inferences with only one premise.

1. The branch is expanded by applying $\neg\mathbb{T}$ to $\sigma\,\mathbb{T}\neg\psi \in \Gamma$. Then the extended branch contains the signed formulas $\Gamma \cup \{\sigma\,\mathbb{F}\,\psi\}$. Suppose $\mathfrak{M}, f \Vdash \Gamma$. In particular, $\mathfrak{M}, f(\sigma) \Vdash \neg\psi$. Thus, $\mathfrak{M}, f(\sigma) \nVdash \psi$, i.e., $\mathfrak{M}$ satisfies $\sigma\,\mathbb{F}\,\psi$ with respect to $f$.

2. The branch is expanded by applying $\neg\mathbb{F}$ to $\sigma\,\mathbb{F}\neg\psi \in \Gamma$: Exercise.

3. The branch is expanded by applying $\wedge\mathbb{T}$ to $\sigma\,\mathbb{T}\psi \wedge \chi \in \Gamma$, which results in two new signed formulas on the branch: $\sigma\,\mathbb{T}\psi$ and $\sigma\,\mathbb{T}\chi$. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular $\mathfrak{M}, f(\sigma) \Vdash \psi \wedge \chi$. Then $\mathfrak{M}, f(\sigma) \Vdash \psi$ and $\mathfrak{M}, f(\sigma) \Vdash \chi$. This means that $\mathfrak{M}$ satisfies both $\sigma\,\mathbb{T}\psi$ and $\sigma\,\mathbb{T}\chi$ with respect to $f$.

4. The branch is expanded by applying $\vee\mathbb{F}$ to $\mathbb{T}\psi \vee \chi \in \Gamma$: Exercise.

5. The branch is expanded by applying $\rightarrow\mathbb{F}$ to $\sigma\,\mathbb{F}\psi \rightarrow \chi \in \Gamma$: This results in two new signed formulas on the branch: $\sigma\,\mathbb{T}\psi$ and $\sigma\,\mathbb{F}\chi$. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular $\mathfrak{M}, f(\sigma) \nVdash \psi \rightarrow \chi$. Then $\mathfrak{M}, f(\sigma) \Vdash \psi$ and $\mathfrak{M}, f(\sigma) \nVdash \chi$. This means that $\mathfrak{M}, f$ satisfies both $\sigma\,\mathbb{T}\psi$ and $\sigma\,\mathbb{F}\chi$.

6. The branch is expanded by applying $\Box\mathbb{T}$ to $\sigma\,\mathbb{T}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma.n\,\mathbb{T}\psi$ on the branch, for some $\sigma.n \in P(\Gamma)$ (since

$\sigma.n$ must be used). Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular, $\mathfrak{M}, f(\sigma) \Vdash \Box\psi$. Since $f$ is an interpretation of prefixes and both $\sigma, \sigma.n \in P(\Gamma)$, we know that $Rf(\sigma)f(\sigma.n)$. Hence, $\mathfrak{M}, f(\sigma.n) \Vdash \psi$, i.e., $\mathfrak{M}, f$ satisfies $\sigma.n\,\mathbb{T}\psi$.

7. The branch is expanded by applying $\Box\mathbb{F}$ to $\sigma\,\mathbb{F}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma.n\,\mathbb{F}\varphi$, where $\sigma.n$ is a new prefix on the branch, i.e., $\sigma.n \notin P(\Gamma)$. Since $\Gamma$ is satisfiable, there is a $\mathfrak{M}$ and interpretation $f$ of $P(\Gamma)$ such that $\mathfrak{M}, f \vDash \Gamma$, in particular $\mathfrak{M}, f(\sigma) \nVdash \Box\psi$. We have to show that $\Gamma \cup \{\sigma.n\,\mathbb{F}\psi\}$ is satisfiable. To do this, we define an interpretation of $P(\Gamma) \cup \{\sigma.n\}$ as follows:

   Since $\mathfrak{M}, f(\sigma) \nVdash \Box\psi$, there is a $w \in W$ such that $Rf(\sigma)w$ and $\mathfrak{M}, w \nVdash \psi$. Let $f'$ be like $f$, except that $f'(\sigma.n) = w$. Since $f'(\sigma) = f(\sigma)$ and $Rf(\sigma)w$, we have $Rf'(\sigma)f'(\sigma.n)$, so $f'$ is an interpretation of $P(\Gamma) \cup \{\sigma.n\}$. Obviously $\mathfrak{M}, f'(\sigma.n) \nVdash \psi$. Since $f(\sigma') = f'(\sigma')$ for all prefixes $\sigma' \in P(\Gamma)$, $\mathfrak{M}, f' \Vdash \Gamma$. So, $\mathfrak{M}, f'$ satisfies $\Gamma \cup \{\sigma.n\,\mathbb{F}\psi\}$.

Now let's consider the possible inferences with two premises.

1. The branch is expanded by applying $\wedge\mathbb{F}$ to $\sigma\,\mathbb{F}\psi \wedge \chi \in \Gamma$, which results in two branches, a left one continuing through $\sigma\,\mathbb{F}\psi$ and a right one through $\sigma\,\mathbb{F}\chi$. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular $\mathfrak{M}, f(\sigma) \nVdash \psi \wedge \chi$. Then $\mathfrak{M}, f(\sigma) \nVdash \psi$ or $\mathfrak{M}, f(\sigma) \nVdash \chi$. In the former case, $\mathfrak{M}, f$ satisfies $\sigma\,\mathbb{F}\psi$, i.e., the left branch is satisfiable. In the latter, $\mathfrak{M}, f$ satisfies $\sigma\,\mathbb{F}\chi$, i.e., the right branch is satisfiable.

2. The branch is expanded by applying $\vee\mathbb{T}$ to $\mathbb{T}\psi \vee \chi \in \Gamma$: Exercise.

3. The branch is expanded by applying $\rightarrow\mathbb{T}$ to $\mathbb{T}\psi \rightarrow \chi \in \Gamma$: Exercise.

□

**Corollary 43.7.** *If $\Gamma \vdash \varphi$ then $\Gamma \vDash \varphi$.*

*Proof.* If $\Gamma \vdash \varphi$ then for some $\psi_1, \ldots, \psi_n \in \Gamma$, $\Delta = \{1\,\mathbb{F}\varphi, 1\,\mathbb{T}\psi_1, \ldots, 1\,\mathbb{T}\psi_n\}$ has a closed tableau. We want to show that $\Gamma \vDash \varphi$. Suppose not, so for some $\mathfrak{M}$ and $w$, $\mathfrak{M}, w \Vdash \psi_i$ for $i = 1, \ldots, n$, but $\mathfrak{M}, w \nVdash \varphi$. Let $f(1) = w$; then $f$ is an interpretation of $P(\Delta)$ into $\mathfrak{M}$, and $\mathfrak{M}$ satisfies $\Delta$ with respect to $f$. But by Theorem 43.6, $\Delta$ is unsatisfiable since it has a closed tableau, a contradiction. So we must have $\Gamma \vdash \varphi$ after all. □

**Corollary 43.8.** *If $\vdash \varphi$ then $\varphi$ is true in all models.*

## 43.5 Rules for Other Accessibility Relations

In order to deal with logics determined by special accessibility relations, we consider the additional rules in table 43.3.

Adding these rules results in systems that are sound and complete for the logics given in section 43.5.

| | |
|---|---|
| $\dfrac{\sigma\,\mathbb{T}\Box\varphi}{\sigma\,\mathbb{T}\varphi}$ T□ | $\dfrac{\sigma\,\mathbb{F}\Diamond\varphi}{\sigma\,\mathbb{F}\varphi}$ T◇ |
| $\dfrac{\sigma\,\mathbb{T}\Box\varphi}{\sigma\,\mathbb{T}\Diamond\varphi}$ D□ | $\dfrac{\sigma\,\mathbb{F}\Diamond\varphi}{\sigma\,\mathbb{F}\Box\varphi}$ D◇ |
| $\dfrac{\sigma.n\,\mathbb{T}\Box\varphi}{\sigma\,\mathbb{T}\varphi}$ B□ | $\dfrac{\sigma.n\,\mathbb{F}\Diamond\varphi}{\sigma\,\mathbb{F}\varphi}$ B◇ |
| $\dfrac{\sigma\,\mathbb{T}\Box\varphi}{\sigma.n\,\mathbb{T}\Box\varphi}$ 4□<br>$\sigma.n$ is used | $\dfrac{\sigma\,\mathbb{F}\Diamond\varphi}{\sigma.n\,\mathbb{F}\Diamond\varphi}$ 4◇<br>$\sigma.n$ is used |
| $\dfrac{\sigma.n\,\mathbb{T}\Box\varphi}{\sigma\,\mathbb{T}\Box\varphi}$ 4r□ | $\dfrac{\sigma.n\,\mathbb{F}\Diamond\varphi}{\sigma\,\mathbb{F}\Diamond\varphi}$ 4r◇ |

Table 43.3: More modal rules.

| Logic | $R$ is ... | Rules |
|---|---|---|
| **T** = **KT** | reflexive | T□, T◇ |
| **D** = **KD** | serial | D□, D◇ |
| **K4** | transitive | 4□, 4◇ |
| **B** = **KTB** | reflexive, symmetric | T□, T◇ B□, B◇ |
| **S4** = **KT4** | reflexive, transitive | T□, T◇, 4□, 4◇ |
| **S5** = **KT4B** | reflexive, transitive, euclidean | T□, T◇, 4□, 4◇, 4r□, 4r◇ |

Table 43.4: Tableau rules for various modal logics.

## 43.6 Tableaux for Other Logics

**Example 43.9.** We give a closed tableau that shows **S5** $\vdash 5$, i.e., $\Box\varphi \to \Box\Diamond\varphi$.

| | | |
|---|---|---|
| 1. | $1\,\mathbb{F}\ \Box\varphi \to \Box\Diamond\varphi$ | Assumption |
| 2. | $1\,\mathbb{T}\ \Box\varphi$ | $\to\mathbb{F}\,1$ |
| 3. | $1\,\mathbb{F}\ \Box\Diamond\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $1.1\,\mathbb{F}\ \Diamond\varphi$ | $\Box\mathbb{F}\,3$ |
| 5. | $1\,\mathbb{F}\ \Diamond\varphi$ | $4\mathrm{r}\Diamond\,4$ |
| 6. | $1.1\,\mathbb{F}\ \varphi$ | $\Diamond\mathbb{F}\,5$ |
| 7. | $1.1\,\mathbb{T}\ \varphi$ | $\Box\mathbb{T}\,2$ |
| | $\otimes$ | |

## 43.7 Soundness for Additional Rules

We say a rule is sound for a class of models if, whenever a branch in a tableau is satisfiable in a model from that class, the branch resulting from applying the rule is also satisfiable in a model from that class.

**Proposition 43.10.** T$\Box$ *and* T$\Diamond$ *are sound for reflexive models.*

*Proof.* 1. The branch is expanded by applying T$\Box$ to $\sigma\,\mathbb{T}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma\,\mathbb{T}\psi$ on the branch. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular, $\mathfrak{M}, f(\sigma) \Vdash \Box\psi$. Since $R$ is reflexive, we know that $Rf(\sigma)f(\sigma)$. Hence, $\mathfrak{M}, f(\sigma) \Vdash \psi$, i.e., $\mathfrak{M}, f$ satisfies $\sigma\,\mathbb{T}\psi$.

2. The branch is expanded by applying T$\Diamond$ to $\sigma\,\mathbb{F}\Diamond\psi \in \Gamma$: Exercise. □

**Proposition 43.11.** D$\Box$ *and* D$\Diamond$ *are sound for serial models.*

*Proof.* 1. The branch is expanded by applying D$\Box$ to $\sigma\,\mathbb{T}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma\,\mathbb{T}\Diamond\psi$ on the branch. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular, $\mathfrak{M}, f(\sigma) \Vdash \Box\psi$. Since $R$ is serial, there is a $w \in W$ such that $Rf(\sigma)w$. Then $\mathfrak{M}, w \Vdash \psi$, and hence $\mathfrak{M}, f(\sigma) \Vdash \Diamond\psi$. So, $\mathfrak{M}, f$ satisfies $\sigma\,\mathbb{T}\Diamond\psi$.

2. The branch is expanded by applying D$\Diamond$ to $\sigma\,\mathbb{F}\Diamond\psi \in \Gamma$: Exercise. □

**Proposition 43.12.** B$\Box$ *and* B$\Diamond$ *are sound for symmetric models.*

*Proof.* 1. The branch is expanded by applying B$\Box$ to $\sigma.n\,\mathbb{T}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma\,\mathbb{T}\psi$ on the branch. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular, $\mathfrak{M}, f(\sigma.n) \Vdash \Box\psi$. Since $f$ is an interpretation of prefixes on the branch into $\mathfrak{M}$, we know that $Rf(\sigma)f(\sigma.n)$. Since $R$ is symmetric, $Rf(\sigma.n)f(\sigma)$. Since $\mathfrak{M}, f(\sigma.n) \Vdash \Box\psi$, $\mathfrak{M}, f(\sigma) \Vdash \psi$. Hence, $\mathfrak{M}, f$ satisfies $\sigma\,\mathbb{T}\psi$.

2. The branch is expanded by applying B$\Diamond$ to $\sigma.n\,\mathbb{F}\Diamond\psi \in \Gamma$: Exercise. □

**Proposition 43.13.** *4□ and 4◇ are sound for transitive models.*

*Proof.* 1. The branch is expanded by applying 4□ to $\sigma\,\mathbb{T}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma.n\,\mathbb{T}\Box\psi$ on the branch. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular, $\mathfrak{M}, f(\sigma) \Vdash \Box\psi$. Since $f$ is an interpretation of prefixes on the branch into $\mathfrak{M}$ and $\sigma.n$ must be used, we know that $Rf(\sigma)f(\sigma.n)$. Now let $w$ be any world such that $Rf(\sigma.n)w$. Since $R$ is transitive, $Rf(\sigma)w$. Since $\mathfrak{M}, f(\sigma) \Vdash \Box\psi$, $\mathfrak{M}, w \Vdash \psi$. Hence, $\mathfrak{M}, f(\sigma.n) \Vdash \Box\psi$, and $\mathfrak{M}, f$ satisfies $\sigma.n\,\mathbb{T}\Box\psi$.

2. The branch is expanded by applying 4◇ to $\sigma\,\mathbb{F}\Diamond\psi \in \Gamma$: Exercise. □

**Proposition 43.14.** *4r□ and 4r◇ are sound for euclidean models.*

*Proof.* 1. The branch is expanded by applying 4r□ to $\sigma.n\,\mathbb{T}\Box\psi \in \Gamma$: This results in a new signed formula $\sigma\,\mathbb{T}\Box\psi$ on the branch. Suppose $\mathfrak{M}, f \Vdash \Gamma$, in particular, $\mathfrak{M}, f(\sigma.n) \Vdash \Box\psi$. Since $f$ is an interpretation of prefixes on the branch into $\mathfrak{M}$, we know that $Rf(\sigma)f(\sigma.n)$. Now let $w$ be any world such that $Rf(\sigma)w$. Since $R$ is euclidean, $Rf(\sigma.n)w$. Since $\mathfrak{M}, f(\sigma).n \Vdash \Box\psi$, $\mathfrak{M}, w \Vdash \psi$. Hence, $\mathfrak{M}, f(\sigma) \Vdash \Box\psi$, and $\mathfrak{M}, f$ satisfies $\sigma\,\mathbb{T}\Box\psi$.

2. The branch is expanded by applying 4r◇ to $\sigma.n\,\mathbb{F}\Diamond\psi \in \Gamma$: Exercise. □

**Corollary 43.15.** *The tableau systems given in section 43.5 are sound for the respective classes of models.*

## 43.8 Simple Tableaux for S5

**S5** is sound and complete with respect to the class of universal models, i.e., models where every world is accessible from every world. In universal models the accessibility relation doesn't matter: "there is a world $w$ where $\mathfrak{M}, w \Vdash \varphi$" is true if and only if there is such a $w$ that's accessible from $u$. So in **S5**, we can define models as simply a set of worlds and a valuation $V$. This suggests that we should be able to simplify the tableau rules as well. In the general case, we take as prefixes sequences of positive integers, so that we can keep track of which such prefixes name worlds which are accessible from others: $\sigma.n$ names a world accessible from $\sigma$. But in **S5** any world is accessible from any world, so there is no need to so keep track. Instead, we can use positive integers as prefixes. The simplified rules are given in table 43.5.

**Example 43.16.** We give a simplified closed tableau that shows **S5** $\vdash$ 5, i.e., $\Diamond\varphi \to \Box\Diamond\varphi$.

| | |
|---|---|
| $\dfrac{n\,\mathbb{T}\,\Box\varphi}{m\,\mathbb{T}\,\varphi}\,\Box\mathbb{T}$ <br> $m$ is used | $\dfrac{n\,\mathbb{F}\,\Box\varphi}{m\,\mathbb{F}\,\varphi}\,\Box\mathbb{F}$ <br> $m$ is new |
| $\dfrac{n\,\mathbb{T}\,\Diamond\varphi}{m\,\mathbb{T}\,\varphi}\,\Diamond\mathbb{T}$ <br> $m$ is new | $\dfrac{n\,\mathbb{F}\,\Diamond\varphi}{m\,\mathbb{F}\,\varphi}\,\Diamond\mathbb{F}$ <br> $m$ is used |

Table 43.5: Simplified rules for **S5**.

| | | |
|---|---|---|
| 1. | $1\,\mathbb{F}\ \ \Diamond\varphi \to \Box\Diamond\varphi$ | Assumption |
| 2. | $1\,\mathbb{T}\ \ \Diamond\varphi$ | $\to\mathbb{F}\,1$ |
| 3. | $1\,\mathbb{F}\ \ \Box\Diamond\varphi$ | $\to\mathbb{F}\,1$ |
| 4. | $2\,\mathbb{F}\ \ \Diamond\varphi$ | $\Box\mathbb{F}\,3$ |
| 5. | $3\,\mathbb{T}\ \ \varphi$ | $\Diamond\mathbb{T}\,2$ |
| 6. | $3\,\mathbb{F}\ \ \varphi$ | $\Diamond\mathbb{F}\,4$ |
| | $\otimes$ | |

## 43.9 Completeness for K

To show that the method of tableaux is complete, we have to show that whenever there is no closed tableau to show $\Gamma \vdash \varphi$, then $\Gamma \nvDash \varphi$, i.e., there is a countermodel. But "there is no closed tableau" means that every way we could try to construct one has to fail to close. The trick is to see that if every such way fails to close, then a specific, *systematic and exhaustive* way also fails to close. And this systematic and exhaustive way would close if a closed tableau exists. The single tableau will contain, among its open branches, all the information required to define a countermodel. The countermodel given by an open branch in this tableau will contain the all the prefixes used on that branch as the worlds, and a propositional variable $p$ is true at $\sigma$ iff $\sigma\,\mathbb{T}\,p$ occurs on the branch.

**Definition 43.17.** A branch in a tableau is called complete if, whenever it contains a prefixed formula $\sigma\,S\,\varphi$ to which a rule can be applied, it also contains

1. the prefixed formulas that are the corresponding conclusions of the rule, in the case of propositional stacking rules;

2. one of the corresponding conclusion formulas in the case of propositional branching rules;

3. at least one possible conclusion in the case of modal rules that require a new prefix;

4. the corresponding conclusion for every prefix occurring on the branch in the case of modal rules that require a used prefix.

For instance, a complete branch contains $\sigma\,\mathbb{T}\psi$ and $\sigma\,\mathbb{T}\chi$ whenever it contains $\mathbb{T}\psi \wedge \chi$. If it contains $\sigma\,\mathbb{T}\psi \vee \chi$ it contains at least one of $\sigma\,\mathbb{F}\psi$ and $\sigma\,\mathbb{T}\chi$. If it contains $\sigma\,\mathbb{F}\Box$ it also contains $\sigma.n\,\mathbb{F}\Box$ for at least one $n$. And whenever it contains $\sigma\,\mathbb{T}\Box$ it also contains $\sigma.n\,\mathbb{T}\Box$ for every $n$ such that $\sigma.n$ is used on the branch.

**Proposition 43.18.** *Every finite $\Gamma$ has a tableau in which every branch is complete.*

*Proof.* Consider an open branch in a tableau for $\Gamma$. There are finitely many prefixed formulas in the branch to which a rule could be applied. In some fixed order (say, top to bottom), for each of these prefixed formulas for which the conditions (1)–(4) do not already hold, apply the rules that can be applied to it to extend the branch. In some cases this will result in branching; apply the rule at the tip of each resulting branch for all remaining prefixed formulas. Since the number of prefixed formulas is finite, and the number of used prefixes on the branch is finite, this procedure eventually results in (possibly many) branches extending the original branch. Apply the procedure to each, and repeat. But by construction, every branch is closed. □

**Theorem 43.19** (Completeness)**.** *If $\Gamma$ has no closed tableau, $\Gamma$ is satisfiable.*

*Proof.* By the proposition, $\Gamma$ has a tableau in which every branch is complete. Since it has no closed tableau, it thas has a tableau in which at least one branch is open and complete. Let $\Delta$ be the set of prefixed formulas on the branch, and $P(\Delta)$ the set of prefixes occurring in it.

We define a model $\mathfrak{M}(\Delta) = \langle P(\Delta), R, V\rangle$ where the worlds are the prefixes occurring in $\Delta$, the accessibility relation is given by:

$$R\sigma\sigma' \quad \text{iff} \quad \sigma' = \sigma.n \quad \text{for some } n$$

and

$$V(p) = \{\sigma : \sigma\,\mathbb{T}p \in \Delta\}.$$

We show by induction on $\varphi$ that if $\sigma\,\mathbb{T}\varphi \in \Delta$ then $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$, and if $\sigma\,\mathbb{F}\varphi \in \Delta$ then $\mathfrak{M}(\Delta), \sigma \nVdash \varphi$.

1. $\varphi \equiv p$: If $\sigma \mathbb{T} \varphi \in \Delta$ then $\sigma \in V(p)$ (by definition of $V$) and so $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$.

   If $\sigma \mathbb{F} \varphi \in \Delta$ then $\sigma \mathbb{T} \varphi \notin \Delta$, since the branch would otherwise be closed. So $\sigma \notin V(p)$ and thus $\mathfrak{M}(\Delta), \sigma \nVdash \varphi$.

2. $\varphi \equiv \neg\psi$: If $\sigma \mathbb{T} \varphi \in \Delta$, then $\sigma \mathbb{F} \psi \in \Delta$ since the branch is complete. By induction hypothesis, $\mathfrak{M}(\Delta), \sigma \nVdash \psi$ and thus $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$.

   If $\sigma \mathbb{F} \varphi \in \Delta$, then $\sigma \mathbb{T} \psi \in \Delta$ since the branch is complete. By induction hypothesis, $\mathfrak{M}(\Delta), \sigma \Vdash \psi$ and thus $\mathfrak{M}(\Delta), \sigma \nVdash \varphi$.

3. $\varphi \equiv \psi \wedge \varphi$: Exercise.

4. $\varphi \equiv \psi \vee \varphi$: If $\sigma \mathbb{T} \varphi \in \Delta$, then either $\sigma \mathbb{T} \psi \in \Delta$ or $\sigma \mathbb{T} \chi \in \Delta$ since the branch is complete. By induction hypothesis, either $\mathfrak{M}(\Delta), \sigma \Vdash \psi$ or $\mathfrak{M}(\Delta), \sigma \Vdash \chi$. Thus $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$.

   If $\sigma \mathbb{F} \varphi \in \Delta$, then both $\sigma \mathbb{F} \psi \in \Delta$ and $\sigma \mathbb{F} \chi \in \Delta$ since the branch is complete. By induction hypothesis, both $\mathfrak{M}(\Delta), \sigma \nVdash \psi$ and $\mathfrak{M}(\Delta), \sigma \nVdash \psi$. Thus $\mathfrak{M}(\Delta), \sigma \nVdash \varphi$.

5. $\varphi \equiv \psi \to \varphi$: Exercise.

6. $\varphi \equiv \Box\psi$: If $\sigma \mathbb{T} \varphi \in \Delta$, then, since the branch is complete, $\sigma.n \mathbb{T} \psi \in \Delta$ for every $\sigma.n$ used on the branch, i.e., for every $\sigma' \in P(\Delta)$ such that $R\sigma\sigma'$. By induction hypothesis, $\mathfrak{M}(\Delta), \sigma' \Vdash \psi$ for every $\sigma'$ such that $R\sigma\sigma'$. Therefore, $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$.

   If $\sigma \mathbb{F} \varphi \in \Delta$, then for some $\sigma.n$, $\sigma.n \mathbb{F} \psi \in \Delta$ since the branch is complete. By induction hypothesis, $\mathfrak{M}(\Delta), \sigma.n \nVdash \psi$. Since $R\sigma(\sigma.n)$, there is a $\sigma'$ such that $\mathfrak{M}(\Delta), \sigma' \nVdash \psi$. Thus $\mathfrak{M}(\Delta), \sigma \nVdash \varphi$.

7. $\varphi \equiv \Diamond\psi$: Exercise.

Since $\Gamma \subseteq \Delta$, $\mathfrak{M}(\Delta) \Vdash \Gamma$. □

**Corollary 43.20.** *If $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.*

**Corollary 43.21.** *If $\varphi$ is true in all models, then $\vdash \varphi$.*

## 43.10 Countermodels from Tableaux

The proof of the completeness theorem doesn't just show that if $\vDash \varphi$ then $\vdash \varphi$, it also gives us a method for constructing countermodels to $\varphi$ if $\nvDash A$. In the case of **K**, this method constitutes a *decision procedure*. For suppose $\nvDash \varphi$. Then the proof of Proposition 43.18 gives a method for constructing a complete tableau. The method in fact always terminates. The propositional rules for **K** only add prefixed formulas of lower complexity, i.e., each propositional rule need only be applied once on a branch for any signed formula

$\sigma\, S\, \varphi$. New prefixes are only generated by the $\Box\mathbb{F}$ and $\Diamond\mathbb{T}$ rules, and also only have to be applied once (and produce a single new prefix). $\Box\mathbb{T}$ and $\Diamond\mathbb{F}$ have to be applied potentially multiple times, but only once per prefix, and only finitely many new prefixes are generated. So the construction either results in a closed branch or a complete branch after finitely many stages.

Once a tableau with an open complete branch is constructed, the proof of Theorem 43.19 gives us an explict model that satisfies the original set of prefixed formulas. So not only is it the case that if $\Gamma \vDash \varphi$, then a closed tableau exists and $\Gamma \vdash \varphi$, if we look for the closed tableau in the right way and end up with a "complete" tableau, we'll not only know that $\Gamma \nvDash \varphi$ but actually be able to construct a countermodel.

**Example 43.22.** We know that $\nvdash \Box(p \vee q) \to (\Box p \vee \Box q)$. The construction of a tableau begins with:

| | | |
|---|---|---|
| 1. | $1\,\mathbb{F}\ \ \Box(p \vee q) \to (\Box p \vee \Box q)$ ✓ | Assumption |
| 2. | $1\,\mathbb{T}\ \ \Box(p \vee q)$ | $\to\mathbb{F}\,1$ |
| 3. | $1\,\mathbb{F}\ \ \Box p \vee \Box q$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $1\,\mathbb{F}\ \ \Box p$ ✓ | $\vee\mathbb{F}\,3$ |
| 5. | $1\,\mathbb{F}\ \ \Box q$ ✓ | $\vee\mathbb{F}\,3$ |
| 6. | $1.1\,\mathbb{F}\ \ p$ ✓ | $\Box\mathbb{F}\,4$ |
| 7. | $1.2\,\mathbb{F}\ \ q$ ✓ | $\Box\mathbb{F}\,5$ |

The tableau is of course not finished yet. In the next step, we consider the only line without a checkmark: the prefixed formula $1\,\mathbb{T}\Box(p \vee q)$ on line 2. The construction of the closed tableau says to apply the $\Box\mathbb{T}$ rule for every prefix used on the branch, i.e., for both 1.1 and 1.2:

| | | |
|---|---|---|
| 1. | $1\,\mathbb{F}\ \ \Box(p \vee q) \to (\Box p \vee \Box q)$ ✓ | Assumption |
| 2. | $1\,\mathbb{T}\ \ \Box(p \vee q)$ | $\to\mathbb{F}\,1$ |
| 3. | $1\,\mathbb{F}\ \ \Box p \vee \Box q$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $1\,\mathbb{F}\ \ \Box p$ ✓ | $\vee\mathbb{F}\,3$ |
| 5. | $1\,\mathbb{F}\ \ \Box q$ ✓ | $\vee\mathbb{F}\,3$ |
| 6. | $1.1\,\mathbb{F}\ \ p$ ✓ | $\Box\mathbb{F}\,4$ |
| 7. | $1.2\,\mathbb{F}\ \ q$ ✓ | $\Box\mathbb{F}\,5$ |
| 8. | $1.1\,\mathbb{T}\ \ p \vee q$ | $\Box\mathbb{T}\,2$ |
| 9. | $1.2\,\mathbb{T}\ \ p \vee q$ | $\Box\mathbb{T}\,2$ |

Now lines 2, 8, and 9, don't have checkmarks. But no new prefix has been added, so we apply $\vee\mathbb{T}$ to lines 8 and 9, on all resulting branches (as long as they don't close):

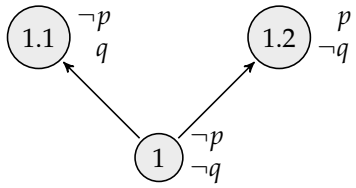



Figure 43.1: A countermodel to $\Box(p \vee q) \to (\Box p \vee \Box q)$.

| | | |
|---|---|---|
| 1. | $1\,\mathbb{F}\ \ \Box(p \vee q) \to (\Box p \vee \Box q)$ ✓ | Assumption |
| 2. | $1\,\mathbb{T}\ \ \Box(p \vee q)$ | $\to\mathbb{F}\,1$ |
| 3. | $1\,\mathbb{F}\ \ \Box p \vee \Box q$ ✓ | $\to\mathbb{F}\,1$ |
| 4. | $1\,\mathbb{F}\ \ \Box p$ ✓ | $\vee\mathbb{F}\,3$ |
| 5. | $1\,\mathbb{F}\ \ \Box q$ ✓ | $\vee\mathbb{F}\,3$ |
| 6. | $1.1\,\mathbb{F}\ \ p$ ✓ | $\Box\mathbb{F}\,4$ |
| 7. | $1.2\,\mathbb{F}\ \ q$ ✓ | $\Box\mathbb{F}\,5$ |
| 8. | $1.1\,\mathbb{T}\ \ p \vee q$ ✓ | $\Box\mathbb{T}\,2$ |
| 9. | $1.2\,\mathbb{T}\ \ p \vee q$ ✓ | $\Box\mathbb{T}\,2$ |
| 10. | $1.1\,\mathbb{T}\ \ p$ ✓ ⊗ \| $1.1\,\mathbb{T}\ \ q$ ✓ | $\vee\mathbb{T}\,8$ |
| 11. | $1.2\,\mathbb{T}\ \ p$ ✓ \| $1.2\,\mathbb{T}\ \ q$ ✓ ⊗ | $\vee\mathbb{T}\,9$ |

There is one remaining open branch, and it is complete. From it we define the model with worlds $W = \{1, 1.1, 1.2\}$ (the only prefixes appearing on the open branch), the accessibility relation $R = \{\langle 1, 1.1\rangle, \langle 1, 1.2\rangle\}$, and the assignment $V(p) = \{1.2\}$ (because line 11 contains $1.2\,\mathbb{T}p$) and $V(q) = \{1.1\}$ (because line 10 contains $1.1\,\mathbb{T}q$). The model is pictured in Figure 43.1, and you can verify that it is a countermodel to $\Box(p \vee q) \to (\Box p \vee \Box q)$.

## Problems

**Problem 43.1.** Find closed tableaux in **K** for the following formulas:

1. $\Box\neg p \to \Box(p \to q)$
2. $(\Box p \vee \Box q) \to \Box(p \vee q)$
3. $\Diamond p \to \Diamond(p \vee q)$

**Problem 43.2.** Complete the proof of Theorem 43.6.

**Problem 43.3.** Give closed tableaux that show the following:

1. **KT5** $\vdash$ B;
2. **KT5** $\vdash$ 4;
3. **KDB4** $\vdash$ T;
4. **KB4** $\vdash$ 5;
5. **KB5** $\vdash$ 4;
6. **KT** $\vdash$ D.

**Problem 43.4.** Complete the proof of Proposition 43.10

**Problem 43.5.** Complete the proof of Proposition 43.11

**Problem 43.6.** Complete the proof of Proposition 43.12

**Problem 43.7.** Complete the proof of Proposition 43.13

**Problem 43.8.** Complete the proof of Proposition 43.14

**Problem 43.9.** Complete the proof of Theorem 43.19.

# Part X

# Intuitionistic Logic

This is a brief introduction to intuitionistic logic produced by Zesen Qian and revised by RZ. It is not yet well integrated with the rest of the text and needs examples and motivations.

# Chapter 44

# Introduction

## 44.1 Constructive Reasoning

In constrast to extensions of classical logic by modal operators or second-order quantifiers, intuitionistic logic is "non-classical" in that it restricts classical logic. Classical logic is *non-constructive* in various ways. Intuitionistic logic is intended to capture a more "constructive" kind of reasoning characteristic of a kind of constructive mathematics. The following examples may serve to illustrate some of the underlying motivations.

Suppose someone claimed that they had determined a natural number $n$ with the property that if $n$ is even, the Riemann hypothesis is true, and if $n$ is odd, the Riemann hypothesis is false. Great news! Whether the Riemann hypothesis is true or not is one of the big open questions of mathematics, and they seem to have reduced the problem to one of calculation, that is, to the determination of whether a specific number is prime or not.

What is the magic value of $n$? They describe it as follows: $n$ is the natural number that is equal to 2 if the Riemann hypothesis is true, and 3 otherwise.

Angrily, you demand your money back. From a classical point of view, the description above does in fact determine a unique value of $n$; but what you really want is a value of $n$ that is given *explicitly*.

To take another, perhaps less contrived example, consider the following question. We know that it is possible to raise an irrational number to a rational power, and get a rational result. For example, $\sqrt{2}^2 = 2$. What is less clear is whether or not it is possible to raise an irrational number to an *irrational* power, and get a rational result. The following theorem answers this in the affirmative:

**Theorem 44.1.** *There are irrational numbers $a$ and $b$ such that $a^b$ is rational.*

*Proof.* Consider $\sqrt{2}^{\sqrt{2}}$. If this is rational, we are done: we can let $a = b = \sqrt{2}$.

Otherwise, it is irrational. Then we have

$$(\sqrt{2}^{\sqrt{2}})^{\sqrt{2}} = \sqrt{2}^{\sqrt{2}\cdot\sqrt{2}} = \sqrt{2}^2 = 2,$$

which is rational. So, in this case, let $a$ be $\sqrt{2}^{\sqrt{2}}$, and let $b$ be $\sqrt{2}$. □

Does this constitute a valid proof? Most mathematicians feel that it does. But again, there is something a little bit unsatisfying here: we have proved the existence of a pair of real numbers with a certain property, without being able to say *which* pair of numbers it is. It is possible to prove the same result, but in such a way that the pair $a$, $b$ *is* given in the proof: take $a = \sqrt{3}$ and $b = \log_3 4$. Then

$$a^b = \sqrt{3}^{\log_3 4} = 3^{1/2\cdot\log_3 4} = (3^{\log_3 4})^{1/2} = 4^{1/2} = 2,$$

since $3^{\log_3 x} = x$.

Intuitionistic logic is designed to capture a kind of reasoning where moves like the one in the first proof are disallowed. Proving the existence of an $x$ satisfying $\varphi(x)$ means that you have to give a specific $x$, and a proof that it satisfies $\varphi$, like in the second proof. Proving that $\varphi$ or $\psi$ holds requires that you can prove one or the other.

Formally speaking, intuitionistic logic is what you get if you restrict a proof system for classical logic in a certain way. From the mathematical point of view, these are just formal deductive systems, but, as already noted, they are intended to capture a kind of mathematical reasoning. One can take this to be the kind of reasoning that is justified on a certain philosophical view of mathematics (such as Brouwer's intuitionism); one can take it to be a kind of mathematical reasoning which is more "concrete" and satisfying (along the lines of Bishop's constructivism); and one can argue about whether or not the formal description captures the informal motivation. But whatever philosophical positions we may hold, we can study intuitionistic logic as a formally presented logic; and for whatever reasons, many mathematical logicians find it interesting to do so.

## 44.2 Syntax of Intuitionistic Logic

The syntax of intuitionistic logic is the same as that for propositional logic. In classical propositional logic it is possible to define connectives by others, e.g., one can define $\varphi \to \psi$ by $\neg\varphi \vee \psi$, or $\varphi \vee \psi$ by $\neg(\neg\varphi \wedge \neg\psi)$. Thus, presentations of classical logic often introduce some connectives as abbreviations for these definitions. This is not so in intuitionistic logic, with two exceptions: $\neg\varphi$ can be—and often is—defined as an abbreviation for $\varphi \to \bot$. Then, of course, $\bot$ must not itself be defined! Also, $\varphi \leftrightarrow \psi$ can be defined, as in classical logic, as $(\varphi \to \psi) \wedge (\psi \to \varphi)$.

Formulas of propositional intuitionistic logic are built up from *propositional variables* and the propositional constant $\bot$ using *logical connectives*. We have:

1. A denumerable set $\mathrm{At}_0$ of propositional variables $p_0$, $p_1$, ...
2. The propositional constant for falsity $\bot$.
3. The logical connectives: $\wedge$ (conjunction), $\vee$ (disjunction), $\to$ (conditional)
4. Punctuation marks: (, ), and the comma.

**Definition 44.2** (Formula)**.** The set $\mathrm{Frm}(\mathcal{L}_0)$ of *formulas* of propositional intuitionistic logic is defined inductively as follows:

1. $\bot$ is an atomic formula.
2. Every propositional variable $p_i$ is an atomic formula.
3. If $\varphi$ and $\psi$ are formulas, then $(\varphi \wedge \psi)$ is a formula.
4. If $\varphi$ and $\psi$ are formulas, then $(\varphi \vee \psi)$ is a formula.
5. If $\varphi$ and $\psi$ are formulas, then $(\varphi \to \psi)$ is a formula.
6. Nothing else is a formula.

In addition to the primitive connectives introduced above, we also use the following *defined* symbols: $\neg$ (negation) and $\leftrightarrow$ (biconditional). Formulas constructed using the defined operators are to be understood as follows:

1. $\neg\varphi$ abbreviates $\varphi \to \bot$.
2. $\varphi \leftrightarrow \psi$ abbreviates $(\varphi \to \psi) \wedge (\psi \to \varphi)$.

Although $\neg$ is officially treated as an abbreviation, we will sometimes give explicit rules and clauses in definitions for $\neg$ as if it were primitive. This is mostly so we can state practice problems.

## 44.3 The Brouwer-Heyting-Kolmogorov Interpretation

> Proofs of validity of intuitionistic propositions using the BHK interpretation are confusing; they have to be explained better.

There is an informal constructive interpretation of the intuitionist connectives, usually known as the Brouwer-Heyting-Kolmogorov interpretation. It uses the notion of a "construction," which you may think of as a constructive proof. (We don't use "proof" in the BHK interpretation so as not to get confused with the notion of a derivation in a formal proof system.) Based on this intuitive notion, the BHK interpretation explains the meanings of the intuitionistic connectives.

1. We assume that we know what constitutes a construction of an atomic statement.
2. A construction of $\varphi_1 \wedge \varphi_2$ is a pair $\langle M_1, M_2 \rangle$ where $M_1$ is a construction of $\varphi_1$ and $M_2$ is a construction of $A_2$.
3. A construction of $\varphi_1 \vee \varphi_2$ is a pair $\langle s, M \rangle$ where $s$ is 1 and $M$ is a construction of $\varphi_1$, or $s$ is 2 and $M$ is a construction of $\varphi_2$.
4. A construction of $\varphi \to \psi$ is a function that converts a construction of $\varphi$ into a construction of $\psi$.
5. There is no construction for $\bot$ (absurdity).
6. $\neg\varphi$ is defined as synonym for $\varphi \to \bot$. That is, a construction of $\neg\varphi$ is a function converting a construction of $\varphi$ into a construction of $\bot$.

**Example 44.3.** Take $\neg\bot$ for example. A construction of it is a function which, given any construction of $\bot$ as input, provides a construction of $\bot$ as output. Obviously, the identity function Id is such a construction: given a construction $M$ of $\bot$, $\mathrm{Id}(M) = M$ yields a construction of $\bot$.

Generally speaking, $\neg\varphi$ means "A construction of $\varphi$ is impossible".

**Example 44.4.** Let us prove $\varphi \to \neg\neg\varphi$ for any proposition $\varphi$, which is $\varphi \to ((\varphi \to \bot) \to \bot)$. The construction should be a function $f$ that, given a construction $M$ of $\varphi$, returns a construction $f(M)$ of $(\varphi \to \bot) \to \bot$. Here is how $f$ constructs the construction of $(\varphi \to \bot) \to \bot$: We have to define a function $g$ which, when given a construction $h$ of $\varphi \to \bot$ as input, outputs a construction of $\bot$. We can define $g$ as follows: apply the input $h$ to the construction $M$ of $\varphi$ (that we received earlier). Since the output $h(M)$ of $h$ is a construction of $\bot$, $f(M)(h) = h(M)$ is a construction of $\bot$ if $M$ is a construction of $\varphi$.

**Example 44.5.** Let us give a construction for $\neg(\varphi \wedge \neg\varphi)$, i.e., $(\varphi \wedge (\varphi \to \bot)) \to \bot$. This is a function $f$ which, given as input a construction $M$ of $\varphi \wedge (\varphi \to \bot)$, yields a construction of $\bot$. A construction of a conjunction $\psi_1 \wedge \psi_2$ is a pair $\langle N_1, N_2 \rangle$ where $N_1$ is a construction of $\psi_1$ and $N_2$ is a construction of $\psi_2$. We can define functions $p_1$ and $p_2$ which recover from a construction of $\psi_1 \wedge \psi_2$ the constructions of $B_1$ and $\psi_2$, respectively:

$$p_1(\langle N_1, N_2 \rangle) = N_1$$
$$p_2(\langle N_1, N_2 \rangle) = N_2$$

Here is what $f$ does: First it applies $p_1$ to its input $M$. That yields a construction of $\varphi$. Then it applies $p_2$ to $M$, yielding a construction of $\varphi \to \bot$. Such a construction, in turn, is a function $p_2(M)$ which, if given as input a construction of $\varphi$, yields a construction of $\bot$. In other words, if we apply $p_2(M)$ to $p_1(M)$, we get a construction of $\bot$. Thus, we can define $f(M) = p_2(p_1(M))$.

**Example 44.6.** Let us give a construction of $((\varphi \wedge \psi) \to \chi) \to (\varphi \to (\psi \to \chi))$, i.e., a function $f$ which turns a construction $g$ of $(\varphi \wedge \psi) \to \chi$ into a construction of $(\varphi \to (\psi \to \chi))$. The construction $g$ is itself a function (from constructions of $\varphi \wedge \psi$ to constructions of $C$). And the output $f(g)$ is a function $h_g$ from constructions of $\varphi$ to functions from constructions of $\psi$ to constructions of $\chi$.

Ok, this is confusing. We have to construct a certain function $h_g$, which will be the output of $f$ for input $g$. The input of $h_g$ is a construction $M$ of $\varphi$. The output of $h_g(M)$ should be a function $k_M$ from constructions $N$ of $\psi$ to constructions of $\chi$. Let $k_{g,M}(N) = g(\langle M, N\rangle)$. Remember that $\langle M, N\rangle$ is a construction of $\varphi \wedge \psi$. So $k_{g,M}$ is a construction of $\psi \to \chi$: it maps constructions $N$ of $\psi$ to constructions of $\chi$. Now let $h_g(M) = k_{g,M}$. That's a function that maps constructions $M$ of $\varphi$ to constructions $k_{g,M}$ of $\psi \to \chi$. Now let $f(g) = h_g$. That's a function that maps constructions $g$ of $(\varphi \wedge \psi) \to \chi$ to constructions of $\varphi \to (\psi \to \chi)$. Whew!

The statement $\varphi \vee \neg\varphi$ is called the Law of Excluded Middle. We can prove it for some specific $\varphi$ (e.g., $\bot \vee \neg\bot$), but not in general. This is because the intuitionistic disjunction requires a construction of one of the disjuncts, but there are statements which currently can neither be proved nor refuted (say, Goldbach's conjecture). However, you can't refute the law of excluded middle either: that is, $\neg\neg(\varphi \vee \neg\varphi)$ holds.

**Example 44.7.** To prove $\neg\neg(\varphi \vee \neg\varphi)$, we need a function $f$ that transforms a construction of $\neg(\varphi \vee \neg\varphi)$, i.e., of $(\varphi \vee (\varphi \to \bot)) \to \bot$, into a construction of $\bot$. In other words, we need a function $f$ such that $f(g)$ is a construction of $\bot$ if $g$ is a construction of $\neg(\varphi \vee \neg\varphi)$.

Suppose $g$ is a construction of $\neg(\varphi \vee \neg\varphi)$, i.e., a function that transforms a construction of $\varphi \vee \neg\varphi$ into a construction of $\bot$. A construction of $\varphi \vee \neg\varphi$ is a pair $\langle s, M\rangle$ where either $s = 1$ and $M$ is a construction of $\varphi$, or $s = 2$ and $M$ is a construction of $\neg\varphi$. Let $h_1$ be the function mapping a construction $M_1$ of $\varphi$ to a construction of $\varphi \vee \neg\varphi$: it maps $M_1$ to $\langle 1, M_2\rangle$. And let $h_2$ be the function mapping a construction $M_2$ of $\neg\varphi$ to a construction of $\varphi \vee \neg\varphi$: it maps $M_2$ to $\langle 2, M_2\rangle$.

Let $k$ be $g \circ h_1$: it is a function which, if given a construction of $\varphi$, returns a construction of $\bot$, i.e., it is a construction of $\varphi \to \bot$ or $\neg\varphi$. Now let $l$ be $g \circ h_2$. It is a function which, given a construction of $\neg\varphi$, provides a construction of $\bot$. Since $k$ is a construction of $\neg\varphi$, $l(k)$ is a construction of $\bot$.

Together, what we've done is describe how we can turn a construction $g$ of $\neg(\varphi \vee \neg\varphi)$ into a construction of $\bot$, i.e., the function $f$ mapping a construction $g$ of $\neg(\varphi \vee \neg\varphi)$ to the construction $l(k)$ of $\bot$ is a construction of $\neg\neg(\varphi \vee \neg\varphi)$.

As you can see, using the BHK interpretation to show the intuitionistic validity of formulas quickly becomes cumbersome and confusing. Luckily,

there are better derivation systems for intuitionistic logic, and more precise semantic interpretations.

## 44.4 Natural Deduction

Natural deduction without the $\bot_C$ rules is a standard derivation system for intuitionistic logic. We repeat the rules here and indicate the motivation using the BHK interpretation. In each case, we can think of a rule which allows us to conclude that if the premises have constructions, so does the conclusion.

Since natural deduction derivations have undischarged assumptions, we should consider such a derivation, say, of $\varphi$ from undischarged assumptions $\Gamma$, as a function that turns constructions of all $\psi \in \Gamma$ into a construction of $\varphi$. If there is a derivation of $\varphi$ from no undischarged assumptions, then there is a construction of $\varphi$ in the sense of the BHK interpretation. For the purpose of the discussion, however, we'll suppress the $\Gamma$ when not needed.

An assumption $\varphi$ by itself is a derivation of $\varphi$ from the undischarged assumption $\varphi$. This agrees with the BHK-interpretation: the identity function on constructions turns any construction of $\varphi$ into a construction of $\varphi$.

### Conjunction

$$\frac{\varphi_1 \qquad \varphi_2}{\varphi_1 \wedge \varphi_2}\,\wedge\text{Intro} \qquad\qquad \frac{\varphi_1 \wedge \varphi_2}{\varphi_i}\,\wedge\text{Elim}_i \quad i \in \{1,2\}$$

Suppose we have constructions $N_1$, $N_2$ of $\varphi_1$ and $\varphi_2$, respectively. Then we also have a construction $\varphi_1 \wedge \varphi_2$, namely the pair $\langle N_1, N_2 \rangle$.

A construction of $\varphi_1 \wedge \varphi_1$ on the BHK interpretation is a pair $\langle N_1, N_2 \rangle$. So assume we have such a pair. Then we also have a construction of each conjunct: $N_1$ is a construction of $\varphi_1$ and $N_2$ is a construction of $\varphi_2$.

### Conditional

$$\begin{array}{c}[\varphi]^u \\ \vdots \\ \psi \\ \hline \varphi \to \psi\end{array}\,{}^{\text{u}}\,\to\text{Intro} \qquad\qquad \frac{\varphi \to \psi \qquad \varphi}{\psi}\,\to\text{Elim}$$

If we have a derivation of $\psi$ from undischarged assumption $\varphi$, then there is a function $f$ that turns constructions of $\varphi$ into constructions of $\psi$. That same

function is a construction of $\varphi \to \psi$. So, if the premise of $\to$Intro has a construction conditional on a construction of $\varphi$, the conclusion $\varphi \to \psi$ has a construction.

On the other hand, suppose there are constructions $N$ of $\varphi$ and $f$ of $\varphi \to \psi$. A construction of $\varphi \to \psi$ is a function that turns constructions of $\varphi$ into constructions of $\psi$. So, $f(N)$ is a construction of $\psi$, i.e., the conclusion of $\to$Elim has a construction.

### Disjunction

$$\frac{\varphi_i}{\varphi_1 \vee \varphi_2} \vee\text{Intro}_i \qquad i \in \{1, 2\}$$

$$\dfrac{\varphi_1 \vee \varphi_2 \quad \overset{[\varphi_1]^u}{\underset{\chi}{\vdots}} \quad \overset{[\varphi_2]^u}{\underset{\chi}{\vdots}}}{\chi}\, u \vee\text{Elim}$$

If we have a construction $N_i$ of $\varphi_i$ we can turn it into a construction $\langle i, N_i \rangle$ of $\varphi_1 \vee \varphi_2$. On the other hand, suppose we have a construction of $\varphi_1 \vee \varphi_2$, i.e., a pair $\langle i, N_i \rangle$ where $N_i$ is a construction of $\varphi_i$, and also functions $f_1$, $f_2$, which turn constructions of $\varphi_1$, $\varphi_2$, respectively, into constructions of $\chi$. Then $f_i(N_i)$ is a construction of $\chi$, the conclusion of $\vee$Elim.

### Absurdity

$$\frac{\bot}{\varphi} \bot_I$$

If we have a derivation of $\bot$ from undischarged assumptions $\psi_1, \ldots, \psi_n$, then there is a function $f(M_1, \ldots, M_n)$ that turns constructions of $\psi_1, \ldots, \psi_n$ into a construction of $\bot$. Since $\bot$ has no construction, there cannot be any constructions of all of $\psi_1, \ldots, \psi_n$ either. Hence, $f$ also has the property that *if* $M_1, \ldots, M_n$ are constructions of $\psi_1, \ldots, \psi_n$, respectively, *then* $f(M_1, \ldots, M_n)$ is a construction of $\varphi$.

### Rules for $\neg$

Since $\neg\varphi$ is defined as $\varphi \to \bot$, we strictly speaking do not need rules for $\neg$. But if we did, this is what they'd look like:

$$\dfrac{\begin{array}{c}[\varphi]^n\\ \vdots\\ \bot\end{array}}{\neg\varphi}\,{}^{n}\neg\text{Intro} \qquad\qquad \dfrac{\neg\varphi \qquad \varphi}{\bot}\,\neg\text{Elim}$$

### Examples of Derivations

1. $\vdash \varphi \to (\neg\varphi \to \bot)$, i.e., $\vdash \varphi \to ((\varphi \to \bot) \to \bot)$

$$\dfrac{\dfrac{\dfrac{[\varphi]^2 \qquad [\varphi \to \bot]^1}{\bot}\to\text{Elim}}{(\varphi \to \bot) \to \bot}\,{}^{1}\to\text{Intro}}{\varphi \to (\varphi \to \bot) \to \bot}\,{}^{2}\to\text{Intro}$$

2. $\vdash ((\varphi \wedge \psi) \to \chi) \to (\varphi \to (\psi \to \chi))$

$$\dfrac{\dfrac{\dfrac{\dfrac{[(\varphi \wedge \psi) \to \chi]^3 \qquad \dfrac{[\varphi]^2 \qquad [\psi]^1}{\varphi \wedge \psi}\wedge\text{Intro}}{\chi}\to\text{Elim}}{\psi \to \chi}\,{}^{1}\to\text{Intro}}{\varphi \to (\psi \to \chi)}\,{}^{2}\to\text{Intro}}{((\varphi \wedge \psi) \to \chi) \to (\varphi \to (\psi \to \chi))}\,{}^{3}\to\text{Intro}$$

3. $\vdash \neg(\varphi \wedge \neg\varphi)$, i.e., $\vdash (\varphi \wedge (\varphi \to \bot)) \to \bot$

$$\dfrac{\dfrac{\dfrac{[\varphi \wedge (\varphi \to \bot)]^1}{\varphi \to \bot}\wedge\text{Elim} \qquad \dfrac{[\varphi \wedge (\varphi \to \bot)]^1}{\varphi}\wedge\text{Elim}}{\bot}\to\text{Elim}}{(\varphi \wedge (\varphi \to \bot)) \to \bot}\,{}^{1}\to\text{Intro}$$

4. $\vdash \neg\neg(\varphi \vee \neg\varphi)$, i.e., $\vdash ((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot$

$$\dfrac{\dfrac{[(\varphi \vee (\varphi \to \bot)) \to \bot]^2 \qquad \dfrac{\dfrac{\dfrac{[\varphi]^1}{\varphi \vee (\varphi \to \bot)}\vee\text{Intro}}{\bot}\to\text{Elim}}{\varphi \to \bot}\,{}^{1}\to\text{Intro}}{\varphi \vee (\varphi \to \bot)}\vee\text{Intro}}{\bot}\to\text{Elim}}{((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot}\,{}^{2}\to\text{Intro}$$

**Proposition 44.8.** *If $\Gamma \vdash \varphi$ in intuitionistic logic, $\Gamma \vdash \varphi$ in classical logic. In particular, if $\varphi$ is an intuitionistic theorem, it is also a classical theorem.*

*Proof.* Every natural deduction rule is also a rule in classical natural deduction, so every derivation in intuitionistic logic is also a derivation in classical logic. □

## 44.5 Axiomatic Derivations

Axiomatic derivations for intuitionistic propositional logic are the conceptually simplest, and historically first, derivation systems. They work just as in classical propositional logic.

**Definition 44.9** (Derivability)**.** If $\Gamma$ is a set of formulas of $\mathcal{L}$ then a *derivation* from $\Gamma$ is a finite sequence $\varphi_1, \ldots, \varphi_n$ of formulas where for each $i \leq n$ one of the following holds:

1. $\varphi_i \in \Gamma$; or
2. $\varphi_i$ is an axiom; or
3. $\varphi_i$ follows from some $\varphi_j$ and $\varphi_k$ with $j < i$ and $k < i$ by modus ponens, i.e., $\varphi_k \equiv \varphi_j \to \varphi_i$.

**Definition 44.10** (Axioms)**.** The set of $\mathrm{Ax}_0$ of *axioms* for the intuitionistic propositional logic are all formulas of the following forms:

$$(\varphi \wedge \psi) \to \varphi \tag{44.1}$$
$$(\varphi \wedge \psi) \to \psi \tag{44.2}$$
$$\varphi \to (\psi \to (\varphi \wedge \psi)) \tag{44.3}$$
$$\varphi \to (\varphi \vee \psi) \tag{44.4}$$
$$\varphi \to (\psi \vee \varphi) \tag{44.5}$$
$$(\varphi \to \chi) \to ((\psi \to \chi) \to ((\varphi \vee \psi) \to \chi)) \tag{44.6}$$
$$\varphi \to (\psi \to \varphi) \tag{44.7}$$
$$(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi)) \tag{44.8}$$
$$\bot \to \varphi \tag{44.9}$$

**Definition 44.11** (Derivability)**.** A formula $\varphi$ is *derivable* from $\Gamma$, written $\Gamma \vdash \varphi$, if there is a derivation from $\Gamma$ ending in $\varphi$.

**Definition 44.12** (Theorems)**.** A formula $\varphi$ is a *theorem* if there is a derivation of $\varphi$ from the empty set. We write $\vdash \varphi$ if $\varphi$ is a theorem and $\nvdash \varphi$ if it is not.

**Proposition 44.13.** *If $\Gamma \vdash \varphi$ in intuitionistic logic, $\Gamma \vdash \varphi$ in classical logic. In particular, if $\varphi$ is an intuitionistic theorem, it is also a classical theorem.*

*Proof.* Every intuitionistic axiom is also a classical axiom, so every derivation in intuitionistic logic is also a derivation in classical logic. □

## Problems

# Chapter 45

# Semantics

This chapter collects definitions for semantics for intuitionistic logic. So far only Kripke and topological semantics are covered. There are no examples yet, either of how models make formulas true or of proofs that formulas are valid.

## 45.1 Introduction

No logic is satisfactorily described without a semantics, and intuitionistic logic is no exception. Whereas for classical logic, the semantics based on valuations is canonical, there are several competing semantics for intuitionistic logic. None of them are completely satisfactory in the sense that they give an intuitionistically acceptable account of the meanings of the connectives.

The semantics based on relational models, similar to the semantics for modal logics, is perhaps the most popular one. In this semantics, propositional variables are assigned to worlds, and these worlds are related by an accessibility relation. That relation is always a partial order, i.e., it is reflexive, antisymmetric, and transitive.

Intuitively, you might think of these worlds as states of knowledge or "evidentiary situations." A state $w'$ is accessible from $w$ iff, for all we know, $w'$ is a possible (future) state of knowledge, i.e., one that is compatible with what's known at $w$. Once a proposition is known, it can't become un-known, i.e., whenever $\varphi$ is known at $w$ and $Rww'$, $\varphi$ is known at $w'$ as well. So "knowledge" is monotonic with respect to the accessibility relation.

If we define "$\varphi$ is known" as in epistemic logic as "true in all epistemic alternatives," then $\varphi \wedge \psi$ is known at $w$ if in all epistemic alternatives, both $\varphi$ and $\psi$ are known. But since knowledge is monotonic and $R$ is reflexive, that means that $\varphi \wedge \psi$ is known at $w$ iff $\varphi$ and $\psi$ are known at $w$. For the same

reason, $\varphi \vee \psi$ is known at $w$ iff at least one of them is known. So for $\wedge$ and $\vee$, the truth conditions of the connectives coincide with those in classical logic.

The truth conditions for the conditional, however, differ from classical logic. $\varphi \to \psi$ is known at $w$ iff at no $w'$ with $Rww'$, $\varphi$ is known without $\psi$ also being known. This is not the same as the condition that $\varphi$ is unknown or $\psi$ is known at $w$. For if we know neither $\varphi$ nor $\psi$ at $w$, there might be a future epistemic state $w'$ with $Rww'$ such that at $w'$, $\varphi$ is known without also coming to know $\psi$.

We know $\neg\varphi$ only if there is no possible future epistemic state in which we know $\varphi$. Here the idea is that if $\varphi$ were knowable, then in some possible future epistemic state $\varphi$ becomes known. Since we can't know $\bot$, in that future epistemic state, we would know $\varphi$ but not know $\bot$.

On this interpretation the principle of excluded middle fails. For there are some $\varphi$ which we don't yet know, but which we might come to know. For such an $\varphi$, both $\varphi$ and $\neg\varphi$ are unknown, so $\varphi \vee \neg\varphi$ is not known. But we do know, e.g., that $\neg(\varphi \wedge \neg\varphi)$. For no future state in which we know both $\varphi$ and $\neg\varphi$ is possible, and we know this independently of whether or not we know $\varphi$ or $\neg\varphi$.

Relational models are not the only available semantics for intuitionistic logic. The topological semantics is another: here propositions are interpreted as open sets in a topological space, and the connectives are interpreted as operations on these sets (e.g., $\wedge$ corresponds to intersection).

## 45.2 Relational models

In order to give a precise semantics for intuitionistic propositional logic, we have to give a definition of what counts as a model relative to which we can evaluate formulas. On the basis of such a definition it is then also possible to define semantics notions such as validity and entailment. One such semantics is given by relational models.

**Definition 45.1.** A relational model for intuitionistic propositional logic is a triple $\mathfrak{M} = \langle W, R, V \rangle$, where

1. $W$ is a non-empty set,
2. $R$ is a reflexive and transitive binary relation on $W$, and
3. $V$ is function assigning to each propositional variable $p$ a subset of $W$, such that
4. $V$ is monotone with respect to $R$, i.e., if $w \in V(p)$ and $Rww'$, then $w' \in V(p)$.

**Definition 45.2.** We define the notion of $\varphi$ *being true at* $w$ *in* $\mathfrak{M}$, $\mathfrak{M}, w \Vdash \varphi$, inductively as follows:

1. $\varphi \equiv p$: $\mathfrak{M}, w \Vdash \varphi$ iff $w \in V(p)$.
2. $\varphi \equiv \bot$: not $\mathfrak{M}, w \Vdash \varphi$.
3. $\varphi \equiv \neg\psi$: $\mathfrak{M}, w \Vdash \varphi$ iff for no $w'$ such that $Rww'$, $\mathfrak{M}, w' \Vdash \psi$.
4. $\varphi \equiv \psi \wedge \chi$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \Vdash \psi$ and $\mathfrak{M}, w \Vdash \chi$.
5. $\varphi \equiv \psi \vee \chi$: $\mathfrak{M}, w \Vdash \varphi$ iff $\mathfrak{M}, w \Vdash \psi$ or $\mathfrak{M}, w \Vdash \chi$ (or both).
6. $\varphi \equiv \psi \to \chi$: $\mathfrak{M}, w \Vdash \varphi$ iff for every $w'$ such that $Rww'$, not $\mathfrak{M}, w \Vdash \psi$ or $\mathfrak{M}, w \Vdash \chi$ (or both).

We write $\mathfrak{M}, w \nVdash \varphi$ if not $\mathfrak{M}, w \Vdash \varphi$. If $\Gamma$ is a set of formulas, $\mathfrak{M}, w \Vdash \Gamma$ means $\mathfrak{M}, w \Vdash \psi$ for all $\psi \in \Gamma$.

**Proposition 45.3.** *Truth at worlds is monotonic with respect to $R$, i.e., if $\mathfrak{M}, w \Vdash \varphi$ and $Rww'$, then $\mathfrak{M}, w' \Vdash \varphi$.*

*Proof.* Exercise. □

## 45.3 Semantic Notions

**Definition 45.4.** We say $\varphi$ is *true in the model* $\mathfrak{M} = \langle W, R, V, w_0 \rangle$, $\mathfrak{M} \Vdash \varphi$, iff $\mathfrak{M}, w \Vdash \varphi$ for all $w \in W$. $\varphi$ is *valid*, $\vDash \varphi$, iff it is true in all models. We say a set of formulas $\Gamma$ *entails* $\varphi$, $\Gamma \vDash \varphi$, iff for every model $\mathfrak{M}$ and every $w$ such that $\mathfrak{M}, w \Vdash \Gamma$, $\mathfrak{M}, w \Vdash \varphi$.

**Proposition 45.5.** 1. *If $\mathfrak{M}, w \Vdash \Gamma$ and $\Gamma \vDash \varphi$, then $\mathfrak{M}, w \Vdash \varphi$.*

2. *If $\mathfrak{M} \Vdash \Gamma$ and $\Gamma \vDash \varphi$, then $\mathfrak{M} \Vdash \varphi$.*

*Proof.* 1. Suppose $\mathfrak{M} \Vdash \Gamma$. Since $\Gamma \vDash \varphi$, we know that if $\mathfrak{M}, w \Vdash \Gamma$, then $\mathfrak{M}, w \Vdash \varphi$. Since $\mathfrak{M}, u \Vdash \Gamma$ for all every $u \in W$, $\mathfrak{M}, w \Vdash \Gamma$. Hence $\mathfrak{M}, w \Vdash \varphi$.

2. Follows immediately from (1).

□

## 45.4 Topological Semantics

Another way to provide a semantics for intuitionistic logic is using the mathematical concept of a topology.

**Definition 45.6.** Let $X$ be a set. A *topology on* $X$ is a set $\mathcal{O} \subseteq \wp(X)$ that satisfies the properties below. The elements of $\mathcal{O}$ are called the *open sets* of the topology. The set $X$ together with $\mathcal{O}$ is called a *topological space*.

1. The empty set and the entire space open: $\emptyset, X \in \mathcal{O}$.

2. Open sets are closed under finite intersections: if $U, V \in \mathcal{O}$ then $U \cap V \in \mathcal{O}$
3. Open sets are closed under arbitrary unions: if $U_i \in \mathcal{O}$ for all $i \in I$, then $\bigcup\{U_i : i \in I\} \in \mathcal{O}$.

We may write $X$ for a topology if the collection of open sets can be inferred from the context; note that, still, only after $X$ is endowed with open sets can it be called a topology.

**Definition 45.7.** A *topological model* of intuitionistic propositional logic is a triple $\mathfrak{X} = \langle X, \mathcal{O}, V\rangle$ where $\mathcal{O}$ is a topology on $X$ and $V$ is a function assigning an open set in $\mathcal{O}$ to each propositional variable.

Given a topological model $\mathfrak{X}$, we can define $[\varphi]_{\mathfrak{X}}$ inductively as follows:

1. $V(\bot) = \emptyset$
2. $[p]_{\mathfrak{X}} = V(p)$
3. $[\varphi \wedge \psi]_{\mathfrak{X}} = [\varphi]_{\mathfrak{X}} \cap [\psi]_{\mathfrak{X}}$
4. $[\varphi \vee \psi]_{\mathfrak{X}} = [\varphi]_{\mathfrak{X}} \cup [\psi]_{\mathfrak{X}}$
5. $[\varphi \to \psi]_{\mathfrak{X}} = \mathrm{Int}((X \setminus [\varphi]_{\mathfrak{X}}) \cup [\psi]_{\mathfrak{X}})$

Here, $\mathrm{Int}(V)$ is the function that maps a set $V \subseteq X$ to its *interior*, that is, the union of all open sets it contains. In other words,

$$\mathrm{Int}(V) = \bigcup\{U : U \subseteq V \text{ and } U \in \mathcal{O}\}.$$

Note that the interior of any set is always open, since it is a union of open sets. Thus, $[\varphi]_{\mathfrak{X}}$ is always an open set.

Although topological semantics is highly abstract, there are ways to think about it that might motivate it. Suppose that the elements, or "points," of $X$ are points at which statements can be evaluated. The set of all points where $\varphi$ is true is the proposition expressed by $\varphi$. Not every set of points is a potential proposition; only the elements of $\mathcal{O}$ are. $\varphi \vDash \psi$ iff $\psi$ is true at every point at which $\varphi$ is true, i.e., $[\varphi]_{\mathfrak{X}} \subseteq [\psi]_{\mathfrak{X}}$, for all $X$. The absurd statement $\bot$ is never true, so $[\bot]_{\mathfrak{X}} = \emptyset$. How must the propositions expressed by $\psi \wedge \chi$, $\psi \vee \chi$, and $\psi \to \chi$ be related to those expressed by $\psi$ and $\chi$ for the intuitionistically valid laws to hold, i.e., so that $\varphi \vdash \psi$ iff $[\varphi]_{\mathfrak{X}} \subset [\psi]_{\mathfrak{X}}$. $\bot \vdash \varphi$ for any $\varphi$, and only $\emptyset \subseteq U$ for all $U$. Since $\psi \wedge \chi \vdash \psi$, $[\psi \wedge \chi]_{\mathfrak{X}} \subseteq [\psi]_{\mathfrak{X}}$, and similarly $[\psi \wedge \chi]_{\mathfrak{X}} \subseteq [\chi]_{\mathfrak{X}}$. The largest set satisfying $W \subseteq U$ and $W \subseteq V$ is $U \cap V$. Conversely, $\psi \vdash \psi \vee \chi$ and $\chi \vdash \psi \vee \chi$, and so $[\psi]_{\mathfrak{X}} \subseteq [\psi \vee \chi]_{\mathfrak{X}}$ and $[\chi]_{\mathfrak{X}} \subseteq [\psi \vee \chi]_{\mathfrak{X}}$. The smallest set $W$ such that $U \subseteq W$ and $V \subseteq W$ is $U \cup V$. The definition for $\to$ is tricky: $\varphi \to \psi$ expresses the weakest proposition that, combined with $\varphi$, entails $\psi$. That $\varphi \to \psi$ combined with $\varphi$ entails $\psi$ is clear from $(\varphi \to \psi) \wedge \varphi \vdash \psi$. So $[\varphi \to \psi]_{\mathfrak{X}}$ should be the greatest open set such that $[\varphi \to \psi]_{\mathfrak{X}} \cap [\varphi]_{\mathfrak{X}} \subset [\psi]_{\mathfrak{X}}$, leading to our definition.

## Problems

**Problem 45.1.** Show that according to Definition 45.2, $\mathfrak{M}, w \Vdash \neg\varphi$ iff $\mathfrak{M}, w \Vdash \varphi \to \bot$.

**Problem 45.2.** Prove Proposition 45.3.

# Chapter 46

# Soundness and Completeness

This chapter collects soundness and completeness results for propositional intuitionistic logic. It needs an introduction. The completeness proof makes use of facts about provability that should be stated and proved explicitly somehwere.

## 46.1 Soundness of Axiomatic Derivations

The soundness proof relies on the fact that all axioms are intuitionistically valid; this still needs to be proved, e.g., in the Semantics chapter.

**Theorem 46.1** (Soundness)**.** *If $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$.*

*Proof.* We prove that if $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$. The proof is by induction on the number $n$ of formulas in the derivation of $\varphi$ from $\Gamma$. We show that if $\varphi_1$, $\ldots$, $\varphi_n = \varphi$ is a derivation from $\Gamma$, then $\Gamma \vDash \varphi_n$. Note that if $\varphi_1$, $\ldots$, $\varphi_n$ is a derivation, so is $\varphi_1, \ldots, \varphi_k$ for any $k < n$.

There are no derivations of length 0, so for $n = 0$ the claim holds vacuously. So the claim holds for all derivations of length $< n$. We distinguish cases according to the justification of $\varphi_n$.

1. $\varphi_n$ is an axiom. All axioms are valid, so $\Gamma \vDash \varphi_n$ for any $\Gamma$.

2. $\varphi_n \in \Gamma$. Then for any $\mathfrak{M}$ and $w$, if $\mathfrak{M}, w \Vdash \Gamma$, obviously $\mathfrak{M} \Vdash \Gamma \varphi_n[w]$, i.e., $\Gamma \vDash \varphi$.

3. $\varphi_n$ follows by MP from $\varphi_i$ and $\varphi_j \equiv \varphi_i \to \varphi_n$. $\varphi_1, \ldots, \varphi_i$ and $\varphi_1, \ldots, \varphi_j$ are derivations from $\Gamma$, so by inductive hypothesis, $\Gamma \vDash \varphi_i$ and $\Gamma \vDash \varphi_i \to \varphi_n$.

Suppose $\mathfrak{M}, w \Vdash \Gamma$. Since $\mathfrak{M}, w \Vdash \Gamma$ and $\Gamma \vDash \varphi_i \to \varphi_n$, $\mathfrak{M}, w \Vdash \varphi_i \to \varphi_n$. By definition, this means that for all $w'$ such that $Rww'$, if $\mathfrak{M}, w' \Vdash \varphi_i$ then $\mathfrak{M}, w' \Vdash \varphi_n$. Since $R$ is reflexive, $w$ is among the $w'$ such that $Rww'$, i.e., we have that if $\mathfrak{M}, w \Vdash \varphi_i$ then $\mathfrak{M}, w \Vdash \varphi_n$. Since $\Gamma \vDash \varphi_i$, $\mathfrak{M}, w \Vdash \varphi_i$. So, $\mathfrak{M}, w \Vdash \varphi_n$, as we wanted to show.

□

## 46.2 Soundness of Natural Deduction

**Theorem 46.2** (Soundness)**.** *If $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$.*

*Proof.* We prove that if $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$. The proof is by induction on the derivation of $\varphi$ from $\Gamma$.

1. If the derivation consists of just the assumption $\varphi$, we have $\varphi \vdash \varphi$, and want to show that $\varphi \vDash \varphi$. Consider any model $\mathfrak{M}$ such that $\mathfrak{M} \Vdash \varphi$. Then trivially $\mathfrak{M} \Vdash \varphi$.

2. The derivation ends in $\wedge$Intro: Exercise.

3. The derivation ends in $\wedge$Elim: Exercise.

4. The derivation ends in $\vee$Intro: Suppose the premise is $\psi$, and the undischarged assumptions of the derivation ending in $\psi$ are $\Gamma$. Then we have $\Gamma \vdash \psi$ and by inductive hypothesis, $\Gamma \vDash B$. We have to show that $\Gamma \vDash \psi \vee \chi$. Suppose $\mathfrak{M} \Vdash \Gamma$. Since $\Gamma \vDash \psi$, $\mathfrak{M} \Vdash \psi$. But then also $\mathfrak{M} \Vdash \psi \vee \chi$. Similarly, if the premise is $\chi$, we have that $\Gamma \vDash \chi$.

5. The derivation ends in $\vee$Elim: The derivations ending in the premises are of $\psi \vee \chi$ from undischarged assumptions $\Gamma$, of $\theta$ from undischarged assumptions $\Delta_1 \cup \{\psi\}$, and of $\theta$ from undischarged assumptions $\Delta_2 \cup \{\chi\}$. So we have $\Gamma \vdash \psi \vee \chi$, $\Delta_1 \cup \{\psi\} \vdash \theta$, and $\Delta_2 \cup \{\chi\} \vdash \theta$. By induction hypothesis, $\Gamma \vDash \psi \vee \chi$, $\Delta_1 \cup \{\psi\} \vDash \theta$, and $\Delta_2 \cup \{\chi\} \vDash \theta$. We have to prove that $\Gamma \cup \Delta_1 \cup \Delta_2 \vDash \theta$.

   Suppose $\mathfrak{M} \Vdash \Gamma \cup \Delta_1 \cup \Delta_2$. Then $\mathfrak{M} \Vdash \Gamma$ and since $\Gamma \vDash \psi \vee \chi$, $\mathfrak{M} \Vdash \psi \vee \chi$. By definition of $\mathfrak{M} \Vdash$, either $\mathfrak{M} \Vdash \psi$ or $\mathfrak{M} \Vdash \chi$. So we distinguish cases: (a) $\mathfrak{M} \Vdash \psi$. Then $\mathfrak{M} \Vdash \Delta_1 \cup \{\psi\}$. Since $\Delta_1 \cup \psi \vDash \theta$, we have $\mathfrak{M} \Vdash \theta$. (b) $\mathfrak{M} \Vdash \chi$. Then $\mathfrak{M} \Vdash \Delta_2 \cup \{\chi\}$. Since $\Delta_2 \cup \chi \vDash \theta$, we have $\mathfrak{M} \Vdash \theta$. So in either case, $\mathfrak{M} \Vdash \theta$, as we wanted to show.

6. The derivation ends with $\to$Intro concluding $\psi \to \chi$. Then the premise is $\chi$, and the derivation ending in the premise has undischarged assumptions $\Gamma \cup \{\psi\}$. So we have that $\Gamma \cup \{\psi\} \vdash \chi$, and by induction hypothesis that $\Gamma \cup \{\psi\} \vDash \chi$. We have to show that $\Gamma \vDash \psi \to \chi$.

Suppose $\mathfrak{M}, w \Vdash \Gamma$. We want to show that that for all $w'$ such that $Rww'$, if $\mathfrak{M}, w' \Vdash \psi$, then $\mathfrak{M}, w' \Vdash \chi$. So assume that $Rww'$ and $\mathfrak{M}, w' \Vdash \psi$. By Proposition 45.3, $\mathfrak{M}, w' \Vdash \Gamma$. Since $\Gamma \cup \{\psi\} \vDash \chi$, $\mathfrak{M}, w' \Vdash \chi$, which is what we wanted to show.

7. The derivation ends in $\to$Elim and conclusion $\chi$. The premises are $\psi \to \chi$ amd $\psi$, with derivations from undischarged assumptions $\Gamma$, $\Delta$. So we have $\Gamma \vdash \psi \to \chi$ and $\Delta \vdash \psi$. By inductive hypothesis, $\Gamma \vDash \psi \to \chi$ and $\Delta \vDash \psi$. We have to show that $\Gamma \cup \Delta \vDash \chi$.

   Suppose $\mathfrak{M}, w \Vdash \Gamma \cup \Delta$. Since $\mathfrak{M}, w \Vdash \Gamma$ and $\Gamma \vDash \psi \to \chi$, $\mathfrak{M}, w \Vdash \psi \to \chi$. By definition, this means that for all $w'$ such that $Rww'$, if $\mathfrak{M}, w' \Vdash \psi$ then $\mathfrak{M}, w' \Vdash \chi$. Since $R$ is reflexive, $w$ is among the $w'$ such that $Rww'$, i.e., we have that if $\mathfrak{M}, w \Vdash \psi$ then $\mathfrak{M}, w \Vdash \chi$. Since $\mathfrak{M}, w \Vdash \Delta$ and $\Delta \vDash \psi$, $\mathfrak{M}, w \Vdash \psi$. So, $\mathfrak{M}, w \Vdash \chi$, as we wanted to show.

8. The derivation ends in $\bot_I$, concluding $\varphi$. The premise is $\bot$ and the undischarged assumptions of the derivation of the premise are $\Gamma$. Then $\Gamma \vdash \bot$. By inductive hypothesis, $\Gamma \vDash \bot$. We have to show $\Gamma \vDash \varphi$.

   We proceed indirectly. If $\Gamma \nvDash \varphi$ there is a model $\mathfrak{M}$ and world $w$ such that $\mathfrak{M}, w \Vdash \Gamma$ and $\mathfrak{M}, w \nVdash \varphi$. Since $\Gamma \vDash \bot$, $\mathfrak{M}, w \Vdash \bot$. But that's impossible, since by definition, $\mathfrak{M}, w \nVdash \bot$. So $\Gamma \vDash \varphi$.

9. The derivation endss in $\neg$Intro: Exercise.

10. The derivation ends in $\neg$Elim: Exercise.

□

## 46.3 Lindenbaum's Lemma

**Definition 46.3.** A set of formulas $\Gamma$ is *prime* iff

1. $\Gamma$ is consistent.
2. If $\Gamma \vdash \varphi$ then $\varphi \in \Gamma$, and
3. If $\varphi \vee \psi \in \Gamma$ then $\varphi \in \Gamma$ or $\psi \in \Gamma$.

**Lemma 46.4** (Lindenbaum's Lemma)**.** *If $\Gamma \nvdash \varphi$, there is a $\Gamma^* \supseteq \Gamma$ such that $\Gamma^*$ is prime and $\Gamma^* \nvdash \varphi$.*

*Proof.* Let $\psi_1 \vee \chi_1, \psi_2 \vee \chi_2, \ldots$, be an enumeration of all formulas of the form $\psi \vee \chi$. We'll define an increasing sequence of sets of formulas $\Gamma_n$, where each $\Gamma_{n+1}$ is defined as $\Gamma_n$ together with one new formula. $\Gamma^*$ will be the union of all $\Gamma_n$. The new formulas are selected so as to ensure that $\Gamma^*$ is prime and still $\Gamma^* \nvdash \varphi$. This means that at each step we should find the first disjunction $\psi_i \vee \chi_i$ such that:

1. $\Gamma_n \vdash \psi_i \vee \chi_i$

2. $\psi_i \notin \Gamma_n$ and $\chi_i \notin \Gamma_n$

We add to $\Gamma_n$ either $\psi_i$ if $\Gamma_n \cup \{\psi_i\} \nvdash \varphi$, or $\chi_i$ otherwise. We'll have to show that this works. For now, let's define $i(n)$ as the least $i$ such that (1) and (2) hold.

Define $\Gamma_0 = \Gamma$ and

$$\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{\psi_{i(n)}\} & \text{if } \Gamma_n \cup \{\psi_{i(n)}\} \nvdash \varphi \\ \Gamma_n \cup \{\chi_{i(n)}\} & \text{otherwise} \end{cases}$$

If $i(n)$ is undefined, i.e., whenever $\Gamma \vdash \psi \vee \chi$, either $\psi \in \gamma_n$ or $\chi \in \Gamma_n$, we let $\Gamma_{n+1} = \Gamma_n$. Now let $\Gamma^* = \bigcup_{n=0}^{\infty} \Gamma_n$

First we show that for all $n$, $\Gamma_n \nvdash \varphi$. We proceed by induction on $n$. For $n = 0$ the claim holds by the hypothesis of the theorem, i.e., $\Gamma \nvdash \varphi$. If $n > 0$, we have to show that if $\Gamma_n \nvdash \varphi$ then $\Gamma_{n+1} \nvdash \varphi$. If $i(n)$ is undefined, $\Gamma_{n+1} = \Gamma_n$ and there is nothing to prove. So suppose $i(n)$ is defined. For simplicity, let $i = i(n)$.

We'll prove the contrapositive of the claim. Suppose $\Gamma_{n+1} \vdash \varphi$. By construction, $\Gamma_{n+1} = \Gamma_n \cup \{\psi_i\}$ if $\Gamma_n \cup \{\psi_i\} \nvdash \varphi$, or else $\Gamma_{n+1} = \Gamma_n \cup \{\chi_i\}$. It clearly can't be the first, since then $\Gamma_{n+1} \nvdash \varphi$. Hence, $\Gamma_n \cup \{\psi_i\} \vdash \varphi$ and $\Gamma_{n+1} = \Gamma_n \cup \{\chi_i\}$. By definition of $i(n)$, we have that $\Gamma_n \vdash \psi_i \vee \chi_i$. We have $\Gamma_n \cup \{\psi_i\} \vdash \varphi$. We also have $\Gamma_{n+1} = \Gamma_n \cup \{\chi_i\} \vdash \varphi$. Hence, $\Gamma_n \vdash \varphi$, which is what we wanted to show.

If $\Gamma^* \vdash \varphi$, there would be some finite subset $\Gamma' \subseteq \Gamma^*$ such that $\Gamma' \vdash \varphi$. Each $\theta \in \Gamma'$ must be in $\Gamma_i$ for some $i$. Let $n$ be the largest of these. Since $\Gamma_i \subseteq \Gamma_n$ if $i \leq n$, $\Gamma' \subseteq \Gamma_n$. But then $\Gamma_n \vdash \varphi$, contrary to our proof above that $\Gamma_n \nvdash \varphi$.

Lastly, we show that $\Gamma^*$ is prime, i.e., satisfies conditions (1), (2), and (3) of Definition 46.3.

First, $\Gamma^* \nvdash \varphi$, so $\Gamma^*$ is consistent, so (1) holds.

We now show that if $\Gamma^* \vdash \psi \vee \chi$, then either $\psi \in \Gamma^*$ or $\chi \in \Gamma^*$. This proves (3), since if $\psi \in \Gamma^*$ then also $\Gamma^* \vdash \psi$, and similarly for $\chi$. So assume $\Gamma^* \vdash \psi \vee \chi$ but $\psi \notin \Gamma^*$ and $\chi \notin \Gamma^*$. Since $\Gamma^* \vdash \psi \vee \chi$, $\Gamma_n \vdash \psi \vee \chi$ for some $n$. $\psi \vee \chi$ appears on the enumeration of all disjunctions, say as $\psi_j \vee \chi_j$. $\psi_j \vee \chi_j$ satisfies the properties in the definition of $i(n)$, namely we have $\Gamma_n \vdash \psi_j \vee \chi_j$, while $\psi_j \notin \Gamma_n$ and $\chi_j \notin \Gamma_n$. At each stage, at least one fewer disjunction $\psi_i \vee \chi_i$ satisfies the conditions (since at each stage we add either $\psi_i$ or $\chi_i$), so at some stage $m$ we will have $j = i(\Gamma_m)$. But then either $\psi \in \Gamma_{m+1}$ or $\chi \in \Gamma_{m+1}$, contrary to the assumption that $\psi \notin \Gamma^*$ and $\chi \notin \Gamma^*$.

Now suppose $\Gamma^* \vdash \varphi$. Then $\Gamma^* \vdash \varphi \vee \varphi$. But we've just proved that if $\Gamma^* \vdash \varphi \vee \varphi$ then $\varphi \in \Gamma^*$. Hence, $\Gamma^*$ satisfies (1) of Definition 46.3. □

## 46.4 The Canonical Model

The worls in our model will be finite sequences $\sigma$ of natural numbers, i.e., $\sigma \in \mathbb{N}^*$. Note that $\mathbb{N}^*$ is inductively defined by:

1. $\Lambda \in \mathbb{N}^*$.
2. If $\sigma \in \mathbb{N}^*$ and $n \in \Sigma$, then $\sigma.n \in \mathbb{N}^*$ (where $\sigma.n$ is $\sigma \frown \langle n \rangle$).
3. Nothing else is in $\mathbb{N}^*$.

So we can use $\mathbb{N}^*$ to give inductive definitions.

Let $\langle \psi_1, \chi_1 \rangle, \langle \psi_2, \chi_s \rangle, \ldots$, be an enumeration of all pairs of formulas. Given a set of formulas $\Delta$, define $\Delta(\sigma)$ by induction as follows:

1. $\Delta(\Lambda) = \Delta$
2. $\Delta(\sigma.n) =$
$$\begin{cases} (\Delta(\sigma) \cup \{\psi_n\})^* & \text{if } \Delta(\sigma) \cup \{\psi_n\} \nvdash \chi_n \\ \Delta(\sigma) & \text{otherwise} \end{cases}$$

Here by $(\Delta(\sigma) \cup \{\psi_n\})^*$ we mean the prime set of formulas which exists by Lemma 46.4 applied to the set $\Delta(\sigma) \cup \{\psi_n\}$. Note that by this definition, if $\Delta(\sigma) \cup \{\psi_n\} \nvdash \chi_n$, then $\Delta(\sigma.n) \vdash \psi_n$ and $\Delta(\sigma.n) \nvdash \chi_n$. Note also that $\Delta(\sigma) \subseteq \Delta(\sigma.n)$ for any $n$. If $\Delta$ is prime, then $\Delta(\sigma)$ is prime for all $\sigma$.

**Definition 46.5.** Suppose $\Delta$ is prime. Then the *canonical model* for $\Delta$ is defined by:

1. $W = \mathbb{N}^*$, the set of finite sequences of natural numbers.
2. $R$ is the partial order according to which $R\sigma\sigma'$ iff $\sigma$ is an initial segment of $\sigma'$ (i.e., $\sigma' = \sigma \frown \sigma''$ for some sequence $\sigma''$).
3. $V(p) = \{\sigma : p \in \Delta(\sigma)\}$.

It is easy to verify that $R$ is indeed a partial order. Also, the monotonicity condition on $V$ is satisfied. Since $\Delta(\sigma) \subseteq \Delta(\sigma.n)$ we get $\Delta(\sigma) \subseteq \Delta(\sigma')$ whenever $R\sigma\sigma'$ by induction on $\sigma$.

## 46.5 The Truth Lemma

**Lemma 46.6.** *If $\Delta$ is prime, then $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$ iff $\Delta(\sigma) \vdash \varphi$.*

*Proof.* By induction on $\varphi$.

1. $\varphi \equiv \bot$: Since $\Delta(\sigma)$ is prime, it is consistent, so $\Delta(\sigma) \nvdash \varphi$. By definition, $\mathfrak{M}(\Delta), \sigma \nVdash \varphi$.

2. $\varphi \equiv p$: By definition of $\Vdash$, $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$ iff $\sigma \in V(p)$, i.e., $\Delta(\sigma) \vdash \varphi$.

3. $\varphi \equiv \neg\psi$: exercise.

4. $\varphi \equiv \psi \wedge \chi$: $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$ iff $\mathfrak{M}(\Delta), \sigma \Vdash \psi$ and $\mathfrak{M}(\Delta), \sigma \Vdash \chi$. By induction hypothesis, $\mathfrak{M}(\Delta), \sigma \Vdash \psi$ iff $\Delta(\sigma) \vdash \psi$, and similarly for $\chi$. But $\Delta(\sigma) \vdash \psi$ and $\Delta(\sigma) \vdash \chi$ iff $\Delta(\sigma) \vdash \varphi$.

5. $\varphi \equiv \psi \vee \chi$: $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$ iff $\mathfrak{M}(\Delta), \sigma \Vdash \psi$ or $\mathfrak{M}(\Delta), \sigma \Vdash \chi$. By induction hypothesis, this holds iff $\Delta(\sigma) \vdash \psi$ of $\Delta(\sigma) \vdash \chi$. We have to show that this in turn holds iff $\Delta(\sigma) \vdash \varphi$. The left-to-right direction is clear. The right-to-left direction follows since $\Delta(\sigma)$ is prime.

6. $\varphi \equiv \psi \to \chi$: First the contrapositive of the left-to-right direction: Assume $\Delta(\sigma) \nvdash \psi \to \chi$. Then also $\Gamma * (\sigma) \cup \{\psi\} \nvdash \chi$. Since $\langle \psi, \chi \rangle$ is $\langle \psi_n, \chi_n \rangle$ for some $n$, we have $\Delta(\sigma.n) = (\Delta(\sigma) \cup \{\psi\})^*$, and $\Delta(\sigma.n) \vdash \psi$ but $\nvdash \chi$. By inductive hypothesis, $\mathfrak{M}(\Delta), \sigma.n \Vdash \psi$ and $\mathfrak{M}(\Delta), \sigma.n \nVdash \chi$. Since $R\sigma(\sigma.n)$, this means that $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$.

   Now assume $\Delta(\sigma) \vdash \psi \to \chi$, and let $R\sigma\sigma'$. Since $\Delta(\sigma) \subseteq \Delta(\sigma')$, we have: if $\Delta(\sigma') \vdash \psi$, then $\Delta(\sigma') \vdash \chi$. In other words, for every $\sigma'$ such that $R\sigma\sigma'$, either $\Delta(\sigma') \nvdash \psi$ or $\Delta(\sigma') \vdash \chi$. By induction hypothesis, this means that whenever $R\sigma\sigma'$, either $\mathfrak{M}(\Delta), \sigma' \nVdash \psi$ or $\mathfrak{M}(\Delta), \sigma' \Vdash \chi$, i.e., $\mathfrak{M}(\Delta), \sigma \Vdash \varphi$.

□

## 46.6 The Completeness Theorem

**Theorem 46.7.** *If $\Gamma \vDash \varphi$ then $\Gamma \vdash \varphi$.*

*Proof.* We prove the contrapositive: Suppose $\Gamma \nvdash \varphi$. Then by Lemma 46.4, there is a prime set $\Gamma^* \supseteq \Gamma$ such that $\Gamma^* \nvdash \varphi$. Consider the canonical model $\mathfrak{M}(\Gamma^*)$ for $\Gamma^*$ as defined in Definition 46.5. For any $\psi \in \Gamma$, $\Gamma^* \vdash \psi$. Note that $\Gamma * (\Lambda) = \Gamma*$. By the Truth Lemma (Lemma 46.6), we have $\mathfrak{M}(\Gamma^*), \Lambda \Vdash \psi$ for all $\psi \in \Gamma$ and $\mathfrak{M}(\Gamma^*), \Lambda \nVdash \varphi$. This shows that $\Gamma \nvDash \varphi$. □

## Problems

**Problem 46.1.** Complete the proof of Theorem 46.2. For the cases for ¬Intro and ¬Elim, use the definition of $\mathfrak{M}, w \Vdash \neg\varphi$ in Definition 45.2, i.e., don't treat $\neg\varphi$ as defined by $\varphi \to \bot$.

# Chapter 47

# Propositions as Types

This is a *very experimental* draft of a chapter on the Curry-Howard correspondence. It needs more explanation and motivation, and there are probably errors and omissions. The proof of normalization should be reviewed and expanded. There are no examples for the product type. Permuation and simplification conversions are not covered. It will make a lot more sense once there is also material on the (typed) lambda calculus which is basically presupposed here. Use with extreme caution.

## 47.1 Introduction

Historically the lambda calculus and intuitionistic logic were developed separately. Haskell Curry and William Howard independently discovered a close similarity: types in a typed lambda calculus correspond to formulas in intuitionistic logic in such a way that a derivation of a formula corresponds directly to a typed lambda term with that formula as its type. Moreover, beta reduction in the typed lambda calculus corresponds to certain transformations of derivations.

For instance, a derivation of $\varphi \to \psi$ corresponds to a term $\lambda x^{\varphi}. N^{\psi}$, which has the function type $\varphi \to \psi$. The inference rules of natural deduction correspond to typing rules in the typed lambda calculus, e.g.,

$$\begin{array}{c} [\varphi]^x \\ \vdots \\ \psi \\ \hline \varphi \to \psi \end{array} x \to\text{Intro} \qquad \text{corresponds to} \qquad \frac{x : \varphi \;\Rightarrow\; N : \psi}{\Rightarrow\; \lambda x^{\varphi}. N^{\psi} : \varphi \to \psi}\,\lambda$$

where the rule on the right means that if $x$ is of type $\varphi$ and $N$ is of type $\psi$, then $\lambda x^{\varphi}. N$ is of type $\varphi \to \psi$.

The $\to$Elim rule corresponds to the typing rule for composition terms, i.e.,

$$\dfrac{\dfrac{\varphi \to \psi \qquad \varphi}{\psi}\ {\to}\text{Elim}}{\dfrac{\Rightarrow\ P : \varphi \to \psi \qquad \Rightarrow\ Q : \varphi}{\Rightarrow\ P^{\varphi\to\psi} Q^{\varphi} : \psi}\ \text{app}} \quad \text{corresponds to}$$

If a $\to$Intro rule is followed immediately by a $\to$Elim rule, the derivation can be simplified:

$$\dfrac{\dfrac{\begin{array}{c}[\varphi]^x \\ \vdots \\ \psi\end{array}}{\varphi \to \psi}\ {\to}\text{Intro}^{\,x} \qquad \begin{array}{c}\vdots \\ \varphi\end{array}}{\psi}\ {\to}\text{Elim} \qquad \rhd_1 \qquad \begin{array}{c}\vdots \\ \varphi \\ \vdots \\ \psi\end{array}$$

which corresponds to the beta reduction of lambda terms

$$(\lambda x^{\varphi}.\, P^{\psi})Q \quad \rhd_1 \quad P[Q/x].$$

Similar correspondences hold between the rules for $\wedge$ and "product" types, and between the rules for $\vee$ and "sum" types.

This correspondence between terms in the simply typed lambda calculus and natural deduction derivations is called the "Curry-Howard", or "propositions as types" correspondence. In addition to formulas (propositions) corresponding to types, and proofs to terms, we can summarize the correspondences as follows:

| logic | program |
|---|---|
| proposition | type |
| proof | term |
| assumption | variable |
| discharged assumption | bind variable |
| not discharged assumption | free variable |
| implication | function type |
| conjunction | product type |
| disjunction | sum type |
| absurdity | bottom type |

The Curry-Howard correspondence is one of the cornerstones of automated proof assistants and type checkers for programs, since checking a proof witnessing a proposition (as we did above) amounts to checking if a program (term) has the declared type.

## 47.2 Sequent Natural Deduction

Let us write $\Gamma \Rightarrow \varphi$ if there is a natural deduction derivation with $\Gamma$ as undischarged assumptions and $\varphi$ as conclusion; or $\Rightarrow \varphi$ if $\Gamma$ is empty.

We write $\Gamma, \varphi_1, \ldots, \varphi_n$ for $\Gamma \cup \{\varphi_1, \ldots, \varphi_n\}$, and $\Gamma, \Delta$ for $\Gamma \cup \Delta$.

Observe that when we have $\Gamma \Rightarrow \varphi \wedge \varphi$, meaning we have a derivation with $\Gamma$ as undischarged assumptions and $\varphi \wedge \varphi$ as end-formula, then by applying $\wedge$Elim at the bottom, we can get a derivation with the same undischarged assumptions and $\varphi$ as conclusion. In other words, if $\Gamma \Rightarrow \varphi \wedge \psi$, then $\Gamma \Rightarrow \varphi$.

$$\frac{\Gamma \Rightarrow \varphi \wedge \psi}{\Gamma \Rightarrow \varphi} \wedge\text{Elim} \qquad \frac{\Gamma \Rightarrow \varphi \wedge \psi}{\Gamma \Rightarrow \psi} \wedge\text{Elim}$$

The label $\wedge$Elim hints at the relation with the rule of the same name in natural deduction.

Likewise, suppose we have $\Gamma, \varphi \Rightarrow \psi$, meaning we have a derivation with undischarged assumptions $\Gamma, \varphi$ and end-formula $\psi$. If we apply the $\rightarrow$Intro rule, we have a derivation with $\Gamma$ as undischarged assumptions and $\varphi \rightarrow \psi$ as the end-formula, i.e., $\Gamma \Rightarrow \varphi \rightarrow \psi$. Note how this has made the discharge of assumptions more explicit.

$$\frac{\Gamma, \varphi \Rightarrow \psi}{\Gamma \Rightarrow \varphi \rightarrow \psi} \rightarrow\text{Intro}$$

We can draw conclusions from other rules in the same fashion, which is spelled out as follows:

$$\frac{\Gamma \Rightarrow \varphi \qquad \Delta \Rightarrow \psi}{\Gamma, \Delta \Rightarrow \varphi \wedge \psi} \wedge\text{Intro}$$

$$\frac{\Gamma \Rightarrow \varphi \wedge \psi}{\Gamma \Rightarrow \varphi} \wedge\text{Elim}_1 \qquad \frac{\Gamma \Rightarrow \varphi \wedge \psi}{\Gamma \Rightarrow \psi} \wedge\text{Elim}_2$$

$$\frac{\Gamma \Rightarrow \varphi}{\Gamma \Rightarrow \varphi \vee \psi} \vee\text{Intro}_1 \qquad \frac{\Gamma \Rightarrow \psi}{\Gamma \Rightarrow \varphi \vee \psi} \vee\text{Intro}_2$$

$$\frac{\Gamma \Rightarrow \varphi \vee \psi \qquad \Delta, \varphi \Rightarrow \chi \qquad \Delta', \psi \Rightarrow \chi}{\Gamma, \Delta, \Delta' \Rightarrow \chi} \vee\text{Elim}$$

$$\frac{\Gamma, \varphi \Rightarrow \psi}{\Gamma \Rightarrow \varphi \rightarrow \psi} \rightarrow\text{Intro} \qquad \frac{\Delta \Rightarrow \varphi \rightarrow \psi \qquad \Gamma \Rightarrow \varphi}{\Gamma, \Delta \Rightarrow \psi} \rightarrow\text{Elim}$$

$$\frac{\Gamma \Rightarrow \bot}{\Gamma \Rightarrow \varphi} \bot_I$$

Any assumption by itself is a derivation of $\varphi$ from $\varphi$, i.e., we always have $\varphi \Rightarrow \varphi$.

$$\frac{}{\varphi \Rightarrow \varphi}$$

Together, these rules can be taken as a calculus about what natural deduction derivations exist. They can also be taken as a notational variant of natural deduction, in which each step records not only the formula derived but also the undischarged assumptions from which it was derived.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\varphi \Rightarrow \varphi}}{\varphi \Rightarrow \varphi \vee (\varphi \to \bot)} \quad \psi \Rightarrow \psi}}{\varphi, \psi \to \Rightarrow \bot}}{(\psi \Rightarrow \varphi \to \bot}}{(\psi \Rightarrow \varphi \vee (\varphi \to \bot)} \quad (\psi \Rightarrow \psi}}{(\psi \Rightarrow \bot}}{\Rightarrow \psi \to \bot}$$

where $\psi$ is short for $(\varphi \vee (\varphi \to \bot)) \to \bot$.

## 47.3 Proof Terms

We give the definition of proof terms, and then establish its relation with natural deduction derivations.

**Definition 47.1** (Proof terms)**.** Proof terms are inductively generated by the following rules:

1. A single variable $x$ is a proof term.
2. If $P$ and $Q$ are proof terms, then $PQ$ is also a proof term.
3. If $x$ is a variable, $\varphi$ is a formula, and $N$ is a proof term, then $\lambda x^{\varphi}. N$ is also a proof term.
4. If $P$ and $Q$ are proof terms, then $\langle P, Q \rangle$ is a proof term.
5. If $M$ is a proof term, then $\mathrm{p}_i(M)$ is also a proof term, where $i$ is 1 or 2.
6. If $M$ is a proof term, and $\varphi$ is a formula, then $\mathrm{in}_i^{\varphi}(M)$ is a proof term, where $i$ is 1 or 2.
7. If $M, N_1, N_2$ is proof terms, and $x_1, x_2$ are variables, then $\mathrm{case}(M, x_1.N_1, x_2.N_2)$ is a proof term.
8. If $M$ is a proof term and $\varphi$ is a formula, then $\mathrm{contr}_{\varphi}(M)$ is proof term.

Each of the above rules corresponds to an inference rule in natural deduction. Thus we can inductively assign proof terms to the formulas in a derivation. To make this assignment unique, we must distinguish between the two versions of $\wedge$Elim and of $\vee$Intro. For instance, the proof terms assigned to the conclusion of $\vee$Intro must carry the information whether $\varphi \vee \psi$ is inferred

from $\varphi$ or from $\psi$. Suppose $M$ is the term assigned to $\varphi$from which $\varphi \vee \psi$ is inferred. Then the proof term assigned to $\varphi \vee \psi$ is $\text{in}_1^{\varphi}(M)$. If we instead infer $\psi \vee \varphi$ then the proof term assigned is $\text{in}_2^{\varphi}(M)$.

The term $\lambda x^{\varphi}.\, N$ is assigned to the conclusion of $\rightarrow$Intro. The $\varphi$ represents the assumption being discharged; only have we included it can we infer the formula of $\lambda x^{\varphi}.\, N$ based on the formula of $N$.

**Definition 47.2** (Typing context)**.** A *typing context* is a mapping from variables to formulas. We will call it simply the "context" if there is no confusion. We write a context $\Gamma$ as a set of pairs $\langle x, \varphi \rangle$.

A pair $\Gamma \Rightarrow M$ where $M$ is a proof term represents a derivation of a formula with context $\Gamma$.

**Definition 47.3** (Typing pair)**.** A *typing pair* is a pair $\langle \Gamma, M \rangle$, where $\Gamma$ is a typing context and $M$ is a proof term.

Since in general terms only make sense with specific contexts, we will speak simply of "terms" from now on instead of "typing pair"; and it will be apparent when we are talking about the literal term $M$.

## 47.4 Converting Derivations to Proof Terms

We will describe the process of converting natural deduction derivations to pairs. We will write a proof term to the left of each formula in the derivation, resulting in expressions of the form $M : \varphi$. We'll then say that, *$M$ witnesses $\varphi$*. Let's call such an expression a *judgment*.

First let us assign to each assumption a variable, with the following constraints:

1. Assumptions discharged in the same step (that is, with the same number on the square bracket) must be assigned the same variable.

2. For assumptions not discharged, assumptions of different formulas should be assigned different variables.

Such an assignment translates all assumptions of the form

$$\varphi \qquad \text{into} \qquad x : \varphi.$$

With assumptions all associated with variables (which are terms), we can now inductively translate the rest of the deduction tree. The modified natural deduction rules taking into account context and proof terms are given below. Given the proof terms for the premise(s), we obtain the corresponding proof term for conclusion.

$$\frac{M_1 : \varphi_1 \qquad M_2 : \varphi_2}{\langle M_1, M_2 \rangle : \varphi_1 \wedge \varphi_2} \wedge\text{Intro}$$

$$\frac{M : \varphi_1 \wedge \varphi_2}{\mathrm{p}_i(M) : \varphi_1} \wedge\text{Elim}_1 \qquad \frac{M : \varphi_1 \wedge \varphi_2}{\mathrm{p}_i(M) : \varphi_2} \wedge\text{Elim}_2$$

In $\wedge$Intro we assume we have $\varphi_1$ witnessed by term $M_1$ and $\varphi_2$ witnessed by term $M_2$. We pack up the two terms into a pair $\langle M_1, M_2 \rangle$ which witnesses $\varphi_1 \wedge \varphi_2$.

In $\wedge\text{Elim}_i$ we assume that $M$ witnesses $\varphi_1 \wedge \varphi_2$. The term witnessing $\varphi_i$ is $\mathrm{p}_i(M)$. Note that $M$ is not necessary of the form $\langle M_1, M_2 \rangle$, so we cannot simply assign $M_1$ to the conclusion $\varphi_i$.

Note how this coincides with the BHK interpretation. What the BHK interpretation does not specify is how the function used as proof for $\varphi \to \psi$ is supposed to be obtained. If we think of proof terms as proofs or functions of proofs, we can be more explicit.

$$\frac{\begin{array}{c}[x : \varphi] \\ \vdots \\ N : \psi\end{array}}{\lambda x^{\varphi}.\, N : \varphi \to \psi} \to\text{Intro} \qquad \frac{P : \varphi \to \psi \qquad Q : \varphi}{PQ : \psi} \to\text{Elim}$$

The $\lambda$ notation should be understood as the same as in the lambda calculus, and $PQ$ means applying $P$ to $Q$.

$$\frac{M_1 : \varphi_1}{\mathrm{in}_1^{\varphi_1}(M_1) : \varphi_1 \vee \varphi_2} \vee\text{Intro}_1 \qquad \frac{M_2 : \varphi_2}{\mathrm{in}_2^{\varphi_2}(M_2) : \varphi_1 \vee \varphi_2} \vee\text{Intro}_2$$

$$\frac{M : A_1 \vee \varphi_2 \qquad \begin{array}{c}[x_1 : \varphi_1] \\ \vdots \\ N_1 : \chi\end{array} \qquad \begin{array}{c}[x_2 : \varphi_2] \\ \vdots \\ N_2 : \chi\end{array}}{\mathrm{case}(M, x_1.N_1, x_2.N_2) : \chi} \vee\text{Elim}$$

The proof term $\mathrm{in}_1^{\varphi_1}(M_1)$ is a term witnessing $\varphi_1 \vee \varphi_2$, where $M_1$ witnesses $\varphi_1$.

The term $\mathrm{case}(M, x_1.N_1, x_2.N_2)$ mimics the case clause in programming languages: we already have the derivation of $\varphi \vee \psi$, a derivation of $\chi$ assuming $\varphi$, and a derivation of $\chi$ assuming $\psi$. The *case* operator thus select the appropriate proof depending on $M$; either way it's a proof of $\chi$.

$$\frac{N : \bot}{\mathrm{contr}_{\varphi}(N) : \varphi} \bot_I$$

$\mathrm{contr}_\varphi(N)$ is a term witnessing $\varphi$, whenever $N$ is a term witnessing $\bot$.

Now we have a natural deduction derivation with all formulas associated with a term. At each step, the relevant typing context $\Gamma$ is given by the list of assumptions remaining undischarged at that step. Note that $\Gamma$ is well defined: since we have forbidden assumptions of different undischarged assumptions to be assigned the same variable, there won't be any disagreement about the formulas mapped to which a variable is mapped.

We now give some examples of such translations:

Consider the derivation of $\neg\neg(\varphi \vee \neg\varphi)$, i.e., $((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot$. Its translation is:

$$\dfrac{\dfrac{[y : (\varphi \vee (\varphi \to \bot)) \to \bot]^2 \qquad \dfrac{\dfrac{[x : \varphi]^1}{\mathrm{in}_1^{\varphi\to\bot}(x) : \varphi \vee (\varphi \to \bot)}}{y(\mathrm{in}_1^{\varphi\to\bot}(x)) : \bot}\,1}{\lambda x^\varphi.\, y(\mathrm{in}_1^{\varphi\to\bot}(x)) : \varphi \to \bot}}{\dfrac{[y : (\varphi \vee (\varphi \to \bot)) \to \bot]^2 \qquad \mathrm{in}_2^\varphi(\lambda x^\varphi.\, y(\mathrm{in}_1^{\varphi\to\bot}(x))) : \varphi \vee (\varphi \to \bot)}{y(\mathrm{in}_2^\varphi(\lambda x^\varphi.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) : \bot}\,2}$$
$$\lambda y^{(\varphi\vee(\varphi\to\bot))\to\bot}.\, y(\mathrm{in}_2^\varphi(\lambda x^\varphi.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) : ((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot$$

The tree has no assumptions, so the context is empty; we get:

$$\vdash \lambda y^{(\varphi\vee(\varphi\to\bot))\to\bot}.\, y(\mathrm{in}_2^\varphi(\lambda x^\varphi.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) : ((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot$$

If we leave out the last $\to$Intro, the assumptions denoted by $y$ would be in the context and we would get:

$$y : ((\varphi \vee (\varphi \to \bot)) \to \bot) \vdash y(\mathrm{in}_2^\varphi(\lambda x^\varphi.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) : \bot$$

Another example: $\vdash \varphi \to (\varphi \to \bot) \to \bot$

$$\dfrac{\dfrac{\dfrac{[x : \varphi]^2 \qquad [y : \varphi \to \bot]^1}{yx : \bot}}{\lambda y^{\varphi\to\bot}.\, yx : (\varphi \to \bot) \to \bot}\,1}{\lambda x^\varphi.\, \lambda y^{\varphi\to\bot}.\, yx : \varphi \to (\varphi \to \bot) \to \bot}\,2$$

Again all assumptions are discharged and thus the context is empty, the resulting term is

$$\vdash \lambda x^\varphi.\, \lambda y^{\varphi\to\bot}.\, yx : \varphi \to (\varphi \to \bot) \to \bot$$

If we leave out the last two $\to$Intro inferences, the assumptions denoted by both $x$ and $y$ would be in context and we would get

$$x : \varphi, y : \varphi \to \bot \vdash yx : \bot$$

## 47.5 Recovering Derivations from Proof Terms

Now let us consider the other direction: translating terms back to natural deduction trees. We will use still use the double refutation of the excluded middle as example, and let $S$ denote this term, i.e.,

$$\lambda y^{(\varphi\vee(\varphi\to\bot))\to\bot}.\, y(\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) : ((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot$$

For each natural deduction rule, the term in the conclusion is always formed by wrapping some operator around the terms assigned to the premise(s). Rules correspond uniquely to such operators. For example, from the structure of the $S$ we infer that the last rule applied must be $\to$Intro, since it is of the form $\lambda y^{\cdots}.\ \ldots$, and the $\lambda$ operator corresponds to $\to$Intro. In general we can recover the skeleton of the derivation solely by the structure of the term, e.g.,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[x]^1}{\mathrm{in}_1^{\varphi\to\bot}(x) :}\vee\text{Intro}_1 \quad [y :]^2}{y(\mathrm{in}_1^{\varphi\to\bot}(x)) :}\to\text{Elim}}{\lambda x^{\varphi}.\, y(\mathrm{in}_1^{\varphi\to\bot}(x)) :}\,1\to\text{Intro}}{\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y\mathrm{in}_1^{\varphi\to\bot}(x)) :}\vee\text{Intro}_2 \quad [y :]^2}{y(\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) :}\to\text{Elim}}{\lambda y^{(\varphi\vee(\varphi\to\bot))\to\bot}.\, y(\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y(\mathrm{in}_1^{\varphi\to\bot}(x)))) :}\,2\to\text{Intro}$$

Our next step is to recover the formulas these terms witness. We define a function $F(\Gamma, M)$ which denotes the formula witnessed by $M$ in context $\Gamma$, by induction on $M$ as follows:

$$\begin{aligned}
F(\Gamma, x) &= \Gamma(x) \\
F(\Gamma, \langle N_1, N_2\rangle &= F(\Gamma, N_1) \wedge F(\Gamma, N_2) \\
F(\Gamma, \mathrm{p}_i(N) &= \varphi_i \text{ if } F(\Gamma, N) = \varphi_1 \wedge \varphi_2 \\
F(\Gamma, \mathrm{in}_i^{\varphi}(N) &= \begin{cases} F(N) \vee \varphi & \text{if } i = 1 \\ \varphi \vee F(N) & \text{if } i = 2 \end{cases} \\
F(\Gamma, \mathrm{case}(M, x_1.N_1, x_2.N_2)) &= F(\Gamma \cup \{x_i : F(\Gamma, M)\}, N_i) \\
F(\Gamma, \lambda x^{\varphi}.\, N) &= \varphi \to F(\Gamma \cup \{x : \varphi\}, N) \\
F(\Gamma, NM) &= \psi \text{ if } F(\Gamma, N) = \varphi \to \psi
\end{aligned}$$

where $\Gamma(x)$ means the formula mapped to by $x$ in $\Gamma$ and $\Gamma \cup \{x : \varphi\}$ is a context exactly as $\Gamma$ except mapping $x$ to $\varphi$, whether or not $x$ is already in $\Gamma$.

Note there are cases where $F(\Gamma, M)$ is not defined, for example:

1. In the first line, it is possible that $x$ is not in $\Gamma$.

2. In recursive cases, the inner invocation may be undefined, making the outer one undefined too.

3. In the third line, its only defined when $F(\Gamma, M)$ is of the form $\varphi_1 \vee \varphi_2$, and the right hand is independent on $i$.

As we recursively compute $F(\Gamma, M)$, we work our way up the natural deduction derivation. The every step in the computation of $F(\Gamma, M)$ corresponds to a term in the derivation to which the derivation-to-term translation assigns $M$, and the formula computed is the end-formula of the derivation. However, the result may not be defined for some choices of $\Gamma$. We say that such pairs $\langle \Gamma, M \rangle$ are *ill-typed*, and otherwise *well-typed*. However, if the term $M$ results from translating a derivation, and the formulas in $\Gamma$ correspond to the undischarged assumptions of the derivation, the pair $\langle \Gamma, M \rangle$ will be well-typed.

**Proposition 47.4.** *If $D$ is a derivation with undischarged assumptions $\varphi_1, \ldots, \varphi_n$, $M$ is the proof term associated with $D$ and $\Gamma = \{x_1 : \varphi_1, \ldots, x_n : \varphi_n\}$, then the result of recovering derivation from $M$ in context $\Gamma$ is $D$.*

In the other direction, if we first translate a typing pair to natural deduction and then translate it back, we won't get the same pair back since the choice of variables for the undischarged assumptions is underdetermined. For example, consider the pair $\langle \{x : \varphi, y : \varphi \to \psi\}, yx \rangle$. The corresponding derivation is

$$\dfrac{\varphi \to \psi \qquad \varphi}{\psi} \to\text{Elim}$$

By assigning different variables to the undischarged assumptions, say, $u$ to $\varphi \to \psi$ and $v$ to $\varphi$, we would get the term $uv$ rather than $yx$. There is a connection, though: the terms will be the same up to renaming of variables.

Now we have established the correspondence between typing pairs and natural deduction, we can prove theorems for typing pairs and transfer the result to natural deduction derivations.

Similar to what we did in the natural deduction section, we can make some observations here too. Let $\Gamma \vdash M : \varphi$ denote that there is a pair $(\Gamma, M)$ witnessing the formula $\varphi$. Then always $\Gamma \vdash x : \varphi$ if $x : \varphi \in \Gamma$, and the following rules are valid:

$$\frac{\Gamma \vdash M_1 : \varphi_1 \qquad \Delta \vdash M_2 : \varphi_2}{\Gamma, \Delta \vdash \langle M_1, M_2 \rangle : \varphi_1 \wedge \varphi_2} \wedge\text{Intro} \qquad \frac{\Gamma \vdash M : \varphi_1 \wedge \varphi_2}{\Gamma \vdash \mathrm{p}_i(M) : \varphi_i} \wedge\text{Elim}_i$$

$$\frac{\Gamma \vdash M_1 : \varphi_1}{\Gamma \vdash \mathrm{in}_1^{\varphi_2}(M) : \varphi_1 \vee \varphi_2} \vee\text{Intro}_1 \qquad \frac{\Gamma \vdash M_2 : \varphi_2}{\Gamma \vdash \mathrm{in}_2^{\varphi_1}(M) : \varphi_1 \vee \varphi_2} \vee\text{Intro}_2$$

$$\frac{\Gamma \vdash M : \varphi \vee \psi \qquad \Delta_1, x_1 : \varphi_1 \vdash N_1 : \chi \qquad \Delta_2, x_2 : \varphi_2 \vdash N_2 : \chi}{\Gamma, \Delta, \Delta' \vdash \mathrm{case}(M, x_1.N_1, x_2.N_2) : \chi} \vee\text{Elim}$$

$$\frac{\Gamma, x : \varphi \vdash N : \psi}{\Gamma \vdash \lambda x^{\varphi}.\, N : \varphi \to \psi} \to\text{Intro} \qquad \frac{\Gamma \vdash Q : \varphi \qquad \Delta \vdash P : \varphi \to \psi}{\Gamma, \Delta \vdash PQ : \psi} \to\text{Elim}$$

$$\frac{\Gamma \vdash M : \bot}{\Gamma \vdash \mathrm{contr}_{\varphi}(M) : \varphi} \bot\text{Elim}$$

These are the typing rules of the simply typed lambda calculus extended with product, sum and bottom.

In addition, the $F(\Gamma, M)$ is actually a type checking algorithm; it returns the type of the term with respect to the context, or is undefined if the term is ill-typed with respect to the context.

## 47.6 Reduction

In natural deduction derivations, an introduction rule that is followed by an elimination rule is redundant. For instance, the derivation

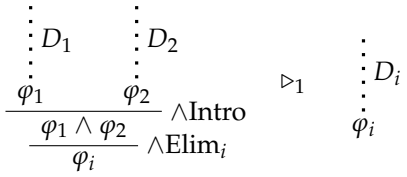
$$\dfrac{\dfrac{\dfrac{\varphi \qquad \varphi \to \psi}{\psi} \to\text{Elim} \qquad [\chi]}{\psi \wedge \chi} \wedge\text{Intro}}{\dfrac{\psi}{\chi \to \psi} \wedge\text{Elim}} \to\text{Intro}$$

can be replaced with the simpler derivation:

$$\dfrac{\dfrac{\varphi \qquad \varphi \to \psi}{\psi} \to\text{Elim}}{\chi \to \psi} \to\text{Intro}$$

As we see, an $\wedge$Intro followed by $\wedge$Elim "cancel out." In general, we see that the conclusion of $\wedge$Elim is always the formula on one side of the conjunction, and the premises of $\wedge$Intro requires both sides of the conjunction, thus if we need a derivation of either side, we can simply use that derivation without introducing the conjunction followed by eliminating it.

Thus in general we have




The $\triangleright_1$ symbol has a similar meaning as in the lambda calculus, i.e., a single step of a reduction. In the proof term syntax for derivations, the above reduction rule thus becomes:

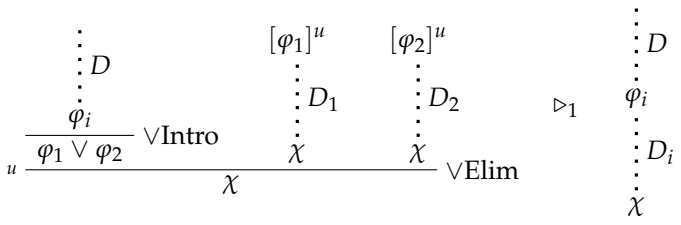
$$(\Gamma, p_i\langle M_1^{\varphi_1}, M_2^{\varphi_2}\rangle) \triangleright_1 (\Gamma, M_i)$$

In the typed lambda calculus, this is the beta reduction rule for the product type.

Note the type annotation on $M_1$ and $M_2$: while in the standard term syntax only $\lambda x^{\varphi}. N$ has such notion, we reuse the notation here to remind us of the formula the term is associated with in the corresponding natural deduction derivation, to reveal the correspondence between the two kinds of syntax.

In natural deduction, a pair of inferences such as those on the left, i.e., a pair that is subject to cancelling is called a *cut*. In the typed lambda calculus the term on the left of $\triangleright_1$ is called a *redex*, and the term to the right is called the *reductum*. Unlike untyped lambda calculus, where only $(\lambda x. N)Q$ is considered to be redex, in the typed lambda calculus the syntax is extended to terms involving $\langle N, M\rangle$, $\mathrm{p}_i(N)$, $\mathrm{in}_i^{\varphi}(N)$, $\mathrm{case}(N, x_1.M_1, x_2.M_2)$, and $\mathrm{contr}_N()$, with corresponding redexes.

Similarly we have reduction for disjunction:




This corresponds to a reduction on proof terms:

$$(\Gamma, \mathrm{case}(\mathrm{in}_i^{\varphi_i}(M^{\varphi_i}), x_1^{\varphi_1}.N_1^{\chi}, x_2^{\varphi_2}.N_2^{\chi})) \triangleright_1 (\Gamma, N_i^{\chi}[M^{\varphi_i}/x_i^{\varphi_i}])$$

This is the beta reduction rule of for sum types. Here, $M[N/x]$ means replacing all assumptions denoted by variable $x$ in $M$ with $N$,

It would be nice if we pass the context $\Gamma$ to the substitution function so that it can check if the substitution makes sense. For example, $xy[ab/y]$ does not make sense under the context $\{x : \varphi \to \theta, y : \varphi, a : \psi \to \chi, b : \psi\}$ since then we

would be substituting a derivation of $\chi$ where a derivation of $\varphi$ is expected. However, as long as our usage of substitution is careful enough to avoid such errors, we won't have to worry about such conflicts. Thus we can define it recursively as we did for untyped lambda calculus as if we are dealing with untyped terms.

Finally, the reduction of the function type corresponds to removal of a detour of a $\to$Intro followd by a $\to$Elim.

$$\dfrac{\dfrac{\begin{array}{c}[\varphi]^u\\ \vdots D \vdots\\ \psi\end{array}}{{}^u\,\varphi \to \psi}\to\text{Intro} \qquad \begin{array}{c}\vdots D' \vdots\\ \varphi\end{array}}{\psi}\to\text{Elim} \qquad \rhd_1 \qquad \begin{array}{c}\vdots D' \vdots\\ \varphi\\ \vdots D \vdots\\ \psi\end{array}$$

For proof terms, this amounts to ordinary beta reduction:

$$(\Gamma, (\lambda x^{\varphi}.\, N^{\psi})Q^{\varphi}) \rhd_1 (\Gamma, N^{\psi}[Q^{\varphi}/x^{\varphi}])$$

Absurdity has only an elimination rule and no introduction rule, thus there is no such reduction for it.

Note that the above notion of reduction concerns only deductions with a cut at the end of a derivation. We would of course like to extend it to reduction of cuts anywhere in a derivation, or reductions of subterms of proof terms which constitute redexes. Note that, however, the conclusion of the reduction does not change after reduction, thus we are free to continue applying rules to both sides of $\rhd_1$. The resulting pairs of trees constitutes an extended notion of reduction; it is analogous to compatibility in the untyped lambda calculus.

It's easy to see that the context $\Gamma$ does not change during the reduction (both the original and the extended version), thus it's unnecessary to mention the context when we are discussing reductions. In what follows we will assume that every term is accompanied by a context which does no change during reduction. We then say "proof term" when we mean a proof term accompanied by a context which makes it well-typed.

As in lambda calculus, the notion of normal-form term and normal deduction is given:

**Definition 47.5.** A proof term with no redex is said to be in *normal form*; likewise, a derivation without cuts is a *normal derivation*. A proof term is in normal form if and only if its counterpart derivation is normal.

## 47.7 Normalization

In this section we prove that, via some reduction order, any deduction can be reduced to a normal deduction, which is called the *normalization property*.

We will make use of the propositions-as-types correspondence: we show that every proof term can be reduced to a normal form; normalization for natural deduction derivations then follows.

Firstly we define some functions that measure the complexity of terms. The *length* $\mathrm{len}(\varphi)$ of a formulas is defined by

$$\begin{aligned}\mathrm{len}(p) &= 0\\ \mathrm{len}(\varphi \wedge \psi) &= \mathrm{len}(\varphi) + \mathrm{len}(\psi) + 1\\ \mathrm{len}(\varphi \vee \psi) &= \mathrm{len}(\varphi) + \mathrm{len}(\psi) + 1\\ \mathrm{len}(\varphi \to \psi) &= \mathrm{len}(\varphi) + \mathrm{len}(\psi) + 1.\end{aligned}$$

The complexity of a redex $M$ is measured by its *cut rank* $\mathrm{cr}(M)$:

$$\begin{aligned}\mathrm{cr}((\lambda x^{\varphi}.\, N^{\psi})Q) &= \mathrm{len}(\varphi) + \mathrm{len}(\psi) + 1\\ \mathrm{cr}(\mathrm{p}_i(\langle M^{\varphi}, N^{\psi}\rangle)) &= \mathrm{len}(\varphi) + \mathrm{len}(\psi) + 1\\ \mathrm{cr}(\mathrm{case}(\mathrm{in}_i^{\varphi_i}(M^{\varphi_i}), x_1^{\varphi_1}.N_1^{\chi}, x_2^{\varphi_2}.N_2^{\chi})) &= \mathrm{len}(\varphi) + \mathrm{len}(\psi) + 1\end{aligned}$$

The complexity of a proof term is measured by the most complex redex in it, and 0 if it is normal:

$$\mathrm{mr}(M) = \max\{\mathrm{cr}(N) | N \text{ is a sub term of } M \text{ and is redex}\}$$

**Lemma 47.6.** *If $M[N^{\varphi}/x^{\varphi}]$ is a redex and $M \not\equiv x$, then one of the following cases holds:*

1. *$M$ is itself a redex, or*
2. *$M$ is of the form $\mathrm{p}_i(x)$, and $N$ is of the form $\langle P_1, P_2\rangle$*
3. *$M$ is of the form $\mathrm{case}(i, x_1.P_1, x_2.P_2)$, and $N$ is of the form $\mathrm{in}_i(Q)$*
4. *$M$ is of the form $xQ$, and $N$ is of the form $\lambda x.\, P$*

*In the first case, $\mathrm{cr}(M[N/x]) = \mathrm{cr}(M)$; in the other cases, $\mathrm{cr}(M[N/x]) = \mathrm{len}(\varphi)$).*

*Proof.* Proof by induction on $M$.

1. If $M$ is a single variable $y$ and $y \not\equiv x$, then $y[N/x]$ is $y$, hence not a redex.
2. If $M$ is of the form $\langle N_1, N_2\rangle$, or $\lambda x.\, N$, or $\mathrm{in}_i^{\varphi}(N)$, then $M[N^{\varphi}/x^{\varphi}]$ is also of that form, and so is not a redex.
3. If $M$ is of the form $\mathrm{p}_i(P)$, we consider two cases.
   a) If $P$ is of the form $\langle P_1, P_2\rangle$, then $M \equiv \mathrm{p}_i(\langle P_1, P_2\rangle)$ is a redex, and clearly
   $$M[N/x] \equiv \mathrm{p}_i(\langle P_1[N/x], P_2[N/x]\rangle)$$
   is also a redex. The cut ranks are equal.

b) If $P$ is a single variable, it must be $x$ to make the substitution a redex, and $N$ must be of the form $\langle P_1, P_2 \rangle$. Now consider

$$M[N/x] \equiv \mathrm{p}_i(x)[\langle P_1, P_2 \rangle / x],$$

which is $\mathrm{p}_i(\langle P_1, P_2 \rangle)$. Its cut rank is equal to $\mathrm{cr}(x)$, which is $\mathrm{len}(\varphi)$.

The cases of $\mathrm{case}(N, x_1.N_1, x_2.N_2)$ and $PQ$ are similar. □

**Lemma 47.7.** *If $M$ contracts to $M'$, and $\mathrm{cr}(M) > \mathrm{cr}(N)$ for all proper redex subterms $N$ of $M$, then $\mathrm{cr}(M) > \mathrm{mr}(M')$.*

*Proof.* Proof by cases.

1. If $M$ is of the form $\mathrm{p}_i(\langle M_1, M_2 \rangle)$, then $M'$ is $M_i$; since any sub-term of $M_i$ is also proper sub-term of $M$, the claim holds.
2. If $M$ is of the form $(\lambda x^\varphi. N)Q^\varphi$, then $M'$ is $N[Q^\varphi / x^\varphi]$. Consider a redex in $M'$. Either there is corresponding redex in $N$ with equal cut rank, which is less than $\mathrm{cr}(M)$ by assumption, or the cut rank equals $\mathrm{len}(\varphi)$, which by definition is less than $\mathrm{cr}((\lambda x^\varphi. N)Q)$.
3. If $M$ is of the form

$$\mathrm{case}(\mathrm{in}_i(N^{\varphi_i}), x_1^{\varphi_1}.N_1^\chi, x_2^{\varphi_2}.N_2^\chi),$$

then $M' \equiv N_i[N/x_i^{\varphi_i}]$. Consider a redex in $M'$. Either there is corresponding redex in $N_i$ with equal cut rank, which is less than $\mathrm{cr}(M)$ by assumption; or the cut rank equals $\mathrm{len}(\varphi_i)$, which by definition is less than $\mathrm{cr}(\mathrm{case}(\mathrm{in}_i(N^{\varphi_i}), x_1^{\varphi_1}.N_1^\chi, x_2^{\varphi_2}.N_2^\chi))$.

□

**Theorem 47.8.** *All proof terms reduce to normal form; all derivations reduce to normal derivations.*

*Proof.* The second follows from the first. We prove the first by complete induction on $m = \mathrm{mr}(M)$, where $M$ is a proof term.

1. If $m = 0$, $M$ is already normal.
2. Otherwise, we proceed by induction on $n$, the number of redexes in $M$ with cut rank equal to $m$.
   a) If $n = 1$, select any redex $N$ such that $m = \mathrm{cr}(N) > \mathrm{cr}(P)$ for any proper sub-term $P$ which is also a redex of course. Such a redex must exist, since any term only has finitely many subterms.
   Let $N'$ denote the reductum of $N$. Now by the lemma $\mathrm{mr}(N') < \mathrm{mr}(N)$, thus we can see that $n$, the number of redexes with $\mathrm{cr}(=)m$ is decreased. So $m$ is decreased (by 1 or more), and we can apply the inductive hypothesis for $m$.

b) For the induction step, assume $n > 1$. the process is similar, except that $n$ is only decreased to a positive number and thus $m$ does not change. We simply apply the induction hypothesis for $n$.

□

The normalization of terms is actually not specific to the reduction order we chose. In fact, one can prove that regardless of the order in which redexes are reduced, the term always reduces to a normal form. This property is called *strong normalization*.

# Part XI

# Counterfactuals

# Chapter 48

# Introduction

## 48.1 The Material Conditional

In its simplest form in English, a conditional is a sentence of the form "If ... then ...," where the ... are themselves sentences, such as "If the butler did it, then the gardener is innocent." In introductory logic courses, we earn to symbolize conditionals using the $\rightarrow$ connective: symbolize the parts indicated by ..., e.g., by formulas $\varphi$ and $\psi$, and the entire conditional is symbolized by $\varphi \rightarrow \psi$.

The connective $\rightarrow$ is *truth-functional*, i.e., the truth value—$\mathbb{T}$ or $\mathbb{F}$—of $\varphi \rightarrow \psi$ is determined by the truth values of $\varphi$ and $\psi$: $\varphi \rightarrow \psi$ is true iff $\varphi$ is false or $\psi$ is true, and false otherwise. Relative to a truth value assignment $\mathfrak{v}$, we define $\mathfrak{v} \vDash \varphi \rightarrow \psi$ iff $\mathfrak{v} \nvDash \varphi$ or $\mathfrak{v} \vDash \psi$. The connective $\rightarrow$ with this semantics is called the *material conditional*.

This definition results in a number of elementary logical facts. First of all, the deduction theorem holds for the material conditional:

$$\text{If } \Gamma, \varphi \vDash \psi \text{ then } \Gamma \vDash \varphi \rightarrow \psi \tag{48.1}$$

It is truth-functional: $\varphi \rightarrow \psi$ and $\neg\varphi \vee \psi$ are equivalent:

$$\varphi \rightarrow \psi \vDash \neg\varphi \vee \psi \tag{48.2}$$

$$\neg\varphi \vee \psi \vDash \varphi \rightarrow \psi \tag{48.3}$$

A material conditional is entailed by its consequent and by the negation of its antecedent:

$$\psi \vDash \varphi \rightarrow \psi \tag{48.4}$$

$$\neg\varphi \vDash \varphi \rightarrow \psi \tag{48.5}$$

A false material conditional is equivalent to the conjunction of its antecedent and the negation of its consequent: if $\varphi \to \psi$ is false, $\varphi \wedge \neg\psi$ is true, and vice versa:

$$\neg(\varphi \to \psi) \vDash \varphi \wedge \neg\psi \tag{48.6}$$
$$\varphi \wedge \neg\psi \vDash \neg(\varphi \to \psi) \tag{48.7}$$

The material conditional supports modus ponens:

$$\varphi, \varphi \to \psi \vDash \psi \tag{48.8}$$

The material conditional agglomerates:

$$\varphi \to \psi, \varphi \to \chi \vDash \varphi \to (\psi \wedge \chi) \tag{48.9}$$

We can always strengthen the antecedent, i.e., the conditional is *monotonic*:

$$\varphi \to \psi \vDash (\varphi \wedge \chi) \to \psi \tag{48.10}$$

The material conditional is transitive, i.e., the chain rule is valid:

$$\varphi \to \psi, \psi \to \chi \vDash \varphi \to \chi \tag{48.11}$$

The material conditional is equivalent to its contrapositive:

$$\varphi \to \psi \vDash \neg\psi \to \neg\varphi \tag{48.12}$$
$$\neg\psi \to \neg\varphi \vDash \varphi \to \psi \tag{48.13}$$

These are all useful and unproblematic inferences in mathematical reasoning. However, the philosophical and linguistic literature is replete with purported counterexamples to the equivalent inferences in non-mathematical contexts. These suggest that the material conditional $\to$ is not—or at least not always—the appropriate connective to use when symbolizing English "if ... then ..." statements.

## 48.2 Paradoxes of the Material Conditional

One of the first to criticize the use of $\varphi \to \psi$ as a way to symbolize "if ... then ..." statements of English was C. I. Lewis. Lewis was criticizing the use of the material condition in Whitehead and Russell's *Principia Mathematica*, who pronounced $\to$ as "implies." Lewis rightly complained that if $\to$ meant "implies," then any false proposition $p$ implies that $p$ implies $q$, since $p \to (p \to q)$ is true if $p$ is false, and that any true proposition $q$ implies that $p$ implies $q$, since $q \to (p \to q)$ is true if $q$ is true.

Logicians of course know that *implication*, i.e., logical entailment, is not a connective but a relation between formulas or statements. So we should just not read $\rightarrow$ as "implies" to avoid confusion.[1] As long as we don't, the particular worry that Lewis had simply does not arise: $p$ does not "imply" $q$ even if we think of $p$ as standing for a false English sentence. To determine if $p \vDash q$ we must consider *all* valuations, and $p \nvDash q$ even when we use $p$ to symbolize a sentence which happens to be false.

But there is still something odd about "if ... then..." statements such as Lewis's

> If the moon is made of green cheese, then $2 + 2 = 4$.

and about the inferences

> The moon is not made of green cheese. Therefore, if the moon is made of green cheese, then $2 + 2 = 4$.
>
> $2 + 2 = 4$. Therefore, if the moon is made of green cheese, then $2 + 2 = 4$.

Yet, if "if ... then ..." were just $\rightarrow$, the sentence would be unproblematically true, and the inferences unproblematically valid.

Another example of concerns the tautology $(\varphi \rightarrow \psi) \vee (\psi \rightarrow \varphi)$. This would suggest that if you take two indicative sentences $S$ and $T$ from the newspaper at random, the sentence "If $S$ then $T$, or if $T$ then $S$" should be true.

## 48.3 The Strict Conditional

Lewis introduced the *strict conditional* $\dashv\!\!\!\supset$ and argued that it, not the material conditional, corresponds to implication. In alethic modal logic, $\varphi \dashv\!\!\!\supset \psi$ can be defined as $\Box(\varphi \rightarrow \psi)$. A strict conditional is thus true (at a world) iff the corresponding material conditional is necessary.

How does the strict conditional fare vis-a-vis the paradoxes of the material conditional? A strict conditional with a false antecedent and one with a true consequent, may be true, or it may be false. Moreover, $(\varphi \dashv\!\!\!\supset \psi) \vee (\psi \dashv\!\!\!\supset \varphi)$ is not valid. The strict conditional $\varphi \dashv\!\!\!\supset \psi$ is also not equivalent to $\neg\varphi \vee \psi$, so it is not truth functional.

[1] Reading "$\rightarrow$" as "implies" is still widely practised by mathematicians and computer scientists, although philosophers try to avoid the confusions Lewis highlighted by pronouncing it as "only if."

We have:

$$\varphi \mathrel{-\!\!\!\supset} \psi \vDash \neg\varphi \vee \psi \text{ but:} \tag{48.14}$$
$$\neg\varphi \vee \psi \nvDash \varphi \mathrel{-\!\!\!\supset} \psi \tag{48.15}$$
$$\psi \nvDash \varphi \mathrel{-\!\!\!\supset} \psi \tag{48.16}$$
$$\neg\varphi \nvDash \varphi \mathrel{-\!\!\!\supset} \psi \tag{48.17}$$
$$\neg(\varphi \to \psi) \nvDash \varphi \wedge \neg\psi \text{ but:} \tag{48.18}$$
$$\varphi \wedge \neg\psi \vDash \neg(\varphi \mathrel{-\!\!\!\supset} \psi) \tag{48.19}$$

However, the strict conditional still supports modus ponens:

$$\varphi, \varphi \mathrel{-\!\!\!\supset} \psi \vDash \psi \tag{48.20}$$

The strict conditional agglomerates:

$$\varphi \mathrel{-\!\!\!\supset} \psi, \varphi \mathrel{-\!\!\!\supset} \chi \vDash \varphi \mathrel{-\!\!\!\supset} (\psi \wedge \chi) \tag{48.21}$$

Antecedent strengthening holds for the strict conditional:

$$\varphi \mathrel{-\!\!\!\supset} \psi \vDash (\varphi \wedge \chi) \mathrel{-\!\!\!\supset} \psi \tag{48.22}$$

The strict conditional is also transitive:

$$\varphi \mathrel{-\!\!\!\supset} \psi, \psi \mathrel{-\!\!\!\supset} \chi \vDash \varphi \mathrel{-\!\!\!\supset} \chi \tag{48.23}$$

Finally, the strict conditional is equivalent to its contrapositive:

$$\varphi \mathrel{-\!\!\!\supset} \psi \vDash \neg\psi \mathrel{-\!\!\!\supset} \neg\varphi \tag{48.24}$$
$$\neg\psi \mathrel{-\!\!\!\supset} \neg\varphi \vDash \varphi \mathrel{-\!\!\!\supset} \psi \tag{48.25}$$

However, the strict conditional still has its own "paradoxes." Just as a material conditional with a false antecedent or a true consequent is true, a strict conditional with a *necessarily* false antecedent or a necessarily true consequent is true. Moreover, any true strict conditional is necessarily true, and any false strict conditional is necessarily false. In other words, we have

$$\Box\varphi \vDash \varphi \mathrel{-\!\!\!\supset} \psi \tag{48.26}$$
$$\Box\neg\psi \vDash \varphi \mathrel{-\!\!\!\supset} \psi \tag{48.27}$$
$$\varphi \mathrel{-\!\!\!\supset} \psi \vDash \Box(\varphi \mathrel{-\!\!\!\supset} \psi) \tag{48.28}$$
$$\neg(\varphi \mathrel{-\!\!\!\supset} \psi) \vDash \Box\neg(\varphi \mathrel{-\!\!\!\supset} \psi) \tag{48.29}$$

These are not problems if you think of $\mathrel{-\!\!\!\supset}$ as "implies." Logical entailment relationships are, after all, mathematical facts and so can't be contingent. But they do raise issues if you want to use $\mathrel{-\!\!\!\supset}$ as a logical connective that is supposed to capture "if . . . then . . . ," especially the last two. For surely there are "if . . . then . . . " statements that are contingently true or contingently false—in fact, they generally are neither necessary nor impossible.

## 48.4 Counterfactuals

A very common and important form of "if ... then ..." constructions in English are built using the past subjunctive form of *to be*: "if it were the case that ... then it would be the case that ..." Because usually the antecedent of such a conditional is false, i.e., counter to fact, they are called *counterfactual conditionals* (and because they use the subjunctive form of *to be*, also *subjunctive conditionals*. They are distinguished from *indicative* conditionals which take the form of "if it is the case that ... then it is the case that ..." Counterfactual and indicative conditionals differ in truth conditions. Consider Adams's famous example:

> If Oswald didn't kill Kennedy, someone else did.
>
> If Oswald hadn't killed Kennedy, someone else would have.

The first is indicative, the second counterfactual. The first is clearly true: we know JFK was killed by someone, and if that someone wasn't (contrary to the Warren Report) Lee Harvey Oswald, then someone else killed JFK. The second one says something different. It claims that if Oswald hadn't killed Kennedy, i.e., if the Dallas shooting had been avoided or had been unsuccessful, history would have subsequently unfolded in such a way that another assassination would have been successful. In order for it to be true, it would have to be the case that powerful forces had conspired to ensure JFK's death (as many JFK conspiracy theorists believe).

It is a live debate whether the *indicative* conditional is correctly captured by the material conditional, in particular, whether the paradoxes of the material conditional can be "explained" in a way that is compatible with it giving the truth conditions for English indicative conditionals. By contrast, it is uncontroversial that counterfactual conditionals cannot be symbolized correctly by the material conditionals. That is clear because, even though generally the antecedents of counterfactuals are false, not all counterfactuals with false antecedents are true—for instance, if you believe the Warren Report, and there was no conspiracy to assassinate JFK, then Adams's counterfactual conditional is an example.

Counterfactual conditionals play an important role in causal reasoning: a prime example of the use of counterfactuals is to express causal relationships. E.g., striking a match causes it to light, and you can express this by saying "if this match were struck, it would light." Material, and generally indicative conditionals, cannot be used to express this: "the match is struck $\rightarrow$ the match lights" is true if the match is never struck, regardless of what would happen if it were. Even worse, "the match is struck $\rightarrow$ the match turns into a bouquet of flowers" is also true if it is never struck, but the match would certainly not turn into a bouquet of flowers if it were struck.

It is still debated What exactly the correct logic of counterfactuals is. An influential analysis of counterfactuals was given by Stalnaker and Lewis. According to them, a counterfactual "if it were the case that $S$ then it would be the case that $T$" is true iff $T$ is true in the counterfactual situation ("possible world") that is closest to the way the actual world is and where $S$ is true. This is called an "ontic" analysis, since it makes reference to an ontology of possible worlds. Other analyses make use of conditional probabilities or theories of belief revision. There is a proliferation of different proposed logics of counterfactuals. There isn't even a single Lewis-Stalnaker logic of counterfactuals: even though Stalnaker and Lewis proposed accounts along similar lines with reference to closest possible worlds, the assumptions they made result in different valid inferences.

## Problems

**Problem 48.1.** Give **S5**-counterexamples to the entailment relations which do not hold for the strict conditional, i.e., for:

1. $\neg p \nvDash \Box(p \to q)$
2. $q \nvDash \Box(p \to q)$
3. $\neg\Box(p \to q) \nvDash p \wedge \neg q$
4. $\nvDash \Box(p \to q) \vee \Box(q \to p)$

**Problem 48.2.** Show that the valid entailment relations hold for the strict conditional by giving **S5**-proofs of:

1. $\Box(\varphi \to \psi) \vDash \neg\varphi \vee \psi$
2. $\varphi \wedge \neg\psi \vDash \neg\Box(\varphi \to \psi)$
3. $\varphi, \Box(\varphi \to \psi) \vDash \psi$
4. $\Box(\varphi \to \psi), \Box(\varphi \to \chi) \vDash \Box(\varphi \to (\psi \wedge \chi))$
5. $\Box(\varphi \to \psi) \vDash \Box((\varphi \wedge \chi) \to \psi)$
6. $\Box(\varphi \to \psi), \Box(\psi \to \chi) \vDash \Box(\varphi \to \chi)$
7. $\Box(\varphi \to \psi) \vDash \Box(\neg\psi \to \neg\varphi)$
8. $\Box(\neg\psi \to \neg\varphi) \vDash \Box(\varphi \to \psi)$

**Problem 48.3.** Give proofs in **S5** of:

1. $\Box\neg\psi \vDash \varphi \dashv \psi$
2. $\varphi \dashv \psi \vDash \Box(\varphi \dashv \psi)$

3. $\neg(\varphi \dashv \psi) \vDash \Box\neg(\varphi \dashv \psi)$

Use the definition of $\dashv$ to do so.

# Chapter 49

# Minimal Change Semantics

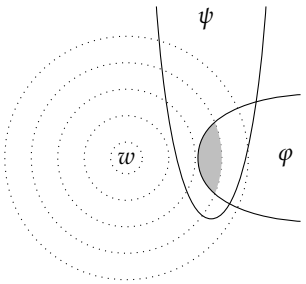
## 49.1 Introduction

Stalnaker and Lewis proposed accounts of counterfactual conditionals such as "If the match were struck, it would light." Their accounts were proposals for how to properly understand the truth conditions for such sentences. The idea behind both proposals is this: to evaluate whether a counterfactual conditional is true, we have to consider those possible worlds which are minimally different from the way the world actually is to make the antecedent true. If the consequent is true in these possible worlds, then the counterfactual is true. For instance, suppose I hold a match and a matchbook in my hand. In the actual world I only look at them and ponder what would happen if I were to strike the match. The minimal change from the actual world where I strike the match is that where I decide to act and strike the match. It is minimal in that nothing else changes: I don't also jump in the air, striking the match doesn't also light my hair on fire, I don't suddenly lose all strength in my fingers, I am not simultaneously doused with water in a SuperSoaker ambush, etc. In that alternative possibility, the match lights. Hence, it's true that if I were to strike the match, it would light.

This intuitive account can be paired with formal semantics for logics of counterfactuals. Lewis introduced the symbol "$\boxrightarrow$" for the counterfactual while Stalnaker used the symbol "$>$". We'll use $\boxrightarrow$, and add it as a binary connective to propositional logic. So, we have, in addition to formulas of the form $\varphi \to \psi$ also formulas of the form $\varphi \boxrightarrow \psi$. The formal semantics, like the relational semantics for modal logic, is based on models in which formulas are evaluated at worlds, and the satisfaction condition defining $\mathfrak{M}, w \Vdash \varphi \boxrightarrow \psi$ is given in terms of $\mathfrak{M}, w' \Vdash \varphi$ and $\mathfrak{M}, w' \Vdash \psi$ for some (other) worlds $w'$. Which $w'$? Intuitively, the one(s) closest to $w$ for which it holds that $\mathfrak{M}, w' \Vdash \varphi$. This requires that a relation of "closeness" has to be included in the model as well.

Lewis introduced an instructive way of representing counterfactual situations graphically. Each possible world is at the center of a set of nested spheres

containing other worlds—we draw these spheres as concentric circles. The worlds between two spheres are equally close to the world at the center as each other, those contained in a nested sphere are closer, and those in a surrounding sphere further away.




The closest $\varphi$-worlds are those worlds $w'$ where $\varphi$ is satisfied which lie in the smallest sphere around the center world $w$ (the gray area). Intuitively, $\varphi \boxdot\!\!\rightarrow \psi$ is satisfied at $w$ if $\psi$ is true at all closest $\varphi$-worlds.

## 49.2 Sphere Models

One way of providing a formal semantics for counterfactuals is to turn Lewis's informal account into a mathematical structure. The spheres around a world $w$ then are sets of worlds. Since the spheres are nested, the sets of worlds around $w$ have to be linearly ordered by the subset relation.

**Definition 49.1.** A *sphere model* is a triple $\mathfrak{M} = \langle W, O, V\rangle$ where $W$ is a non-empty set of worlds, $V\colon \mathrm{At}_0 \to \wp(W)$ is a valuation, and $O\colon W \to \wp(\wp(W))$ assigns to each world $w$ a *system of spheres* $O_w$. For each $w$, $O_w$ is a set of sets of worlds, and must satisfy:

1. $O_w$ is *centered* on $w$: $\{w\} \in O_w$.
2. $O_w$ is *nested*: whenever $S_1$, $S_2 \in O_w$, $S_1 \subseteq S_2$ or $S_2 \subseteq S_1$, i.e., $O_w$ is linearly ordered by $\subseteq$.
3. $O_w$ is closed under non-empty unions.
4. $O_w$ is closed under non-empty intersections.

The intuition behind $O_w$ is that the worlds "around" $w$ are stratified according to how far away they are from $w$. The innermost sphere is just $w$ by itself, i.e., the set $\{w\}$: $w$ is closer to $w$ than the worlds in any other sphere. If $S \subsetneq S'$, then the worlds in $S' \setminus S$ are further way from $w$ than the worlds in $S$: $S' \setminus S$ is the "layer" between the $S$ and the worlds outside of $S'$. In particular, we have to think of the spheres as containing all the worlds within their outer surface; they are not just the individual layers.

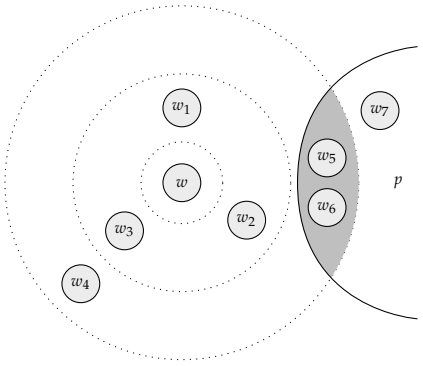



Figure 49.1: Diagram of a sphere model

The diagram in Figure 49.1 corresponds to the sphere model with $W = \{w, w_1, \ldots, w_7\}$, $V(p) = \{w_5, w_6, w_7\}$. The innermost sphere $S_1 = \{w\}$. The closest worlds to $w$ are $w_1, w_2, w_3$, so the next larger sphere is $S_2 = \{w, w_1, w_2, w_3\}$. The worlds further out are $w_4, w_5, w_6$, so the outermost sphere is $S_3 = \{w, w_1, \ldots, w_6\}$. The system of spheres around $w$ is $O_w = \{S_1, S_2, S_3\}$. The world $w_7$ is not in any sphere around $w$. The closest worlds in which $p$ is true are $w_5$ and $w_6$, and so the smallest $p$-admitting sphere is $S_3$.

To define satisfaction of a formula $\varphi$ at world $w$ in a sphere model $\mathfrak{M}$, $\mathfrak{M}, w \Vdash \varphi$, we expand the definition for modal formulas to include a clause for $\psi \boxdot\!\!\rightarrow \chi$:

**Definition 49.2.** $\mathfrak{M}, w \Vdash \psi \boxdot\!\!\rightarrow \chi$ iff either

1. For all $u \in \bigcup O_w$, $\mathfrak{M}, u \nVdash \chi$, or
2. For some $S \in O_w$,
   a) $\mathfrak{M}, u \Vdash \psi$ for some $u \in S$, and
   b) for all $v \in S$, either $\mathfrak{M}, v \nVdash \psi$ or $\mathfrak{M}, v \Vdash \chi$.

According to this definition, $\mathfrak{M}, w \Vdash \psi \boxdot\!\!\rightarrow \chi$ iff either the antecedent $\psi$ is false everywhere in the spheres around $w$, or there is a sphere $S$ where $\psi$ is true, and the material conditional $\psi \rightarrow \chi$ is true at all worlds in that "$\psi$-admitting" sphere. Note that we didn't require in the definition that $S$ is the *innermost* $\psi$-admitting sphere, contrary to what one might expect from the intuitive explanation. But if the condition in (2) is satisfied for some sphere $S$, then it is also satisfied for all spheres $S$ contains, and hence in particular for the innermost sphere.

Note also that the definition of sphere models does not require that there *is* an innermost $\psi$-admitting sphere: we may have an infinite sequence $S_1 \supsetneq$

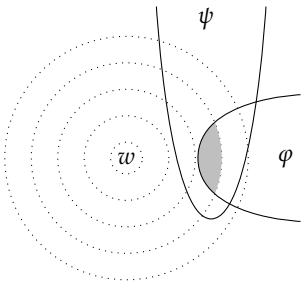



Figure 49.2: Non-vacuously true counterfactual

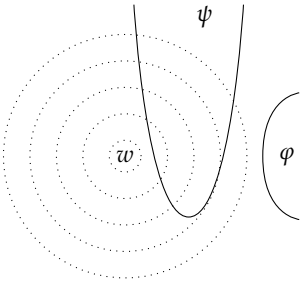



Figure 49.3: Vacuously true counterfactual

$S_2 \supsetneq \cdots \supsetneq \{w\}$ of $\psi$-admitting spheres, and hence no innermost $\psi$-admitting spheres. In that case, $\mathfrak{M}, w \Vdash \psi \boxdot\!\!\rightarrow \chi$ iff $\psi \rightarrow \chi$ holds throughout the spheres $S_i, S_{i+1}, \ldots$, for some $i$.

## 49.3 Truth and Falsity of Counterfactuals

A counterfactual $\varphi \boxdot\!\!\rightarrow \psi$ is (non-vacuously) true if the closest $\varphi$-worlds are all $\psi$-worlds, as depicted in Figure 49.2. A counterfactual is also true at $w$ if the system of spheres around $w$ has no $\varphi$-admitting spheres at all. In that case it is *vacuously* true (see Figure 49.3).

It can be false in two ways. One way is if the closest $\varphi$-worlds are not all $\psi$-worlds, but some of them are. In this case, $\varphi \boxdot\!\!\rightarrow \neg\psi$ is also false (see Figure 49.4). If the closest $\varphi$-worlds do not overlap with the $\psi$-worlds at all, then $\varphi \boxdot\!\!\rightarrow \psi$. But, in this case all the closest $\varphi$-worlds are $\neg\psi$-worlds, and so $\varphi \boxdot\!\!\rightarrow \neg\psi$ is true (see Figure 49.5).

In contrast to the strict conditional, counterfactuals may be contingent. Consider the sphere model in Figure 49.6. The $\varphi$-worlds closest to $u$ are all $\psi$-worlds, so $\mathfrak{M}, u \Vdash \varphi \boxdot\!\!\rightarrow \psi$. But there are $\varphi$-worlds closest to $v$ which are not

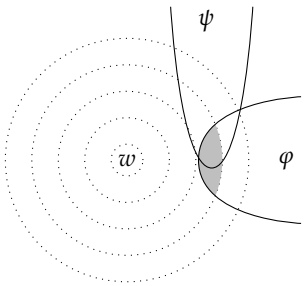



Figure 49.4: False counterfactual, false opposite

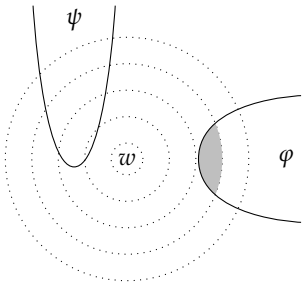



Figure 49.5: False counterfactual, true opposite

$\psi$-worlds, so $\mathfrak{M}, v \nVdash \varphi \boxdot\!\!\rightarrow \psi$.

## 49.4 Antecedent Strengthenng

"Strengthening the antecedent" refers to the inference $\varphi \rightarrow \chi \vDash (\varphi \wedge \psi) \rightarrow \chi$. It is valid for the material conditional, but invalid for counterfactuals. Suppose it is true that if I were to strike this match, it would light. (That means, there is nothing wrong with the match or the matchbook surface, I will not break the match, etc.) But it is not true that if I were to light this match in outer space, it would light. So the following inference is invalid:

> I the match were struck, it would light.
>
> Therefore, if the match were struck in outer space, it would light.

The Lewis-Stalnaker account of conditionals explains this: the closest world where I light the match and I do so in outer space is much further removed from the actual world than the closest world where I light the match is. So although it's true that the match lights in the latter, it is not in the former. And that is as it schould be.

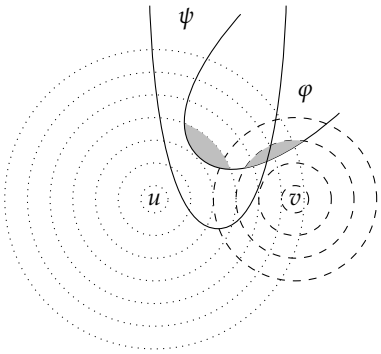



Figure 49.6: Contingent counterfactual

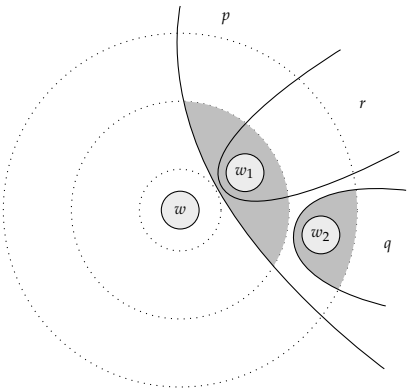



Figure 49.7: Counterexample to antecedent strengthening

**Example 49.3.** The sphere semantics invalidates the inference, i.e., we have $p \boxrightarrow r \not\models (p \wedge q) \boxrightarrow r$. Consider the model $\mathfrak{M} = \langle W, O, V \rangle$ where $W = \{w, w_1, w_2\}$, $O_w = \{\{w\}, \{w, w_1\}, \{w, w_1, w_2\}\}$, $V(p) = \{w_1, w_2\}$, $V(q) = \{w_2\}$, and $V(r) = \{w_1\}$. There is a $p$-admitting sphere $S = \{w, w_1\}$ and $p \to r$ is true at all worlds in it, so $\mathfrak{M}, w \Vdash p \boxrightarrow r$. There is also a $(p \wedge q)$-admitting sphere $S' = \{w, w_1, w_2\}$ but $\mathfrak{M}, w_2 \nVdash (p \wedge q) \to r$, so $\mathfrak{M}, w \nVdash (p \wedge q) \boxrightarrow r$ (see Figure 49.7).

## 49.5 Transitivity

For the material conditional, the chain rule holds: $\varphi \to \psi, \psi \to \chi \models \varphi \to \chi$. In other words, the material conditional is transitive. Is the same true for counterfactuals? Consider the following example due to Stalnaker.

> If J. Edgar Hoover had been born a Russian, he would have been a Communist.
>
> If J. Edgar Hoover were a Communist, he would have been be a traitor.
>
> Therefore, If J. Edgar Hoover had been born a Russian, he would have been be a traitor.

If Hoover had been born (at the same time he actually did), not in the United States, but in Russia, he would have grown up in the Soviet Union and become a Communist (let's assume). So the first premise is true. Likewise, the second premise, considered in isolation is true. The conclusion, however, is false: in all likelihood, Hoover would have been a fervent Communist if he had been born in the USSR, and not been a traitor (to his country). The intuitive assignment of truth values is borne out by the Stalnaker-Lewis account. The closest possible world to ours with the only change being Hoover's place of birth is the one where Hoover grows up to be a good citizen of the USSR. This is the closest possible world where the antecedent of the first premise and of the conclusion is true, and in that world Hoover is a loyal member of the Communist party, and so not a traitor. To evaluate the second premise, we have to look at a different world, however: the closest world where Hoover is a Communist, which is one where he was born in the United States, turned, and thus became a traitor.[1]

**Example 49.4.** The sphere semantics invalidates the inference, i.e., we have $p \boxdot\!\!\rightarrow q, q \boxdot\!\!\rightarrow r \not\models p \boxdot\!\!\rightarrow r$. Consider the model $\mathfrak{M} = \langle W, O, V\rangle$ where $W = \{w, w_1, w_2\}, O_w = \{\{w\}, \{w, w_1\}, \{w, w_1, w_2\}\}, V(p) = \{w_2\}, V(q) = \{w_1, w_2\}$, and $V(r) = \{w_1\}$. There is a $p$-admitting sphere $S = \{w, w_1, w_2\}$ and $q \rightarrow q$ is true at all worlds in it, so $\mathfrak{M}, w \Vdash p \boxdot\!\!\rightarrow q$. There is also a $q$-admitting sphere $S' = \{w, w_1\}$ and $\mathfrak{M} \nVdash q \rightarrow r$ is true at all worlds in it, so $\mathfrak{M}, w \Vdash q \boxdot\!\!\rightarrow r$. However, the $p$-admitting sphere $\{w, w_1, w_2\}$ contains a world, namely $w_2$, where $\mathfrak{M}, w_2 \nVdash p \rightarrow r$.

## 49.6 Contraposition

Material and strict conditionals are equivalent to their contrapositives. Counterfactuals are not. Here is an example due to Kratzer:

> If Goethe hadn't died in 1832, he would (still) be dead now.
>
> If Goethe weren't dead now, he would have died in 1832.

[1] Of course, to appreciate the force of the example we have to take on board some metaphysical and political assumptions, e.g., that it is possible that Hoover could have been born to Russian parents, or that Communists in the US of the 1950s were traitors to their country.

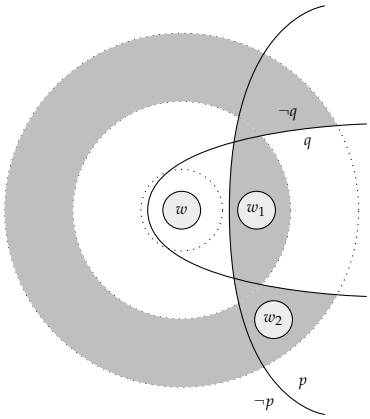



Figure 49.8: Counterexample to contraposition

The first sentence is true: humans don't live hundreds of years. The second is clearly false: if Goethe weren't dead now, he would be still alive, and so couldn't have died in 1832.

**Example 49.5.** The sphere semantics invalidates contraposition, i.e., we have $p \boxdot\!\!\rightarrow q \not\models \neg q \boxdot\!\!\rightarrow \neg p$. Think of $p$ as "Goethe didn't die in 1832" and $q$ as "Goethe is dead now." We can capture this in a model $\mathfrak{M}_1 = \langle W, O, V\rangle$ with $W = \{w, w_1, w_2\}$, $O = \{\{w\}, \{w, w_1\}, \{w, w_1, w_2\}\}$, $V(p) = \{w_1, w_2\}$ and $V(q) = \{w, w_1\}$. So $w$ is the actual world where Goethe died in 1832 and is still dead; $w_1$ is the (close) world where Goethe died in, say, 1833, and is still dead; and $w_2$ is a (remote) world where Goethe is still alive. There is a $p$-admitting sphere $S = \{w, w_1\}$ and $p \rightarrow q$ is true at all worlds in it, so $\mathfrak{M}, w \Vdash p \boxdot\!\!\rightarrow q$. However, the $\neg q$-admitting sphere $\{w, w_1, w_2\}$ contains a world, namely $w_2$, where $q$ is false and $p$ is true, so $\mathfrak{M}, w_2 \nVdash \neg q \rightarrow \neg p$.

## Problems

**Problem 49.1.** Find a convincing, intuitive example for the failure of transitivity of counterfactuals.

**Problem 49.2.** Draw the sphere diagram corresponding to the counterexample in Example 49.4.

**Problem 49.3.** In Example 49.4, world $w_2$ is where Hoover is born in Russia, is a communist, and not a traitor, and $w_1$ is the world where Hoover is born in the US, is a communist, and a traitor. In this model, $w_1$ is closer to $w$ than $w_2$ is. Is this necessary? Can you give a counterexample that does not assume that

Hoover's being born in Russia is a more remote possibility than him being a Communist?

# Part XII

# Methods

This part covers general and methodological material, especially explanations of various proof methods a non-methematics student may be unfamiliar with. It currently contains a chapter on how to write proofs, and a chapter on induction, but additional sections for thos, exercises, and a chapter on mathematical terminology is also planned.

# Chapter 50

# Proofs

## 50.1 Introduction

Based on your experiences in introductory logic, you might be comfortable with a proof system—probably a natural deduction or Fitch style proof system, or perhaps a proof-tree system. You probably remember doing proofs in these systems, either proving a formula or show that a given argument is valid. In order to do this, you applied the rules of the system until you got the desired end result. In reasoning *about* logic, we also prove things, but in most cases we are not using a proof system. In fact, most of the proofs we consider are done in English (perhaps, with some symbolic language thrown in) rather than entirely in the language of first-order logic. When constructing such proofs, you might at first be at a loss—how do I prove something without a proof system? How do I start? How do I know if my proof is correct?

Before attempting a proof, it's important to know what a proof is and how to construct one. As implied by the name, a *proof* is meant to show that something is true. You might think of this in terms of a dialogue—someone asks you if something is true, say, if every prime other than two is an odd number. To answer "yes" is not enough; they might want to know *why*. In this case, you'd give them a proof.

In everyday discourse, it might be enough to gesture at an answer, or give an incomplete answer. In logic and mathematics, however, we want rigorous proof—we want to show that something is true beyond *any* doubt. This means that every step in our proof must be justified, and the justification must be cogent (i.e., the assumption you're using is actually assumed in the statement of the theorem you're proving, the definitions you apply must be correctly applied, the justifications appealed to must be correct inferences, etc.).

Usually, we're proving some statement. We call the statements we're proving by various names: propositions, theorems, lemmas, or corollaries. A proposition is a basic proof-worthy statement: important enough to record, but perhaps not particularly deep nor applied often. A theorem is a signifi-

cant, important proposition. Its proof often is broken into several steps, and sometimes it is named after the person who first proved it (e.g., Cantor's Theorem, the Löwenheim-Skolem theorem) or after the fact it concerns (e.g., the completeness theorem). A lemma is a proposition or theorem that is used to in the proof of a more important result. Confusingly, sometimes lemmas are important results in themselves, and also named after the person who introduced them (e.g., Zorn's Lemma). A corollary is a result that easily follows from another one.

A statement to be proved often contains some assumption that clarifies about which kinds of things we're proving something. It might begin with "Let $\varphi$ be a formula of the form $\psi \to \chi$" or "Suppose $\Gamma \vdash \varphi$" or something of the sort. These are *hypotheses* of the proposition, theorem, or lemma, and you may assume these to be true in your proof. They restrict what we're proving about, and also introduce some names for the objects we're talking about. For instance, if your proposition begins with "Let $\varphi$ be a formula of the form $\psi \to \chi$," you're proving something about all formulas of a certain sort only (namely, conditionals), and it's understood that $\psi \to \chi$ is an arbitrary conditional that your proof will talk about.

## 50.2 Starting a Proof

But where do you even start?

You've been given something to prove, so this should be the last thing that is mentioned in the proof (you can, obviously, *announce* that you're going to prove it at the beginning, but you don't want to use it as an assumption). Write what you are trying to prove at the bottom of a fresh sheet of paper—this way you don't lose sight of your goal.

Next, you may have some assumptions that you are able to use (this will be made clearer when we talk about the *type* of proof you are doing in the next section). Write these at the top of the page and make sure to flag that they are assumptions (i.e., if you are assuming x, write "assume that x," or "suppose that x"). Finally, there might be some definitions in the question that you need to know. You might be told to use a specific definition, or there might be various definitions in the assumptions or conclusion that you are working towards. *Write these down and ensure that you understand what they mean.*

How you set up your proof will also be dependent upon the form of the question. The next section provides details on how to set up your proof based on the type of sentence.

## 50.3 Using Definitions

We mentioned that you must be familiar with all definitions that may be used in the proof, and that you can properly apply them. This is a really impor-

tant point, and it is worth looking at in a bit more detail. Definitions are used to abbreviate properties and relations so we can talk about them more succinctly. The introduced abbreviation is called the *definiendum*, and what it abbreviates is the *definiens*. In proofs, we often have to go back to how the definiendum was introduced, because we have to exploit the logical structure of the definiens (the long version of which the defined term is the abbreviation) to get through our proof. By unpacking definitions, you're ensuring that you're getting to the heart of where the logical action is.

We'll start with an example. Suppose you want to prove the following:

**Proposition 50.1.** *For any sets $X$ and $Y$, $X \cup Y = Y \cup X$.*

In order to even start the proof, we need to know what it means for two sets to be identical; i.e., we need to know what the "$=$" in that equation means for sets. Sets are defined to be identical whenever they have the same elements. So the definition we have to unpack is:

**Definition 50.2.** Sets $X$ and $Y$ are *identical*, $X = Y$, iff every element of $X$ is an element of $Y$, and vice versa.

This definition uses $X$ and $Y$ as placeholders for arbitrary sets. What it defines—the *definiendum*—is the expression "$X = Y$" by giving the condition under which $X = Y$ is true. This condition—"every element of $X$ is an element of $Y$, and vice versa"—is the *definiens*.[1] The definition specifies that $X = Y$ is true if, and only if (we abbreviate this to "iff") the condition holds.

When you apply the definition, you have to match the $X$ and $Y$ in the definition to the case you're dealing with. In our case, it means that in order for $X \cup Y = Y \cup X$ to be true, each $z \in X \cup Y$ must also be in $Y \cup X$, and vice versa. The expression $X \cup Y$ in the proposition plays the role of $X$ in the definition, and $Y \cup X$ that of $Y$. Since $X$ and $Y$ are used both in the definition and in the statement of the proposition we're proving, but in different uses, you have to be careful to make sure you don't mix up the two. For instance, it would be a mistake to think that you could prove the proposition by showing that every element of $X$ is an element of $Y$, and vice versa—that would show that $X = Y$, not that $X \cup Y = Y \cup X$. (Also, since $X$ and $Y$ may be any two sets, you won't get very far, because if nothing is assumed about $X$ and $Y$ they may well be different sets.)

Within the proof we are dealing with set-theoretic notions such as union, and so we must also know the meanings of the symbol $\cup$ in order to understand how the proof should proceed. And sometimes, unpacking the definition gives rise to further definitions to unpack. For instance, $X \cup Y$ is defined as $\{z : z \in X \text{ or } z \in Y\}$. So if you want to prove that $x \in X \cup Y$, unpacking

[1] In this particular case—and very confusingly!—when $X = Y$, the sets $X$ and $Y$ are just one and the same set, even though we use different letters for it on the left and the right side. But the ways in which that set is picked out may be different, and that makes the definition non-trivial.

the definition of $\cup$ tells you that you have to prove $x \in \{z : z \in X \text{ or } z \in Y\}$. Now you also have to remember that $x \in \{z : \ldots z \ldots\}$ iff $\ldots x \ldots$. So, further unpacking the definition of the $\{z : \ldots z \ldots\}$ notation, what you have to show is: $x \in X$ or $x \in Y$. So, "every element of $X \cup Y$ is also an element of $Y \cup X$" really means: "for every $x$, if $x \in X$ or $x \in Y$, then $x \in Y$ or $x \in X$." If we fully unpack the definitions in the proposition, we see that what we have to show is this:

**Proposition 50.3.** *For any sets $X$ and $Y$: (a) for every $x$, if $x \in X$ or $x \in Y$, then $x \in Y$ or $x \in X$, and (b) for every $x$, if $x \in Y$ or $x \in X$, then $x \in X$ or $x \in Y$.*

What's important is that unpacking definitions is a necessary part of constructing a proof. Properly doing it is sometimes difficult: you must be careful to distinguish and match the variables in the definition and the terms in the claim you're proving. In order to be successful, you must know what the question is asking and what all the terms used in the question mean—you will often need to unpack more than one definition. In simple proofs such as the ones below, the solution follows almost immediately from the definitions themselves. Of course, it won't always be this simple.

## 50.4 Inference Patterns

Proofs are composed of individual inferences. When we make an inference, we typically indicate that by using a word like "so," "thus," or "therefore." The inference often relies on one or two facts we already have available in our proof—it may be something we have assumed, or something that we've concluded by an inference already. To be clear, we may label these things, and in the inference we indicate what other statements we're using in the inference. An inference will often also contain an explanation of *why* our new conclusion follows from the things that come before it. There are some common patterns of inference that are used very often in proofs; we'll go through some below. Some patterns of inference, like proofs by induction, are more involved (and will be discussed later).

We've already discussed one pattern of inference: unpacking, or applying, a definition. When we unpack a definition, we just restate something that involves the definiendum by using the definiens. For instance, suppose that we have already established in the course of a proof that $U = V$ (a). Then we may apply the definition of $=$ for sets and infer: "Thus, by definition from (a), every element of $U$ is an element of $V$ and vice versa."

Somewhat confusingly, we often do not write the justification of an inference when we actually make it, but before. Suppose we haven't already proved that $U = V$, but we want to. If $U = V$ is the conclusion we aim for, then we can restate this aim also by applying the definition: to prove $U = V$ we have to prove that every element of $U$ is an element of $V$ and vice versa. So

our proof will have the form: (a) prove that every element of $U$ is an element of $V$; (b) every element of $V$ is an element of $U$; (c) therefore, from (a) and (b) by definition of $=$, $U = V$. But we would usually not write it this way. Instead we might write something like,

> We want to show $U = V$. By definition of $=$, this amounts to showing that every element of $U$ is an element of $V$ and vice versa.
>
> (a) ... (a proof that every element of $U$ is an element of $V$) ...
>
> (b) ... (a proof that every element of $V$ is an element of $U$) ...

### Using a Conjunction

Perhaps the simplest inference pattern is that of drawing as conclusion one of the conjuncts of a conjunction. In other words: if we have assumed or already proved that $p$ and $q$, then we're entitled to infer that $p$ (and also that $q$). This is such a basic inference that it is often not mentioned. For instance, once we've unpacked the definition of $U = V$ we've established that every element of $U$ is an element of $V$ and vice versa. From this we can conclude that every element of $V$ is an element of $U$ (that's the "vice versa" part).

### Proving a Conjunction

Sometimes what you'll be asked to prove will have the form of a conjunction; you will be asked to "prove $p$ and $q$." In this case, you simply have to do two things: prove $p$, and then prove $q$. You could divide your proof into two sections, and for clarity, label them. When you're making your first notes, you might write "(1) Prove $p$" at the top of the page, and "(2) Prove $q$" in the middle of the page. (Of course, you might not be explicitly asked to prove a conjunction but find that your proof requires that you prove a conjunction. For instance, if you're asked to prove that $U = V$ you will find that, after unpacking the definition of $=$, you have to prove: every element of $U$ is an element of $V$ *and* every element of $V$ is an element of $U$).

### Proving a Disjunction

When what you are proving takes the form of a disjunction (i.e., it is an statement of the form "$p$ or $q$"), it is enough to show that one of the disjuncts is true. However, it basically never happens that either disjunct just follows from the assumptions of your theorem. More often, the assumptions of your theorem are themselves disjunctive, or you're showing that all things of a certain kind have one of two properties, but some of the things have the one and others have the other property. This is where proof by cases is useful (see below).

### Conditional Proof

Many theorems you will encounter are in conditional form (i.e., show that if $p$ holds, then $q$ is also true). These cases are nice and easy to set up—simply assume the antecedent of the conditional (in this case, $p$) and prove the conclusion $q$ from it. So if your theorem reads, "If $p$ then $q$," you start your proof with "assume $p$" and at the end you should have proved $q$.

Conditionals may be stated in different ways. So instead of "If $p$ then $q$," a theorem may state that "$p$ only if $q$," "$q$ if $p$," or "$q$, provided $p$." These all mean the same and require assuming $p$ and proving $q$ from that assumption. Recall that a biconditional ("$p$ if and only if (iff) $q$") is really two conditionals put together: if $p$ then $q$, and if $q$ then $p$. All you have to do, then, is two instances of conditional proof: one for the first conditional and another one for the second. Sometimes, however, it is possible to prove an "iff" statement by chaining together a bunch of other "iff" statements so that you start with "$p$" an end with "$q$"—but in that case you have to make sure that each step really is an "iff."

### Universal Claims

Using a universal claim is simple: if something is true for anything, it's true for each particular thing. So if, say, the hypothesis of your proof is $X \subseteq Y$, that means (unpacking the definition of $\subseteq$), that, for every $x \in X$, $x \in Y$. Thus, if you already know that $z \in X$, you can conclude $z \in Y$.

Proving a universal claim may seem a little bit tricky. Usually these statements take the following form: "If $x$ has $P$, then it has $Q$" or "All $P$s are $Q$s." Of course, it might not fit this form perfectly, and it takes a bit of practice to figure out what you're asked to prove exactly. But: we often have to prove that all objects with some property have a certain other property.

The way to prove a universal claim is to introduce names or variables, for the things that have the one property and then show that they also have the other property. We might put this by saying that to prove something for *all* $P$s you have to prove it for an *arbitrary* $P$. And the name introduced is a name for an arbitrary $P$. We typically use single letters as these names for arbitrary things, and the letters usually follow conventions: e.g., we use $n$ for natural numbers, $\varphi$ for formulas, $X$ for sets, $f$ for functions, etc.

The trick is to maintain generality throughout the proof. You start by assuming that an arbitrary object ("$x$") has the property $P$, and show (based only on definitions or what you are allowed to assume) that $x$ has the property $Q$. Because you have not stipulated what $x$ is specifically, other that it has the property $P$, then you can assert that all every $P$ has the property $Q$. In short, $x$ is a stand-in for *all* things with property $P$.

**Proposition 50.4.** *For all sets $X$ and $Y$, $X \subseteq X \cup Y$.*

*Proof.* Let $X$ and $Y$ be arbitrary sets. We want to show that $X \subseteq X \cup Y$. By definition of $\subseteq$, this amounts to: for every $x$, if $x \in X$ then $x \in X \cup Y$. So let $x \in X$ be an arbitrary element of $X$. We have to show that $x \in X \cup Y$. Since $x \in X$, $x \in X$ or $x \in Y$. Thus, $x \in \{x : x \in X \vee x \in Y\}$. But that, by definition of $\cup$, means $x \in X \cup Y$. □

### Proof by Cases

Suppose you have a disjunction as an assumption or as an already established conclusion—you have assumed or proved that $p$ or $q$ is true. You want to prove $r$. You do this in two steps: first you assume that $p$ is true, and prove $r$, then you assume that $q$ is true and prove $r$ again. This works because we assume or know that one of the two alternatives holds. The two steps establish that either one is sufficient for the truth of $r$. (If both are true, we have not one but two reasons for why $r$ is true. It is not necessary to separately prove that $r$ is true assuming both $p$ and $q$.) To indicate what we're doing, we announce that we "distinguish cases." For instance, suppose we know that $x \in Y \cup Z$. $Y \cup Z$ is defined as $\{x : x \in Y \text{ or } x \in Z\}$. In other words, by definition, $x \in Y$ or $x \in Z$. We would prove that $x \in X$ from this by first assuming that $x \in Y$, and proving $x \in X$ from this assumption, and then assume $x \in Z$, and again prove $x \in X$ from this. You would write "We distinguish cases" under the assumption, then "Case (1): $x \in Y$" underneath, and "Case (2): $x \in Z$ halfway down the page. Then you'd proceed to fill in the top half and the bottom half of the page.

Proof by cases is especially useful if what you're proving is itself disjunctive. Here's a simple example:

**Proposition 50.5.** *Suppose $Y \subseteq U$ and $Z \subseteq V$. Then $Y \cup Z \subseteq U \cup V$.*

*Proof.* Assume (a) that $Y \subseteq U$ and (b) $Z \subseteq V$. By definition, any $x \in Y$ is also $\in U$ (c) and any $x \in Z$ is also $\in V$ (d). To show that $Y \cup Z \subseteq U \cup V$, we have to show that if $x \in Y \cup Z$ then $x \in U \cup V$ (by definition of $\subseteq$). $x \in Y \cup Z$ iff $x \in Y$ or $x \in Z$ (by definition of $\cup$). Similarly, $x \in U \cup V$ iff $x \in U$ or $x \in V$. So, we have to show: for any $x$, if $x \in Y$ or $x \in Z$, then $x \in U$ or $x \in V$.

> So far we've only unpacked definitions! We've reformulated our proposition without $\subseteq$ and $\cup$ and are left with trying to prove a universal conditional claim. By what we've discussed above, this is done by assuming that $x$ is something about which we assume the "if" part is true, and we'll go on to show that the "then" part is true as well. In other words, we'll assume that $x \in Y$ or $x \in Z$ and show that $x \in U$ or $x \in V$.[2]

[2] This paragraph just explains what we're doing—it's not part of the proof, and you don't have to go into all this detail when you write down your own proofs.

Suppose that $x \in Y$ or $x \in Z$. We have to show that $x \in U$ or $x \in V$. We distinguish cases.

Case 1: $x \in Y$. By (c), $x \in U$. Thus, $x \in U$ or $x \in V$. (Here we've made the inference discussed in the preceding subsection!)

Case 2: $x \in Z$. By (d), $x \in V$. Thus, $x \in U$ or $x \in V$. □

## Proving an Existence Claim

When asked to prove an existence claim, the question will usually be of the form "prove that there is an $x$ such that $\ldots x \ldots$", i.e., that some object that has the property described by "$\ldots x \ldots$". In this case you'll have to identify a suitable object show that is has the required property. This sounds straightforward, but a proof of this kind can be tricky. Typically it involves *constructing* or *defining* an object and proving that the object so defined has the required property. Finding the right object may be hard, proving that it has the required property may be hard, and sometimes it's even tricky to show that you've succeeded in defining an object at all!

Generally, you'd write this out by specifying the object, e.g., "let $x$ be $\ldots$" (where $\ldots$ specifies which object you have in mind), possibly proving that $\ldots$ in fact describes an object that exists, and then go on to show that $x$ has the property $Q$. Here's a simple example.

**Proposition 50.6.** *Suppose that $x \in Y$. Then there is an $X$ such that $X \subseteq Y$ and $X \neq \emptyset$.*

*Proof.* Assume $x \in Y$. Let $X = \{x\}$.

> Here we've defined the set $X$ by enumerating its elements. Since we assume that $x$ is an object, and we can always form a set by enumerating its elements, we don't have to show that we've succeeded in defining a set $X$ here. However, we still have to show that $X$ has the properties required by the proposition. The proof isn't complete without that!

Since $x \in X$, $X \neq \emptyset$.

> This relies on the definition of $X$ as $\{x\}$ and the obvious facts that $x \in \{x\}$ and $x \notin \emptyset$.

Since $x$ is the only element of $\{x\}$, and $x \in Y$, every element of $X$ is also an element of $Y$. By definition of $\subseteq$, $X \subseteq Y$. □

## Using Existence Claims

Suppose you know that some existence claim is true (you've proved it, or it's a hypothesis you can use), say, "for some $x$, $x \in X$" or "there is an $x \in X$." If

you want to use it in your proof, you can just pretend that you have a name for one of the things which your hypothesis says exist. Since $X$ contains at least one thing, there are things to which that name might refer. You might of course not be able to pick one out or describe it further (other than that it is $\in X$). But for the purpose of the proof, you can pretend that you have picked it out and give a name to it. It's important to pick a name that you haven't already used (or that appears in your hypotheses), otherwise things can go wrong. In your proof, you indicate this by going from "for some $x$, $x \in X$" to "Let $a \in X$." Now you can reason about $a$, use some other hypotheses, etc., until you come to a conclusion, $p$. If $p$ no longer mentions $a$, $p$ is independent of the asusmption that $a \in X$, and you've shown that it follows just from the assumption "for some $x$, $x \in X$."

**Proposition 50.7.** *If $X \neq \emptyset$, then $X \cup Y \neq \emptyset$.*

*Proof.* Suppose $X \neq \emptyset$. So for some $x$, $x \in X$.

> Here we first just restated the hypothesis of the proposition. This hypothesis, i.e., $X \neq \emptyset$, hides an existential claim, which you get to only by unpacking a few definitions. The definition of $=$ tells us that $X = \emptyset$ iff every $x \in X$ is also $\in \emptyset$ and every $x \in \emptyset$ is also $\in X$. Negating both sides, we get: $X \neq \emptyset$ iff either some $x \in X$ is $\notin \emptyset$ or some $x \in \emptyset$ is $\notin X$. Since nothing is $\in \emptyset$, the second disjunct can never be true, and "$x \in X$ and $x \notin \emptyset$" reduces to just $x \in X$. So $x \neq \emptyset$ iff for some $x$, $x \in X$. That's an existence claim. Now we use that existence claim by introducing a name for one of the elements of $X$:

Let $a \in X$.

> Now we've introduced a name for one of the things $\in X$. We'll continue to argue about $a$, but we'll be careful to only assume that $a \in X$ and nothing else:

Since $a \in X$, $a \in X \cup Y$, by definition of $\cup$. So for some $x$, $x \in X \cup Y$, i.e., $X \cup Y \neq \emptyset$.

> In that last step, we went from "$a \in X \cup Y$" to "for some $x$, $x \in X \cup Y$." That doesn't mention $a$ anymore, so we know that "for some $x$, $x \in X \cup Y$" follows from "for some $x$, $x \in X$ alone." But that means that $X \cup Y \neq \emptyset$.

□

It's maybe good practice to keep bound variables like "$x$" separate from hypothtical names like $a$, like we did. In practice, however, we often don't and just use $x$, like so:

> Suppose $X \neq \emptyset$, i.e., there is an $x \in X$. By definition of $\cup$, $x \in X \cup Y$. So $X \cup Y \neq \emptyset$.

However, when you do this, you have to be extra careful that you use different $x$'s and $y$'s for different existential claims. For instance, the following is *not* a correct proof of "If $X \neq \emptyset$ and $Y \neq \emptyset$ then $X \cap Y \neq \emptyset$" (which is not true).

> Suppose $X \neq \emptyset$ and $Y \neq \emptyset$. So for some $x$, $x \in X$ and also for some $x$, $x \in Y$. Since $x \in X$ and $x \in Y$, $x \in X \cap Y$, by definition of $\cap$. So $X \cap Y \neq \emptyset$.

Can you spot where the incorrect step occurs and explain why the result does not hold?

## 50.5 An Example

Our first example is the following simple fact about unions and intersections of sets. It will illustrate unpacking definitions, proofs of conjunctions, of universal claims, and proof by cases.

**Proposition 50.8.** *For any sets $X$, $Y$, and $Z$, $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$*

Let's prove it!

*Proof.* We want to show that for any sets $X$, $Y$, and $Z$, $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$

> First we unpack the definition of "$=$" in the statement of the proposition. Recall that proving sets identical means showing that the sets have the same elements. That is, all elements of $X \cup (Y \cap Z)$ are also elements of $(X \cup Y) \cap (X \cup Z)$, and vice versa. The "vice versa" means that also every element of $(X \cup Y) \cap (X \cup Z)$ must be an element of $X \cup (Y \cap Z)$. So in unpacking the definition, we see that we have to prove a conjunction. Let's record this:

By definition, $X \cup (Y \cap Z) = (X \cup Y) \cap (X \cup Z)$ iff every element of $X \cup (Y \cap Z)$ is also an element of $(X \cup Y) \cap (X \cup Z)$, and every element of $(X \cup Y) \cap (X \cup Z)$ is an element of $X \cup (Y \cap Z)$.

> Since this is a conjunction, we must prove each conjunct separately. Lets start with the first: let's prove that every element of $X \cup (Y \cap Z)$ is also an element of $(X \cup Y) \cap (X \cup Z)$.
>
> This is a universal claim, and so we consider an arbitrary element of $X \cup (Y \cap Z)$ and show that it must also be an element of $(X \cup Y) \cap (X \cup Z)$. We'll pick a variable to call this arbitrary element by, say, $z$. Our proof continues:

First, we prove that every element of $X \cup (Y \cap Z)$ is also an element of $(X \cup Y) \cap (X \cup Z)$. Let $z \in X \cup (Y \cap Z)$. We have to show that $z \in (X \cup Y) \cap (X \cup Z)$.

> Now it is time to unpack the definition of $\cup$ and $\cap$. For instance, the definition of $\cup$ is: $X \cup Y = \{z : z \in X \text{ or } z \in Y\}$. When we apply the definition to "$X \cup (Y \cap Z)$," the role of the "$Y$" in the definition is now played by "$Y \cap Z$," so $X \cup (Y \cap Z) = \{z : z \in X \text{ or } z \in Y \cap Z\}$. So our assumption that $z \in X \cup (Y \cap Z)$ amounts to: $z \in \{z : z \in X \text{ or } z \in Y \cap Z\}$. And $z \in \{z : \ldots z \ldots\}$ iff $\ldots z \ldots$, i.e., in this case, $z \in X$ or $z \in Y \cap Z$.

By the definition of $\cup$, either $z \in X$ or $z \in Y \cap Z$.

> Since this is a disjunction, it will be useful to apply proof by cases. We take the two cases, and show that in each one, the conclusion we're aiming for (namely, "$z \in (X \cup Y) \cap (X \cup Z)$") obtains.

Case 1: Suppose that $z \in X$.

> There's not much more to work from based on our assumptions. So let's look at what we have to work with in the conclusion. We want to show that $z \in (X \cup Y) \cap (X \cup Z)$. Based on the definition of $\cap$, if we want to show that $z \in (X \cup Y) \cap (X \cup Z)$, we have to show that it's in both $(X \cup Y)$ and $(X \cup Z)$. But $z \in X \cup Y$ iff $z \in X$ or $z \in Y$, and we already have (as the assumption of case 1) that $z \in X$. By the same reasoning—switching $Z$ for $Y$—$z \in X \cup Z$. This argument went in the reverse direction, so let's record our reasoning in the direction needed in our proof.

Since $z \in X$, $z \in X$ or $z \in Y$, and hence, by definition of $\cup$, $z \in X \cup Y$. Similarly, $z \in X \cup Z$. But this means that $z \in (X \cup Y) \cap (X \cup Z)$, by definition of $\cap$.

> This completes the first case of the proof by cases. Now we want to derive the conclusion in the second case, where $z \in Y \cap Z$.

Case 2: Suppose that $z \in Y \cap Z$.

> Again, we are working with the intersection of two sets. Let's apply the definition of $\cap$:

Since $z \in Y \cap Z$, $z$ must be an element of both $Y$ and $Z$, by definition of $\cap$.

> It's time to look at our conclusion again. We have to show that $z$ is in both $(X \cup Y)$ and $(X \cup Z)$. And again, the solution is immediate.

Since $z \in Y$, $z \in (X \cup Y)$. Since $z \in Z$, also $z \in (X \cup Z)$. So, $z \in (X \cup Y) \cap (X \cup Z)$.

Here we applied the definitions of $\cup$ and $\cap$ again, but since we've already recalled those definitions, and already showed that if $z$ is in one of two sets it is in their union, we don't have to be as explicit in what we've done.

We've completed the second case of the proof by cases, so now we can assert our first conclusion.

So, if $z \in X \cup (Y \cap Z)$ then $z \in (X \cup Y) \cap (X \cup Z)$.

Now we just want to show the other direction, that every element of $(X \cup Y) \cap (X \cup Z)$ is an element of $X \cup (Y \cap Z)$. As before, we prove this universal claim by assuming we have an arbitrary element of the first set and show it must be in the second set. Let's state what we're about to do.

Now, assume that $z \in (X \cup Y) \cap (X \cup Z)$. We want to show that $z \in X \cup (Y \cap Z)$.

We are now working from the hypothesis that $z \in (X \cup Y) \cap (X \cup Z)$. It hopefully isn't too confusing that we're using the same $z$ here as in the first part of the proof. When we finished that part, all the assumptions we've made there are no longer in effect, so now we can make new assumptions about what $z$ is. If that is confusing to you, just replace $z$ with a different variable in what follows.

We know that $z$ is in both $X \cup Y$ and $X \cup Z$, by definition of $\cap$. And by the definition of $\cup$, we can further unpack this to: either $z \in X$ or $z \in Y$, and also either $z \in X$ or $z \in Z$. This looks like a proof by cases again—except the "and" makes it confusing. You might think that this amounts to there being three possibilities: $z$ is either in $X$, $Y$ or $Z$. But that would be a mistake. We have to be careful, so let's consider each disjunction in turn.

By definition of $\cap$, $z \in X \cup Y$ and $z \in X \cup Z$. By definition of $\cup$, $z \in X$ or $z \in Y$. We distinguish cases.

Since we're focusing on the first disjunction, we haven't gotten our second disjunction (from unpacking $X \cup Z$) yet. In fact, we don't need it yet. The first case is $z \in X$, and an element of a set is also an element of the union of that set with any other. So case 1 is easy:

Case 1: Suppose that $z \in X$. It follows that $z \in X \cup (Y \cap Z)$.

Now for the second case, $z \in Y$. Here we'll unpack the second $\cup$ and do another proof-by-cases:

Case 2: Suppose that $z \in Y$. Since $z \in X \cup Z$, either $z \in X$ or $z \in Z$. We distinguish cases further:

Case 2a: $z \in X$. Then, again, $z \in X \cup (Y \cap Z)$.

> Ok, this was a bit weird. We didn't actually need the assumption that $z \in Y$ for this case, but that's ok.

Case 2b: $z \in Z$. Then $z \in Y$ and $z \in Z$, so $z \in Y \cap Z$, and consequently, $z \in X \cup (Y \cap Z)$.

> This concludes both proofs-by-cases and so we're done with the second half.

So, if $z \in (X \cup Y) \cap (X \cup Z)$ then $z \in X \cup (Y \cap Z)$. □

## 50.6 Another Example

**Proposition 50.9.** *If $X \subseteq Z$, then $X \cup (Z \setminus X) = Z$.*

*Proof.* Suppose that $X \subseteq Z$. We want to show that $X \cup (Z \setminus X) = Z$.

> We begin by observing that this is a conditional statement. It is tacitly universally quantified: the proposition holds for all sets $X$ and $Z$. So $X$ and $Z$ are variables for arbitrary sets. To prove such a statement, we assume the antecedent and prove the consequent.
>
> We continue by using the assumption that $X \subseteq Z$. Let's unpack the definition of $\subseteq$: the assumption means that all elements of $X$ are also elements of $Z$. Let's write this down—it's an important fact that we'll use throughout the proof.

By the definition of $\subseteq$, since $X \subseteq Z$, for all $z$, if $z \in X$, then $z \in Z$.

> We've unpacked all the definitions that are given to us in the assumption. Now we can move onto the conclusion. We want to show that $X \cup (Z \setminus X) = Z$, and so we set up a proof similarly to the last example: we show that every element of $X \cup (Z \setminus X)$ is also an element of $Z$ and, conversely, every element of $Z$ is an element of $X \cup (Z \setminus X)$. We can shorten this to: $X \cup (Z \setminus X) \subseteq Z$ and $Z \subseteq X \cup (Z \setminus X)$. (Here we're doing the opposite of unpacking a definition, but it makes the proof a bit easier to read.) Since this is a conjunction, we have to prove both parts. To show the first part, i.e., that every element of $X \cup (Z \setminus X)$ is also an element of $Z$, we assume that $z \in X \cup (Z \setminus X)$ for an arbitrary $z$ and show that $z \in Z$. By the definition of $\cup$, we can conclude that $z \in X$ or $z \in Z \setminus X$ from $z \in X \cup (Z \setminus X)$. You should now be getting the hang of this.

$X \cup (Z \setminus X) = Z$ iff $X \cup (Z \setminus X) \subseteq Z$ and $Z \subseteq (X \cup (Z \setminus X)$. First we prove that $X \cup (Z \setminus X) \subseteq Z$. Let $z \in X \cup (Z \setminus X)$. So, either $z \in X$ or $z \in (Z \setminus X)$.

> We've arrived at a disjunction, and from it we want to prove that $z \in Z$. We do this using proof by cases.

Case 1: $z \in X$. Since for all $z$, if $z \in X$, $z \in Z$, we have that $z \in Z$.

> Here we've used the fact recorded earlier which followed from the hypothesis of the proposition that $X \subseteq Z$. The first case is complete, and we turn to the second case, $z \in (Z \setminus X)$. Recall that $Z \setminus X$ denotes the *difference* of the two sets, i.e., the set of all elements of $Z$ which are not elements of $X$. But any element of $Z$ not in $X$ is in particular an element of $Z$.

Case 2: $z \in (Z \setminus X)$. This means that $z \in Z$ and $z \notin X$. So, in particular, $z \in Z$.

> Great, we've proved the first direction. Now for the second direction. Here we prove that $Z \subseteq X \cup (Z \setminus X)$. So we assume that $z \in Z$ and prove that $z \in X \cup (Z \setminus X)$.

Now let $z \in Z$. We want to show that $z \in X$ or $z \in Z \setminus X$.

> Since all elements of $X$ are also elements of $Z$, and $Z \setminus X$ is the set of all things that are elements of $Z$ but not $X$, it follows that $z$ is either in $X$ or in $Z \setminus X$. This may be a bit unclear if you don't already know why the result is true. It would be better to prove it step-by-step. It will help to use a simple fact which we can state without proof: $z \in X$ or $z \notin X$. This is called the "principle of excluded middle:" for any statement $p$, either $p$ is true or its negation is true. (Here, $p$ is the statement that $z \in X$.) Since this is a disjunction, we can again use proof-by-cases.

Either $z \in X$ or $z \notin X$. In the former case, $z \in X \cup (Z \setminus X)$. In the latter case, $z \in Z$ and $z \notin X$, so $z \in Z \setminus X$. But then $z \in X \cup (Z \setminus X)$.

> Our proof is complete: we have shown that $X \cup (Z \setminus X) = Z$. □

## 50.7 Proof by Contradiction

In the first instance, proof by contradiction is an inference pattern that is used to prove negative claims. Suppose you want to show that some claim $p$ is *false*, i.e., you want to show $\neg p$. The most promising strategy is to (a) suppose that $p$ is true, and (b) show that this assumption leads to something you know to be false. "Something known to be false" may be a result that conflicts with—contradicts—$p$ itself, or some other hypothesis of the overall claim you are considering. For instance, a proof of "if $q$ then $\neg p$" involves assuming that

$q$ is true and proving $\neg p$ from it. If you prove $\neg p$ by contradiction, that means assuming $p$ in addition to $q$. If you can prove $\neg q$ from $p$, you have shown that the assumption $p$ leads to something that contradicts your other assumption $q$, since $q$ and $\neg q$ cannot both be true. Of course, you have to use other inference patterns in your proof of the contradiction, as well as unpacking definitions. Let's consider an example.

**Proposition 50.10.** *If $X \subseteq Y$ and $Y = \emptyset$, then $X$ has no elements.*

*Proof.* Suppose $X \subseteq Y$ and $Y = \emptyset$. We want to show that $X$ has no elements.

> Since this is a conditional claim, we assume the antecedent and want to prove the consequent. The consequent is: $X$ has no elements. We can make that a bit more explicit: it's not the case that there is an $x \in X$.

$X$ has no elements iff it's not the case that there is an $x$ such that $x \in X$.

> So we've determined that what we want to prove is really a negative claim $\neg p$, namely: it's not the case that there is an $x \in X$. To use proof by contradiction, we have to assume the corresponding positive claim $p$, i.e., there is an $x \in X$, and prove a contradiction from it. We indicate that we're doing a proof by contradiction by writing "by way of contradiction, assume" or even just "suppose not," and then state the assumption $p$.

Suppose not: there is an $x \in X$.

> This is now the new assumption we'll use to obtain a contradiction. We have two more assumptions: that $X \subseteq Y$ and that $Y = \emptyset$. The first gives us that $x \in Y$:

Since $X \subseteq Y$, $x \in Y$.

> But since $Y = \emptyset$, every element of $Y$ (e.g., $x$) must also be an element of $\emptyset$.

Since $Y = \emptyset$, $x \in \emptyset$. This is a contradiction, since by definition $\emptyset$ has no elements.

> This already completes the proof: we've arrived at what we need (a contradiction) from the assumptions we've set up, and this means that the assumptions can't all be true. Since the first two assumptions ($X \subseteq Y$ and $Y = \emptyset$) are not contested, it must be the last assumption introduced (there is an $x \in X$) that must be false. But if we want to be thorough, we can spell this out.

Thus, our assumption that there is an $x \in X$ must be false, hence, $X$ has no elements by proof by contradiction. □

Every positive claim is trivially equivalent to a negative claim: $p$ iff $\neg\neg p$. So proofs by contradiction can also be used to establish positive claims "indirectly," as follows: To prove $p$, read it as the negative claim $\neg\neg p$. If we can prove a contradiction from $\neg p$, we've established $\neg\neg p$ by proof by contradiction, and hence $p$.

In the last example, we aimed to prove a negative claim, namely that $X$ has no elements, and so the assumption we made for the purpose of proof by contradiction (i.e., that there is an $x \in X$) was a positive claim. It gave us something to work with, namely the hypothetical $x \in X$ about which we continued to reason until we got to $x \in \emptyset$.

When proving a positive claim indirectly, the assumption you'd make for the purpose of proof by contradiction would be negative. But very often you can easily reformulate a positive claim as a negative claim, and a negative claim as a positive claim. Our previous proof would have been essentially the same had we proved "$X = \emptyset$" instead of the negative consequent "$X$ has no elements." (By definition of $=$, "$X = \emptyset$" is a general claim, since it unpacks to "every element of $X$ is an element of $\emptyset$ and vice versa".) But it is easily seen to be equivalent to the negative claim "not: there is an $x \in X$."

So it is sometimes easier to work with $\neg p$ as an assumption than it is to prove $p$ directly. Even when a direct proof is just as simple or even simpler (as in the next example), some people prefer to proceed indirectly. If the double negation confuses you, think of a proof by contradiction of some claim as a proof of a contradiction from the *opposite* claim. So, a proof by contradiction of $\neg p$ is a proof of a contradiction from the assumption $p$; and proof by contradiction of $p$ is a proof of a contradiction from $\neg p$.

**Proposition 50.11.** $X \subseteq X \cup Y$.

*Proof.* We want to show that $X \subseteq X \cup Y$.

> On the face of it, this is a positive claim: every $x \in X$ is also in $X \cup Y$. The negation of that is: some $x \in X$ is $\notin X \cup Y$. So we can prove the claim indirectly by assuming this negated claim, and showing that it leads to a contradiction.

Suppose not, i.e., $X \not\subseteq X \cup Y$.

> We have a definition of $X \subseteq X \cup Y$: every $x \in X$ is also $\in X \cup Y$. To understand what $X \not\subseteq X \cup Y$ means, we have to use some elementary logical manipulation on the unpacked definition: it's false that every $x \in X$ is also $\in X \cup Y$ iff there is *some* $x \in X$ that is $\notin Z$. (This is a place where you want to be very careful: many students' attempted proofs by contradiction fail because they analyze the negation of a claim like "all $A$s are $B$s" incorrectly.) In other words, $X \not\subseteq X \cup Y$ iff there is an $x$ such that $x \in X$ and $x \notin X \cup Y$. From then on, it's easy.

So, there is an $x \in X$ such that $x \notin X \cup Y$. By definition of $\cup$, $x \in X \cup Y$ iff $x \in X$ or $x \in Y$. Since $x \in X$, we have $x \in X \cup Y$. This contradicts the assumption that $x \notin X \cup Y$. □

**Proposition 50.12.** *If $X \subseteq Y$ and $Y \subseteq Z$ then $X \subseteq Z$.*

*Proof.* Suppose $X \subseteq Y$ and $Y \subseteq Z$. We want to show $X \subseteq Z$.

> Let's proceed indirectly: we assume the negation of what we want to etablish.

Suppose not, i.e., $X \nsubseteq Z$.

> As before, we reason that $X \nsubseteq Z$ iff not every $x \in X$ is also $\in Z$, i.e., some $x \in X$ is $\notin Z$. Don't worry, with practice you won't have to think hard anymore to unpack negations like this.

In other words, there is an $x$ such that $x \in X$ and $x \notin Z$.

> Now we can use this to get to our contradiction. Of course, we'll have to use the other two assumptions to do it.

Since $X \subseteq Y$, $x \in Y$. Since $Y \subseteq Z$, $x \in Z$. But this contradicts $x \notin Z$. □

**Proposition 50.13.** *If $X \cup Y = X \cap Y$ then $X = Y$.*

*Proof.* Suppose $X \cup Y = X \cap Y$. We want to show that $X = Y$.

> The beginning is now routine:

Assume, by way of contradiction, that $X \neq Y$.

> Our assumption for the proof by contradiction is that $X \neq Y$. Since $X = Y$ iff $X \subseteq Y$ an $Y \subseteq X$, we get that $X \neq Y$ iff $X \nsubseteq Y$ *or* $Y \nsubseteq X$. (Note how important it is to be careful when manipulating negations!) To prove a contradiction from this disjunction, we use a proof by cases and show that in each case, a contradiction follows.

$X \neq Y$ iff $X \nsubseteq Y$ or $Y \nsubseteq X$. We distinguish cases.

> In the first case, we assume $X \nsubseteq Y$, i.e., for some $x$, $x \in X$ but $\notin Y$. $X \cap Y$ is defined as those elements that $X$ and $Y$ have in common, so if something isn't in one of them, it's not in the intersection. $X \cup Y$ is $X$ together with $Y$, so anything in either is also in the union. This tells us that $x \in X \cup Y$ but $x \notin X \cap Y$, and hence that $X \cap Y \neq Y \cap X$.

Case 1: $X \not\subseteq Y$. Then for some $x$, $x \in X$ but $x \notin Y$. Since $x \notin Y$, then $x \notin X \cap Y$. Since $x \in X$, $x \in X \cup Y$. So, $X \cap Y \neq Y \cap X$, contradicting the assumption that $X \cap Y = X \cup Y$.

Case 2: $Y \not\subseteq X$. Then for some $y$, $y \in Y$ but $y \notin X$. As before, we have $y \in X \cup Y$ but $y \notin X \cap Y$, and so $X \cap Y \neq X \cup Y$, again contradicting $X \cap Y = X \cup Y$. □

## 50.8 Reading Proofs

Proofs you find in textbooks and articles very seldom give all the details we have so far included in our examples. Authors ofen do not draw attention to when they distinguish cases, when they give an indirect proof, or don't mention that they use a definition. So when you read a proof in a textbook, you will often have to fill in those details for yourself in order to understand the proof. Doing this is also good practice to get the hang of the various moves you have to make in a proof. Let's look at an example.

**Proposition 50.14** (Absorption)**.** *For all sets $X$, $Y$,*

$$X \cap (X \cup Y) = X$$

*Proof.* If $z \in X \cap (X \cup Y)$, then $z \in X$, so $X \cap (X \cup Y) \subseteq X$. Now suppose $z \in X$. Then also $z \in X \cup Y$, and therefore also $z \in X \cap (X \cup Y)$. □

The preceding proof of the absorption law is very condensed. There is no mention of any definitions used, no "we have to prove that" before we prove it, etc. Let's unpack it. The proposition proved is a general claim about any sets $X$ and $Y$, and when the proof mentions $X$ or $Y$, these are variables for arbitrary sets. The general claims the proof establishes is what's required to prove identity of sets, i.e., that every element of the left side of the identity is an element of the right and vice versa.

> "If $z \in X \cap (X \cup Y)$, then $z \in X$, so $X \cap (X \cup Y) \subseteq X$."

This is the first half of the proof of the identity: it estabishes that if an arbitrary $z$ is an element of the left side, it is also an element of the right, i.e., $X \cap (X \cup Y) \subseteq X$. Assume that $z \in X \cap (X \cup Y)$. Since $z$ is an element of the intersection of two sets iff it is an element of both sets, we can conclude that $z \in X$ and also $z \in X \cup Y$. In particular, $z \in X$, which is what we wanted to show. Since that's all that has to be done for the first half, we know that the rest of the proof must be a proof of the second half, i.e., a proof that $X \subseteq X \cap (X \cup Y)$.

> "Now suppose $z \in X$. Then also $z \in X \cup Y$, and therefore also $z \in X \cap (X \cup Y)$."

We start by assuming that $z \in X$, since we are showing that, for any $z$, if $z \in X$ then $z \in X \cap (X \cup Y)$. To show that $z \in X \cap (X \cup Y)$, we have to show (by definition of "$\cap$") that (i) $z \in X$ and also (ii) $z \in X \cup Y$. Here (i) is just our assumption, so there is nothing further to prove, and that's why the proof does not mention it again. For (ii), recall that $z$ is an element of a union of sets iff it is an element of at least one of those sets. Since $z \in X$, and $X \cup Y$ is the union of $X$ and $Y$, this is the case here. So $z \in X \cup Y$. We've shown both (i) $z \in X$ and (ii) $z \in X \cup Y$, hence, by definition of "$\cap$," $z \in X \cap (X \cup Y)$. The proof doesn't mention those definitions; it's assumed the reader has already internalized them. If you haven't, you'll have to go back and remind yourself what they are. Then you'll also have to recognize why it follows from $z \in X$ that $z \in X \cup Y$, and from $z \in X$ and $z \in X \cup Y$ that $z \in X \cap (X \cup Y)$.

Here's another version of the proof above, with everything made explicit:

*Proof.* [By definition of $=$ for sets, $X \cap (X \cup Y) = X$ we have to show (a) $X \cap (X \cup Y) \subseteq X$ and (b) $X \cap (X \cup Y) \subseteq X$. (a): By definition of $\subseteq$, we have to show that if $z \in X \cap (X \cup Y)$, then $z \in X$.] If $z \in X \cap (X \cup Y)$, then $z \in X$ [since by definition of $\cap$, $z \in X \cap (X \cup Y)$ iff $z \in X$ and $z \in X \cup Y$], so $X \cap (X \cup Y) \subseteq X$. [(b): By definition of $\subseteq$, we have to show that if $z \in X$, then $z \in X \cap (X \cup Y)$.] Now suppose [(1)] $z \in X$. Then also [(2)] $z \in X \cup Y$ [since by (1) $z \in X$ or $z \in Y$, which by definition of $\cup$ means $z \in X \cup Y$], and therefore also $z \in X \cap (X \cup Y)$ [since the definition of $\cap$ requires that $z \in X$, i.e., (1), and $z \in X \cup Y$), i.e., (2)]. □

## 50.9 I Can't Do It!

We all get to a point where we feel like giving up. But you *can* do it. Your instructor and teaching assistant, as well as your fellow students, can help. Ask them for help! Here are a few tips to help you avoid a crisis, and what to do if you feel like giving up.

To make sure you can solve problems successfully, do the following:

1. *Start as far in advance as possible.* We get busy throughout the semester and many of us struggle with procrastination, one of the best things you can do is to start your homework assignments early. That way, if you're stuck, you have time to look for a solution (that isn't crying).

2. *Talk to your classmates*. You are not alone. Others in the class may also struggle—but the may struggle with different things. Talking it out with your peers can give you a different perspective on the problem that might lead to a breakthrough. Of course, don't just copy their solution: ask them for a hint, or explain where you get stuck and ask them for the next step. And when you do get it, reciprocate. Helping someone else along, and explaining things will help you understand better, too.

3. *Ask for help.* You have many resources available to you—your instructor and teaching assistant are there for you and *want* you to succeed. They should be able to help you work out a problem and identify where in the process you're struggling.

4. *Take a break.* If you're stuck, it *might* be because you've been staring at the problem for too long. Take a short break, have a cup of tea, or work on a different problem for a while, then return to the problem with a fresh mind. Sleep on it.

Notice how these strategies require that you've started to work on the proof well in advance? If you've started the proof at 2am the day before it's due, these might not be so helpful.

This might sound like doom and gloom, but solving a proof is a challenge that pays off in the end. Some people do this as a career—so there must be something to enjoy about it. Like basically everything, solving problems and doing proofs is something that requires practice. You might see classmates who find this easy: they've probably just had lots of practice already. Try not to give in too easily.

If you do run out of time (or patience) on a particular problem: that's ok. It doesn't mean you're stupid or that you will never get it. Find out (from your instructor or another student) how it is done, and identify where you went wrong or got stuck, so you can avoid doing that the next time you encounter a similar issue. Then try to do it without looking at the solution. And next time, start (and ask for help) earlier.

## 50.10 Other Resources

There are many books on how to do proofs in mathematics which may be useful. Check out *How to Read and do Proofs: An Introduction to Mathematical Thought Processes* by Daniel Solow and *How to Prove It: A Structured Approach* by Daniel Velleman in particular. The *Book of Proof* by Richard Hammack and *Mathematical Reasoning* by Ted Sundstrom are books on proof that are freely available. Philosophers might find *More Precisely: The Math you need to do Philosophy* by Eric Steinhart to be a good primer on mathematical reasoning.

There are also various shorter guides to proofs available on the internet; e.g., "Introduction to Mathematical Arguments" by Michael Hutchings and "How to write proofs" by Eugenia Chang.

### Motivational Videos

Feel like you have no motivation to do your homework? Feeling down? These videos might help!

- `https://www.youtube.com/watch?v=ZXsQAXx_ao0`

- https://www.youtube.com/watch?v=BQ4yd2W50No
- https://www.youtube.com/watch?v=StTqXEQ2l-Y

## Problems

**Problem 50.1.** Suppose you are asked to prove that $X \cap Y \neq \emptyset$. Unpack all the definitions occuring here, i.e., restate this in a way that does not mention "$\cap$", "=", or "$\emptyset$.

**Problem 50.2.** Prove *indirectly* that $X \cap Y \subseteq X$.

**Problem 50.3.** Expand the following proof of $X \cup (X \cap Y) = X$, where you mention all the inference patterns used, why each step follows from assumptions or claims established before it, and where we have to appeal to which definitions.

*Proof.* If $z \in X \cup (X \cap Y)$ then $z \in X$ or $z \in X \cap Y$. If $z \in X \cap Y$, $z \in X$. Any $z \in X$ is also $\in X \cup (X \cap Y)$. □

# Chapter 51

# Induction

## 51.1 Introduction

Induction is an important proof technique which is used, in different forms, in almost all areas of logic, theoretical computer science, and mathematics. It is needed to prove many of the results in logic.

Induction is often contrasted with deduction, and characterized as the inference from the particular to the general. For instance, if we observe many green emeralds, and nothing that we would call an emerald that's not green, we might conclude that all emeralds are green. This is an inductive inference, in that it proceeds from many particlar cases (this emerald is green, that emerald is green, etc.) to a general claim (all emeralds are green). *Mathematical* induction is also an inference that concludes a general claim, but it is of a very different kind that this "simple induction."

Very roughly, an inductive proof in mathematics concludes that all mathematical objects of a certain sort have a certain property. In the simplest case, the mathematical objects an inductive proof is concerned with are natural numbers. In that case an inductive proof is used to establish that all natural numbers have some property, and it does this by showing that (1) 0 has the property, and (2) whenever a number $n$ has the property, so does $n + 1$. Induction on natural numbers can then also often be used to prove general about mathematical objects that can be assigned numbers. For instance, finite sets each have a finite number $n$ of elements, and if we can use induction to show that every number $n$ has the property "all finite sets of size $n$ are ..." then we will have shown something about all finite sets.

Induction can also be generalized to mathematical objects that are *inductively defined*. For instance, expressions of a formal language suchh as those of first-order logic are defined inductively. *Structural induction* is a way to prove results about all such expressions. Structural induction, in particular, is very useful—and widely used—in logic.

## 51.2 Induction on $\mathbb{N}$

In its simplest form, induction is a technique used to prove results for all natural numbers. It uses the fact that by starting from 0 and repeatedly adding 1 we eventually reach every natural number. So to prove that something is true for every number, we can (1) establish that it is true for 0 and (2) show that whenever it is true for a number $n$, it is also true for the next number $n+1$. If we abbreviate "number $n$ has property $P$" by $P(n)$, then a proof by induction that $P(n)$ for all $n \in \mathbb{N}$ consists of:

1. a proof of $P(0)$, and

2. a proof that, for any $n$, if $P(n)$ then $P(n+1)$.

To make this crystal clear, suppose we have both (1) and (2). Then (1) tells us that $P(0)$ is true. If we also have (2), we know in particular that if $P(0)$ then $P(0+1)$, i.e., $P(1)$. (This follows from the general statement "for any $n$, if $P(n)$ then $P(n+1)$" by putting 0 for $n$. So by modus ponens, we have that $P(1)$. From (2) again, now taking 1 for $n$, we have: if $P(1)$ then $P(2)$. Since we've just established $P(1)$, by modus ponens, we have $P(2)$. And so on. For any number $k$, after doing this $k$ steps, we eventually arrive at $P(k)$. So (1) and (2) together establish $P(k)$ for any $k \in \mathbb{N}$.

Let's look at an example. Suppose we want to find out how many different sums we can throw with $n$ dice. Although it might seem silly, let's start with 0 dice. If you have no dice there's only one possible sum you can "throw": no dots at all, which sums to 0. So the number of different possible throws is 1. If you have only one die, i.e., $n = 1$, there are six possible values, 1 through 6. With two dice, we can throw any sum from 2 through 12, that's 11 possibilities. With three dice, we can throw any number from 3 to 18, i.e., 16 different possibilities. 1, 6, 11, 16: looks like a pattern: maybe the answer is $5n+1$? Of course, $5n+1$ is the maximum possible, because there are only $5n+1$ numbers between $n$, the lowest value you can throw with $n$ dice (all 1's) and $6n$, the highest you can throw (all 6's).

**Theorem 51.1.** *With $n$ dice one can throw all $5n+1$ possible values between $n$ and $6n$.*

*Proof.* Let $P(n)$ be the claim: "It is possible to throw any number between $n$ and $6n$ using $n$ dice." To use induction, we prove:

1. The *induction basis* $P(1)$, i.e., with just one die, you can throw any number between 1 and 6.

2. The *induction step*, for all $k$, if $P(k)$ then $P(k+1)$.

(1) Is proved by inspecting a 6-sided die. It has all 6 sides, and every number between 1 and 6 shows up one on of the sides. So it is possible to throw any number between 1 and 6 using a single die.

To prove (2), we assume the antecedent of the conditional, i.e., $P(k)$. This assumption is called the *inductive hypothesis*. We use it to prove $P(k+1)$. The hard part is to find a way of thinking about the possible values of a throw of $k+1$ dice in terms of the possible values of throws of $k$ dice plus of throws of the extra $k+1$-st die—this is what we have to do, though, if we want to use the inductive hypothesis.

The inductive hypothesis says we can get any number between $k$ and $6k$ using $k$ dice. If we throw a 1 with our $(k+1)$-st die, this adds 1 to the total. So we can throw any value between $k+1$ and $6k+1$ by throwing 5 dice and then rolling a 1 with the $(k+1)$-st die. What's left? The values $6k+2$ through $6k+6$. We can get these by rolling $k$ 6s and then a number between 2 and 6 with our $(k+1)$-st die. Together, this means that with $k+1$ dice we can throw any of the numbers between $k+1$ and $6(k+1)$, i.e., we've proved $P(k+1)$ using the assumption $P(k)$, the inductive hypothesis. □

Very often we use induction when we want to prove something about a series of objects (numbers, sets, etc.) that is itself defined "inductively," i.e., by defining the $(n+1)$-st object in terms of the $n$-th. For instance, we can define the sum $s_n$ of the natural numbers up to $n$ by

$$\begin{aligned} s_0 &= 0 \\ s_{n+1} &= s_n + (n+1) \end{aligned}$$

This definition gives:

$$\begin{aligned} s_0 &= 0, && \\ s_1 &= s_0 + 1 && = 1, \\ s_2 &= s_1 + 2 && = 1 + 2 = 3 \\ s_3 &= s_2 + 3 && = 1 + 2 + 3 = 6, \text{ etc.} \end{aligned}$$

Now we can prove, by induction, that $s_n = n(n+1)/2$.

**Proposition 51.2.** $s_n = n(n+1)/2$.

*Proof.* We have to prove (1) that $s_0 = 0 \cdot (0+1)/2$ and (2) if $s_n = n(n+1)/2$ then $s_{n+1} = (n+1)(n+2)/2$. (1) is obvious. To prove (2), we assume the inductive hypothesis: $s_n = n(n+1)/2$. Using it, we have to show that $s_{n+1} = (n+1)(n+2)/2$.

What is $s_{n+1}$? By the definition, $s_{n+1} = s_n + (n+1)$. By inductive hypothesis, $s_n = n(n+1)/2$. We can substitute this into the previous equation, and

then just need a bit of arithmetic of fractions:

$$\begin{aligned} s_{n+1} &= \frac{n(n+1)}{2} + (n+1) = \\ &= \frac{n(n+1)}{2} + \frac{2(n+1)}{2} = \\ &= \frac{n(n+1) + 2(n+1)}{2} = \\ &= \frac{(n+2)(n+1)}{2}. \end{aligned}$$

□

The important lesson here is that if you're proving something about some inductively defined sequence $a_n$, induction is the obvious way to go. And even if it isn't (as in the case of the possibilities of dice throws), you can use induction if you can somehow relate the case for $n+1$ to the case for $n$.

## 51.3 Strong Induction

In the principle of induction discussed above, we prove $P(0)$ and also if $P(n)$, then $P(n+1)$. In the second part, we assume that $P(n)$ is true and use this assumption to prove $P(n+1)$. Equivalently, of course, we could assume $P(n-1)$ and use it to prove $P(n)$—the important part is that we be able to carry out the inference from any number to its successor; that we can prove the claim in question for any number under the assumption it holds for its predecessor.

There is a variant of the principle of induction in which we don't just assume that the claim holds for the predecessor $n-1$ of $n$, but for all numbers smaller than $n$, and use this assumption to establish the claim for $n$. This also gives us the claim $P(k)$ for all $k \in \mathbb{N}$. For once we have established $P(0)$, we have thereby established that $P$ holds for all numbers less than 1. And if we know that if $P(l)$ for all $l < n$ then $P(n)$, we know this in particular for $n = 1$. So we can conclude $P(2)$. With this we have proved $P(0)$, $P(1)$, $P(2)$, i.e., $P(l)$ for all $l < 3$, and since we have also the conditional, if $P(l)$ for all $l < 3$, then $P(3)$, we can conclude $P(3)$, and so on.

In fact, if we can establish the general conditional "for all $n$, if $P(l)$ for all $l < n$, then $P(n)$," we do not have to establish $P(0)$ anymore, since it follows from it. For remember that a general claim like "for all $l < n$, $P(l)$" is true if there are no $l < n$. This is a case of vacuous quantification: "all $A$s are $B$s" is true if there are no $A$s, $\forall x\,(\varphi(x) \to \psi(x))$ is true if no $x$ satisfies $\varphi(x)$. In this case, the formalized version would be "$\forall l\,(l < n \to P(l))$"—and that is true if there are no $l < n$. And if $n = 0$ that's exactly the case: no $l < 0$, hence "for all $l < 0$, $P(0)$" is true, whatever $P$ is. A proof of "if $P(l)$ for all $l < n$, then $P(n)$" thus automatically establishes $P(0)$.

This variant is useful if establishing the claim for $n$ can't be made to just rely on the claim for $n-1$ but may require the assumption that it is true for one or more $l < n$.

## 51.4 Inductive Definitions

In logic we very often define kinds of objects *inductively*, i.e., by specifying rules for what counts as an object of the kind to be defined which explain how to get new objects of that kind from old objects of that kind. For instance, we often define special kinds of sequences of symbols, such as the terms and formulas of a language, by induction. For a simple example, consider strings of consisting of letters a, b, c, d, the symbol $\circ$, and brackets [ and ], such as "[[c $\circ$ d][", "[a[]$\circ$]", "a" or "[[a $\circ$ b] $\circ$ d]". You probably feel that there's something "wrong" with the first two strings: the brackets don't "balance" at all in the first, and you might feel that the "$\circ$" should "connect" expressions that themselves make sense. The third and fourth string look better: for every "[" there's a closing "]" (if there are any at all), and for any $\circ$ we can find "nice" expressions on either side, surrounded by a pair of parenteses.

We would like to precisely specify what counts as a "nice term." First of all, every letter by itself is nice. Anything that's not just a letter by itself should be of the form "$[t \circ s]$" where $s$ and $t$ are themselves nice. Conversely, if $t$ and $s$ are nice, then we can form a new nice term by putting a $\circ$ between them and surround them by a pair of brackets. We might use these operations to *define* the set of nice terms. This is an *inductive definition*.

**Definition 51.3** (Nice terms)**.** The set of *nice terms* is inductively defined as follows:

1. Any letter a, b, c, d is a nice term.
2. If $s$ and $s'$ are nice terms, then so is $[s \circ s']$.
3. Nothing else is a nice term.

This definition tells us that something counts as a nice term iff it can be constructed according to the two conditions (1) and (2) in some finite number of steps. In the first step, we construct all nice terms just consisting of letters by themselves, i.e.,

$$\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}$$

In the second step, we apply (2) to the terms we've constructed. We'll get

$$[\mathrm{a} \circ \mathrm{a}], [\mathrm{a} \circ \mathrm{b}], [\mathrm{b} \circ \mathrm{a}], \ldots, [\mathrm{d} \circ \mathrm{d}]$$

for all combinations of two letters. In the third step, we apply (2) again, to any two nice terms we've constructed so far. We get new nice term such as [a $\circ$ [a $\circ$

a]]—where $t$ is a from step 1 and $s$ is [a ∘ a] from step 2—and [[b ∘ c] ∘ [d ∘ b]] constructed out of the two terms [b ∘ c] and [d ∘ b] from step 2. And so on. Clause (3) rules out that anything not constructed in this way sneaks into the set of nice terms.

Note that we have not yet proved that every sequence of symbols that "feels" nice is nice according to this definition. However, it should be clear that everything we can construct does in fact "feel nice:" brackets are balanced, and ∘ connects parts that are themselves nice.

The key feature of inductive definitions is that if you want to prove something about all nice terms, the definition tells you which cases you must consider. For instance, if you are told that $t$ is a nice term, the inductive definition tells you what $t$ can look like: $t$ can be a letter, or it can be $[r \circ s]$ for some other pair of nice terms $r$ and $s$. Because of clause (3), those are the only possibilities.

When proving claims about all of an inductively defined set, the strong form of induction becomes particularly important. For instance, suppose we want to prove that for every nice term of length $n$, the number of [ in it is $< n/2$. This can be seen as a claim about all $n$: for every $n$, the number of [ in any nice term of length $n$ is $< n/2$.

**Proposition 51.4.** *For any $n$, the number of [ in a nice term of length $n$ is $< n/2$.*

*Proof.* To prove this result by (strong) induction, we have to show that the following conditional claim is true:

> If for every $k < n$, any parexpression of length $k$ has $k/2$ ['s, then any parexpression of length $n$ has $n/2$ ['s.

To show this conditional, assume that its antecedent is true, i.e., assume that for any $k < n$, parexpressions of length $k$ contain $< k/2$ ['s. We call this assumption the inductive hypothesis. We want to show the same is true for parexpressions of length $n$.

So suppose $t$ is a nice term of length $n$. Because parexpressions are inductively defined, we have three two cases: (1) $t$ is a letter by itself, or $t$ is $[r \circ s]$ for some nice terms $r$ and $s$.

1. $t$ is a letter. Then $n = 1$, and the number of [ in $t$ is 0. Since $0 < 1/2$, the claim holds.

2. $t$ is $[s \circ s']$ for some nice terms $s$ and $s'$. Let's let $k$ be the length of $s$ and $k'$ be the length of $s'$. Then the length $n$ of $t$ is $k + k' + 3$ (the lengths of $s$ and $s'$ plus three symbols [, ∘, ]). Since $k + k' + 3$ is always greater than $k$, $k < n$. Similarly, $k' < n$. That means that the induction hypothesis applies to the terms $s$ and $s'$: the number $m$ of [ in $s$ is $< k/2$, and the number of [ in $s'$ is $< k'/2$.

The number of [ in $t$ is the number of [ in $s$, plus the number of [ in $s'$, plus 1, i.e., it is $m + m' + 1$. Since $m < k/2$ and $m' < k'/2$ we have:

$$m + m' + 1 < \frac{k}{2} + \frac{k'}{2} + 1 = \frac{k + k' + 2}{2} < \frac{k + k' + 3}{2} = n/2.$$

In each case, we've shown that the number of [ in $t$ is $< n/2$ (on the basis of the inductive hypothesis). By strong induction, the proposition follows. □

## 51.5 Structural Induction

So far we have used induction to establish results about all natural numbers. But a corresponding principle can be used directly to prove results about all elements of an inductively defined set. This often called *structural* induction, because it depends on the structure of the inductively defined objects.

Generally, an inductive definition is given by (a) a list of "initial" elements of the set and (b) a list of operations which produce new elements of the set from old ones. In the case of nice terms, for instance, the initial objects are the letters. We only have one operation: the operations are

$$o(s, s') = [s \circ s']$$

You can even think of the natural numbers $\mathbb{N}$ themselves as being given be an inductive definition: the initial object is 0, and the operation is the successor function $x + 1$.

In order to prove something about all elements of an inductively defined set, i.e., that every element of the set has a property $P$, we must:

1. Prove that the initial objects have $P$
2. Prove that for each operation $o$, if the arguments have $P$, so does the result.

For instance, in order to prove something about all nice terms, we would prove that it is true about all letters, and that it is true about $[s \circ s']$ provided it is true of $s$ and $s'$ individually.

**Proposition 51.5.** *The number of [ equals the number of ] in any nice term $t$.*

*Proof.* We use structural induction. Nice terms are inductively defined, with letters as initial objects and the operations $o$ for constructing new nice terms out of old ones.

1. The claim is true for every letter, since the number of [ in a letter by itself is 0 and the number of ] in it is also 0.

2. Suppose the number of [ in $s$ equals the number of ], and the same is true for $s'$. The number of [ in $o(s, s')$, i.e., in $[s \circ s']$, is the sum of the number of [ in $s$ and $s'$. The number of ] in $o(s, s')$ is the sum of the number of ] in $s$ and $s'$. Thus, the number of [ in $o(s, s')$ equals the number of ] in $o(s, s')$.

□

Let's give another proof by structural induction: a proper initial segment of a string of symbols $t$ is any string $t'$ that agrees with $t$ symbol by symbol, read from the left, but $t'$ is longer. So, e.g., $[a \circ$ is a proper initial segment of $[a \circ b]$, but neither are $[b \circ$ (they disagree at the second symbol) nor $[a \circ b]$ (they are the same length).

**Proposition 51.6.** *Every proper initial segment of a nice term $t$ has more [ 's than ] 's.*

*Proof.* By induction on $t$:

1. $t$ is a letter by itself: Then $t$ has no proper initial segments.

2. $t = [s \circ s']$ for some nice terms $s$ and $s'$. If $r$ is a proper initial segment of $t$, there are a number of possibilities:

   a) $r$ is just [: Then $r$ has one more [ than it does ].

   b) $r$ is $[r'$ where $r'$ is a proper initial segment of $s$: Since $s$ is a nice term, by induction hypothesis, $r'$ has more [ than ] and the same is true for $[r'$.

   c) $r$ is $[s$ or $[s \circ$: By the previous result, the number of [ and ] in $s$ is equal; so the number of [ in $[s$ or $[s \circ$ is one more than the number of ].

   d) $r$ is $[s \circ r'$ where $r'$ is a proper initial segment of $s'$: By induction hypothesis, $r'$ contains more [ than ]. By the previous result, the number of [ and of ] in $s$ is equal. So the number of [ in $[s \circ r'$ is greater than the number of ].

   e) $r$ is $[s \circ s'$: By the previous result, the number of [ and ] in $s$ is equal, and the same for $s'$. So there is one more [ in $[s \circ s'$ than there are ].

□

## 51.6 Relations and Functions

When we have defined a set of objects (such as the natural numbers or the nice terms) inductively, we can also define *relations on* these objects by induction. For instance, consider the following idea: a nice term $t$ is a subterm of a nice term $t'$ if it occurs as a part of it. Let's use a symbol for it: $t \sqsubseteq t'$. Every nice

term is a subterm of itself, of course: $t \sqsubseteq t$. We can give an inductive definition of this relation as follows:

**Definition 51.7.** The relation of a nice term $t$ being a subterm of $t'$, $t \sqsubseteq t'$, is defined by induction on $s'$ as follows:

1. If $t'$ is a letter, then $t \sqsubseteq t'$ iff $t = t'$.

2. If $t'$ is $[s \circ s']$, then $t \sqsubseteq t'$ iff $t = t'$, $t \sqsubseteq s$, or $t \sqsubseteq s'$.

This definition, for instance, will tell us that $\mathrm{a} \sqsubseteq [\mathrm{b} \circ \mathrm{a}]$. For (2) says that $\mathrm{a} \sqsubseteq [\mathrm{b} \circ \mathrm{a}]$ iff $\mathrm{a} = [\mathrm{b} \circ \mathrm{a}]$, or $\mathrm{a} \sqsubseteq b$, or $\mathrm{a} \sqsubseteq \mathrm{a}$. The first two are false: a clearly isn't identical to $[\mathrm{b} \circ \mathrm{a}]$, and by (1), $\mathrm{a} \sqsubseteq \mathrm{b}$ iff $\mathrm{a} = \mathrm{b}$, which is also false. However, also by (1), $\mathrm{a} \sqsubseteq \mathrm{a}$ iff $\mathrm{a} = \mathrm{a}$, which is true.

It's important to note that the success of this definition depends on a fact that we haven't proved yet: every nice term $t$ is either a letter by itself, or there are uniquely determined nice terms $s$ and $s'$ such that $t = [s \circ s']$. "Uniquely determined" here means that if $t = [s \circ s']$ it isn't *also* $= [r \circ r']$ with $s \neq r$ or $s' \neq r'$. If this were the case, then clause (2) may come in conflict with itself: reading $t'$ as $[s \circ s']$ we might get $t \sqsubseteq t'$, but if we read $t'$ as $[r \circ r']$ we might get not $t \sqsubseteq t'$. Before we prove that this can't happen, let's look at an example where it *can* happen.

**Definition 51.8.** Define *bracketless terms* inductively by

1. Every letter is a bracketles term.

2. If $s$ and $s'$ are bracketless terms, then $s \circ s'$ is a bracketless term.

3. Nothing else is a bracketless term.

Bracketless terms are, e.g., $\mathrm{a}$, $\mathrm{b} \circ \mathrm{d}$, $\mathrm{b} \circ \mathrm{a} \circ \mathrm{b}$. Now if we defined "subterm" for bracketless terms the way we did above, the second clause would read

$$\text{If } t' = s \circ s', \text{ then } t \sqsubseteq t' \text{ iff } t = t', t \sqsubseteq s, \text{ or } t \sqsubseteq s'.$$

Now $\mathrm{b} \circ \mathrm{a} \circ \mathrm{b}$ is of the form $s \circ s'$ with $s = \mathrm{b}$ and $s' = \mathrm{a} \circ \mathrm{b}$. It is also of the form $r \circ r'$ with $r = \mathrm{b} \circ \mathrm{a}$ and $r' = \mathrm{b}$. Now is $\mathrm{a} \circ \mathrm{b}$ a subterm of $\mathrm{b} \circ \mathrm{a} \circ \mathrm{b}$? The answer is yes if we go by the first reading, and no if we go by the second.

The property that the way a nice term is built up from other nice terms is unique is called *unique readability*. Since inductive definitions of relations for such inductively defined objects are important, we have to prove that it holds.

**Proposition 51.9.** *Suppose $t$ is a nice term. Then either $t$ is a letter by itself, or there are uniquely determined nice terms $s$, $s'$ such that $t = [s \circ s']$.*

*Proof.* If $t$ is a letter by itself, the condition issatisfied. So assume $t$ isn't a letter by itself. We can tell from the inductive definition that then $t$ must be of the form $[s \circ s']$ for some nice terms $s$ and $s'$. It remains to show that these are uniquely determined, i.e., if $t = [r \circ r']$, then $s = r$ and $s' = r'$.

So suppose $t = [s \circ s']$ and $t = [r \circ r']$ for nice terms $s$, $s'$, $r$, $r'$. We have to show that $s = r$ and $s' = r'$. First, $s$ and $r$ must be identical, for otherwise one is a proper initial segment of the other. But by Proposition 51.6, that is impossible if $s$ and $r$ are both nice terms. But if $s = r$, then clearly also $s' = r'$. □

We can also define functions inductively: e.g., we can define the function $f$ that maps any nice term to the maximum depth of nested $[\dots]$ in it as follows:

**Definition 51.10.** The *depth* of a nice term, $f(t)$, is defined inductively as follows:

$$f(s) = 0 \text{ if } s \text{ is a letter}$$
$$f([s \circ s'] = \max(f(s), f(s')) + 1$$

For instance

$$\begin{aligned} f([\mathrm{a} \circ \mathrm{b}]) &= \max(f(\mathrm{a}), f(\mathrm{b})) + 1 = \\ &= \max(0, 0) + 1 = 1, \text{ and} \\ f([[\mathrm{a} \circ \mathrm{b}] \circ \mathrm{c}]) &= \max(f([\mathrm{a} \circ \mathrm{b}]), f(\mathrm{c})) + 1 = \\ &= \max(1, 0) + 1 = 2. \end{aligned}$$

Here, of course, we assume that $s$ an $s'$ are nice terms, and make use of the fact that every nice term is either a letter or of the form $[s \circ s']$. It is again important that it can be of this form in only one way. To see why, consider again the bracketless terms we defined earlier. The corresponding "definition" would be:

$$g(s) = 0 \text{ if } s \text{ is a letter}$$
$$g(s \circ s') = \max(g(s), g(s')) + 1$$

Now consider the bracketless term $\mathrm{a} \circ \mathrm{b} \circ \mathrm{c} \circ \mathrm{d}$. It can be read in more than one way, e.g., as $s \circ s'$ with $s = \mathrm{a}$ and $s' = \mathrm{b} \circ \mathrm{c} \circ \mathrm{d}$, or as $r \circ r'$ with $r = \mathrm{a} \circ b$ and $r' = \mathrm{c} \circ \mathrm{d}$. Calculating $g$ according to the first way of reading it would give

$$\begin{aligned} g(s \circ s') &= \max(g(\mathrm{a}), g(\mathrm{b} \circ \mathrm{c} \circ \mathrm{d})) + 1 = \\ &= \max(0, 2) + 1 = 3 \end{aligned}$$

while according to the other reading we get

$$\begin{aligned} g(r \circ r') &= \max(g(\mathrm{a} \circ \mathrm{b}), g(\mathrm{c} \circ \mathrm{d})) + 1 = \\ &= \max(1, 1) + 1 = 2 \end{aligned}$$

But a function must always yield a unique value; so our "definition" of $g$ doesn't define a function at all.

## Problems

**Problem 51.1.** Define the set of supernice terms by

1. Any letter a, b, c, d is a supernice term.
2. If $s$ is a supernice term, then so is $[s]$.
3. If $t$ and $s$ are supernice terms, then so is $[t \circ s]$.
4. Nothing else is a supernice term.

Show that the number of $[$ in a supernice term $s$ of length $n$ is $\leq n/2 + 1$.

**Problem 51.2.** Prove by structural induction that no nice term starts with $]$.

**Problem 51.3.** Give an inductive definition of the function $l$, where $l(t)$ is the number of symbols in the nice term $t$.

**Problem 51.4.** Prove by induction on nice terms $t$ that $f(t) < l(t)$ (where $l(t)$ is the number of symbols in $t$ and $f(t)$ is the depth of $t$ as defined in Definition 51.10).

# Part XIII

# History

# Chapter 52

# Biographies

## 52.1 Georg Cantor

Figure 52.1: Georg Cantor

An early biography of Georg Cantor (GAY-org KAHN-tor) claimed that he was born and found on a ship that was sailing for Saint Petersburg, Russia, and that his parents were unknown. This, however, is not true; although he was born in Saint Petersburg in 1845.

Cantor received his doctorate in mathematics at the University of Berlin in 1867. He is known for his work in set theory, and is credited with founding set theory as a distinctive research discipline. He was the first to prove that there are infinite sets of different sizes. His theories, and especially his theory of infinities, caused much debate among mathematicians at the time, and his work was controversial.

Cantor's religious beliefs and his mathematical work were inextricably tied; he even claimed that the theory of transfinite numbers had been communicated to him directly by God. In later life, Cantor suffered from mental illness. Beginning in 1984, and more frequently towards his later years, Cantor was hospitalized. The heavy criticism of his work, including a falling out with the mathematician Leopold Kronecker, led to depression and a lack of interest in mathematics. During depressive episodes, Cantor would turn to philosophy and literature, and even published a theory that Francis Bacon was the author of Shakespeare's plays.

Cantor died on January 6, 1918, in a sanatorium in Halle.

**Further Reading** For full biographies of Cantor, see Dauben (1990) and Grattan-Guinness (1971). Cantor's radical views are also described in the BBC Radio 4 program *A Brief History of Mathematics* (du Sautoy, 2014). If you'd like to hear about Cantor's theories in rap form, see Rose (2012).

## 52.2 Alonzo Church

Figure 52.2: Alonzo Church

Alonzo Church was born in Washington, DC on June 14, 1903. In early childhood, an air gun incident left Church blind in one eye. He finished preparatory school in Connecticut in 1920 and began his university education at Princeton that same year. He completed his doctoral studies in 1927. After a couple years abroad, Church returned to Princeton. Church was known exceedingly polite and careful. His blackboard writing was immaculate, and he would preserve important papers by carefully covering them in Duco cement (a clear glue). Outside of his academic pursuits, he enjoyed reading science fiction magazines and was not afraid to write to the editors if he spotted any inaccuracies in the writing.

Church's academic achievements were great. Together with his students Stephen Kleene and Barkley Rosser, he developed a theory of effective calculability, the lambda calculus, independently of Alan Turing's development of the Turing machine. The two definitions of computability are equivalent, and give rise to what is now known as the *Church-Turing Thesis*, that a function of the natural numbers is effectively computable if and only if it is computable via Turing machine (or lambda calculus). He also proved what is now known as *Church's Theorem*: The decision problem for the validity of first-order formulas is unsolvable.

Church continued his work into old age. In 1967 he left Princeton for UCLA, where he was professor until his retirement in 1990. Church passed away on August 1, 1995 at the age of 92.

**Further Reading** For a brief biography of Church, see Enderton (forthcoming). Church's original writings on the lambda calculus and the Entscheidungsproblem (Church's Thesis) are Church (1936a,b). Aspray (1984) records

an interview with Church about the Princeton mathematics community in the 1930s. Church wrote a series of book reviews of the *Journal of Symbolic Logic* from 1936 until 1979. They are all archived on John MacFarlane's website (MacFarlane, 2015).

## 52.3 Gerhard Gentzen

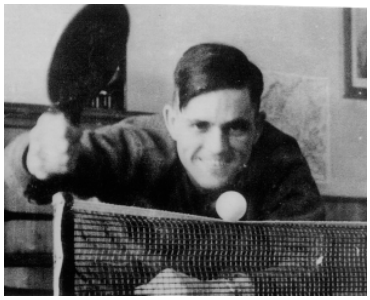


Figure 52.3: Gerhard Gentzen

Gerhard Gentzen is known primarily as the creator of structural proof theory, and specifically the creation of the natural deduction and sequent calculus proof systems. He was born on November 24, 1909 in Greifswald, Germany. Gerhard was homeschooled for three years before attending preparatory school, where he was behind most of his classmates in terms of education. Despite this, he was a brilliant student and showed a strong aptitude for mathematics. His interests were varied, and he, for instance, also write poems for his mother and plays for the school theatre.

Gentzen began his university studies at the University of Greifswald, but moved around to Göttingen, Munich, and Berlin. He received his doctorate in 1933 from the University of Göttingen under Hermann Weyl. (Paul Bernays supervised most of his work, but was dismissed from the university by the Nazis.) In 1934, Gentzen began work as an assistant to David Hilbert. That same year he developed the sequent calculus and natural deduction proof systems, in his papers *Untersuchungen über das logische Schließen I–II [Investigations Into Logical Deduction I–II]*. He proved the consistency of the Peano axioms in 1936.

Gentzen's relationship with the Nazis is complicated. At the same time his mentor Bernays was forced to leave Germany, Gentzen joined the university branch of the SA, the Nazi paramilitary organization. Like many Germans, he was a member of the Nazi party. During the war, he served as a telecommunications officer for the air intelligence unit. However, in 1942 he was released from duty due to a nervous breakdown. It is unclear whether or not Gentzen's loyalties lay with the Nazi party, or whether he joined the party in order to ensure academic success.

In 1943, Gentzen was offered an academic position at the Mathematical Institute of the German University of Prague, which he accepted. However, in 1945 the citizens of Prague revolted against German occupation. Soviet forces arrived in the city and arrested all the professors at the university. Because of his membership in Nazi organizations, Gentzen was taken to a forced labour

camp. He died of malnutrition while in his cell on August 4, 1945 at the age of 35.

**Further Reading** For a full biography of Gentzen, see Menzler-Trott (2007). An interesting read about mathematicians under Nazi rule, which gives a brief note about Gentzen's life, is given by Segal (2014). Gentzen's papers on logical deduction are available in the original german (Gentzen, 1935a,b). English translations of Gentzen's papers have been collected in a single volume by Szabo (1969), which also includes a biographical sketch.

## 52.4 Kurt Gödel

Kurt Gödel (GER-dle) was born on April 28, 1906 in Brünn in the Austro-Hungarian empire (now Brno in the Czech Republic). Due to his inquisitive and bright nature, young Kurtele was often called "Der kleine Herr Warum" (Little Mr. Why) by his family. He excelled in academics from primary school onward, where he got less than the highest grade only in mathematics. Gödel was often absent from school due to poor health and was exempt from physical education. He was diagnosed with rheumatic fever during his childhood. Throughout his life, he believed this permanently affected his heart despite medical assessment saying otherwise.

Figure 52.4: Kurt Gödel

Gödel began studying at the University of Vienna in 1924 and completed his doctoral studies in 1929. He first intended to study physics, but his interests soon moved to mathematics and especially logic, in part due to the influence of the philosopher Rudolf Carnap. His dissertation, written under the supervision of Hans Hahn, proved the completeness theorem of first-order predicate logic with identity (Gödel, 1929). Only a year later, he obtained his most famous results—the first and second incompleteness theorems (published in Gödel 1931). During his time in Vienna, Gödel was heavily involved with the Vienna Circle, a group of scientifically-minded philosophers that included Carnap, whose work was especially influenced by Gödel's results.

In 1938, Gödel married Adele Nimbursky. His parents were not pleased: not only was she six years older than him and already divorced, but she

worked as a dancer in a nightclub. Social pressures did not affect Gödel, however, and they remained happily married until his death.

After Nazi Germany annexed Austria in 1938, Gödel and Adele emigrated to the United States, where he took up a position at the Institute for Advanced Study in Princeton, New Jersey. Despite his introversion and eccentric nature, Gödel's time at Princeton was collaborative and fruitful. He published essays in set theory, philosophy and physics. Notably, he struck up a particularly strong friendship with his colleague at the IAS, Albert Einstein.

In his later years, Gödel's mental health deteriorated. His wife's hospitalization in 1977 meant she was no longer able to cook his meals for him. Having suffered from mental health issues throughout his life, he succumbed to paranoia. Deathly afraid of being poisoned, Gödel refused to eat. He died of starvation on January 14, 1978, in Princeton.

**Further Reading** For a complete biography of Gödel's life is available, see John Dawson (1997). For further biographical pieces, as well as essays about Gödel's contributions to logic and philosophy, see Wang (1990), Baaz et al. (2011), Takeuti et al. (2003), and Sigmund et al. (2007).

Gödel's PhD thesis is available in the original German (Gödel, 1929). The original text of the incompleteness theorems is (Gödel, 1931). All of Gödel's published and unpublished writings, as well as a selection of correspondence, are available in English in his *Collected Papers* Feferman et al. (1986, 1990).

For a detailed treatment of Gödel's incompleteness theorems, see Smith (2013). For an informal, philosophical discussion of Gödel's theorems, see Mark Linsenmayer's podcast (Linsenmayer, 2014).

## 52.5 Emmy Noether

Emmy Noether (NER-ter) was born in Erlangen, Germany, on March 23, 1882, to an upper-middle class scholarly family. Hailed as the "mother of modern algebra," Noether made groundbreaking contributions to both mathematics and physics, despite significant barriers to women's education. In Germany at the time, young girls were meant to be educated in arts and were not allowed to attend college preparatory schools. However, after auditing classes at the Universities of Göttingen and Erlangen (where her father was professor of mathematics), Noether was eventually able to enrol as a student at Erlangen in 1904, when their policy was updated to allow female students. She received her doctorate in mathematics in 1907.

Despite her qualifications, Noether experienced much resistance during her career. From 1908–1915, she taught at Erlangen without pay. During this time, she caught the attention of David Hilbert, one of the world's foremost mathematicians of the time, who invited her to Göttingen. However, women were prohibited from obtaining professorships, and she was only able to lec-

ture under Hilbert's name, again without pay. During this time she proved what is now known as Noether's theorem, which is still used in theoretical physics today. Noether was finally granted the right to teach in 1919. Hilbert's response to continued resistance of his university colleagues reportedly was: "Gentlemen, the faculty senate is not a bathhouse."

Figure 52.5: Emmy Noether

In the later 1920s, she concentrated on work in abstract algebra, and her contributions revolutionized the field. In her proofs she often made use of the so-called ascending chain condition, which states that there is no infinite strictly increasing chain of certain sets. For instance, certain algebraic structures now known as Noetherian rings have the property that there are no infinite sequences of ideals $I_1 \subsetneq I_2 \subsetneq \ldots$. The condition can be generalized to any partial order (in algebra, it concerns the special case of ideals ordered by the subset relation), and we can also consider the dual descending chain condition, where every strictly *de*creasing sequence in a partial order eventually ends. If a partial order satisfies the descending chain condition, it is possible to use induction along this order in a similar way in which we can use induction along the $<$ order on $\mathbb{N}$. Such orders are called *well-founded* or *Noetherian*, and the corresponding proof principle *Noetherian induction*.

Noether was Jewish, and when the Nazis came to power in 1933, she was dismissed from her position. Luckily, Noether was able to emigrate to the United States for a temporary position at Bryn Mawr, Pennsylvania. During her time there she also lectured at Princeton, although she found the university to be unwelcoming to women (Dick, 1981, 81). In 1935, Noether underwent an operation to remove a uterine tumour. She died from an infection as a result of the surgery, and was buried at Bryn Mawr.

**Further Reading** For a biography of Noether, see Dick (1981). The Perimeter Institute for Theoretical Physics has their lectures on Noether's life and influence available online (Institute, 2015). If you're tired of reading, *Stuff You Missed in History Class* has a podcast on Noether's life and influence (Frey and Wilson, 2015). The collected works of Noether are available in the original German (Jacobson, 1983).

## 52.6 Rózsa Péter

Figure 52.6: Rózsa Péter

Rózsa Péter was born Rósza Politzer, in Budapest, Hungary, on February 17, 1905. She is best known for her work on recursive functions, which was essential for the creation of the field of recursion theory.

Péter was raised during harsh political times—WWI raged when she was a teenager—but was able to attend the affluent Maria Terezia Girls' School in Budapest, from where she graduated in 1922. She then studied at Pázmány Péter University (later renamed Loránd Eötvös University) in Budapest. She began studying chemistry at the insistence of her father, but later switched to mathematics, and graduated in 1927. Although she had the credentials to teach high school mathematics, the economic situation at the time was dire as the Great Depression affected the world economy. During this time, Péter took odd jobs as a tutor and private teacher of mathematics. She eventually returned to university to take up graduate studies in mathematics. She had originally planned to work in number theory, but after finding out that her results had already been proven, she almost gave up on mathematics altogether. She was encouraged to work on Gödel's incompleteness theorems, and unknowingly proved several of his results in different ways. This restored her confidence, and Péter went on to write her first papers on recursion theory, inspired by David Hilbert's foundational program. She received her PhD in 1935, and in 1937 she became an editor for the *Journal of Symbolic Logic*.

Péter's early papers are widely credited as founding contributions to the field of recursive function theory. In Péter (1935a), she investigated the relationship between different kinds of recursion. In Péter (1935b), she showed that a certain recursively defined function is not primitive recursive. This simplified an earlier result due to Wilhelm Ackermann. Péter's simplified function is what's now often called the Ackermann function—and sometimes, more properly, the Ackermann-Péter function. She wrote the first book on recursive function theory (Péter, 1951).

Despite the importance and influence of her work, Péter did not obtain a full-time teaching position until 1945. During the Nazi occupation of Hungary during World War II, Péter was not allowed to teach due to anti-Semitic laws. In 1944 the government created a Jewish ghetto in Budapest; the ghetto was cut off from the rest of the city and attended by armed guards. Péter was

forced to live in the ghetto until 1945 when it was liberated. She then went on to teach at the Budapest Teachers Training College, and from 1955 onward at Eötvös Loránd University. She was the first female Hungarian mathematician to become an Academic Doctor of Mathematics, and the first woman to be elected to the Hungarian Academy of Sciences.

Péter was known as a passionate teacher of mathematics, who preferred to explore the nature and beauty of mathematical problems with her students rather than to merely lecture. As a result, she was affectionately called "Aunt Rosa" by her students. Péter died in 1977 at the age of 71.

**Further Reading** For more biographical reading, see (O'Connor and Robertson, 2014) and (Andrásfai, 1986). Tamassy (1994) conducted a brief interview with Péter. For a fun read about mathematics, see Péter's book *Playing With Infinity* (Péter, 2010).

## 52.7 Julia Robinson

Julia Bowman Robinson was an American mathematician. She is known mainly for her work on decision problems, and most famously for her contributions to the solution of Hilbert's tenth problem. Robinson was born in St. Louis, Missouri on December 8, 1919. At a young age Robinson recalls being intrigued by numbers (Reid, 1986, 4). At age nine she contracted scarlet fever and suffered from several recurrent bouts of rheumatic fever. This forced her to spend much of her time in bed, putting her behind in her education. Although she was able to catch up with the help of private tutors, the physical effects of her illness had a lasting impact on her life.

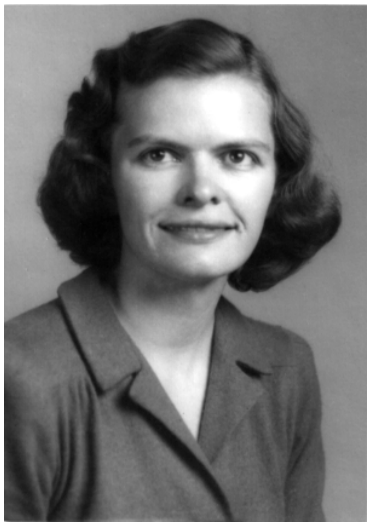


Figure 52.7: Julia Robinson

Despite her childhood struggles, Robinson graduated high school with several awards in mathematics and the sciences. She started her university career at San Diego State College, and transferred to the University of California, Berkeley as a senior. There she was highly influenced by mathematician Raphael Robinson. They quickly became good friends, and married in 1941. As a spouse of a faculty member, Robinson was barred from teaching in the mathematics department at Berkeley. Although she continued to audit mathematics classes, she hoped to leave university and start a family. Not long after

her wedding, however, Robinson contracted pneumonia. She was told that there was substantial scar tissue build up on her heart due to the rheumatic fever she suffered as a child. Due to the severity of the scar tissue, the doctor predicted that she would not live past forty and she was advised not to have children (Reid, 1986, 13).

Robinson was depressed for a long time, but eventually decided to continue studying mathematics. She returned to Berkeley and completed her PhD in 1948 under the supervision of Alfred Tarski. The first-order theory of the real numbers had been shown to be decidable by Tarski, and from Gödel's work it followed that the first-order theory of the natural numbers is undecidable. It was a major open problem whether the first-order theory of the rationals is decidable or not. In her thesis (1949), Robinson proved that it was not.

Interested in decision problems, Robinson next attempted to find a solution Hilbert's tenth problem. This problem was one of a famous list of 23 mathematical problems posed by David Hilbert in 1900. The tenth problem asks whether there is an algorithm that will answer, in a finite amount of time, whether or not a polynomial equation with integer coefficients, such as $3x^2 - 2y + 3 = 0$, has a solution in the integers. Such questions are known as *Diophantine problems*. After some initial successes, Robinson joined forces with Martin Davis and Hilary Putnam, who were also working on the problem. They succeeded in showing that exponential Diophantine problems (where the unknowns may also appear as exponents) are undecidable, and showed that a certain conjecture (later called "J.R.") implies that Hilbert's tenth problem is undecidable (Davis et al., 1961). Robinson continued to work on the problem for the next decade. In 1970, the young Russian mathematician Yuri Matijasevich finally proved the J.R. hypothesis. The combined result is now called the Matijasevich-Robinson-Davis-Putnam theorem, or MDRP theorem for short. Matijasevich and Robinson became friends and collaborated on several papers. In a letter to Matijasevich, Robinson once wrote that "actually I am very pleased that working together (thousands of miles apart) we are obviously making more progress than either one of us could alone" (Matijasevich, 1992, 45).

Robinson was the first female president of the American Mathematical Society, and the first woman to be elected to the National Academy of Science. She died on July 30, 1985 at the age of 65 after being diagnosed with leukemia.

**Further Reading** Robinson's mathematical papers are available in her *Collected Works* (Robinson, 1996), which also includes a reprint of her National Academy of Sciences biographical memoir (Feferman, 1994). Robinson's older sister Constance Reid published an "Autobiography of Julia," based on interviews (Reid, 1986), as well as a full memoir (Reid, 1996). A short documentary about Robinson and Hilbert's tenth problem was directed by George Csicsery

(Csicsery, 2016). For a brief memoir about Yuri Matijasevich's collaborations with Robinson, and her influence on his work, see (Matijasevich, 1992).

## 52.8 Bertrand Russell

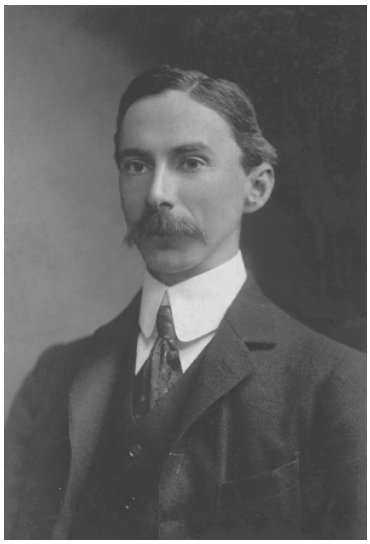


Figure 52.8: Bertrand Russell

Bertrand Russell is hailed as one of the founders of modern analytic philosophy. Born May 18, 1872, Russell was not only known for his work in philosophy and logic, but wrote many popular books in various subject areas. He was also an ardent political activist throughout his life.

Russell was born in Trellech, Monmouthshire, Wales. His parents were members of the British nobility. They were free-thinkers, and even made friends with the radicals in Boston at the time. Unfortunately, Russell's parents died when he was young, and Russell was sent to live with his grandparents. There, he was given a religious upbringing (something his parents had wanted to avoid at all costs). His grandmother was very strict in all matters of morality. During adolescence he was mostly homeschooled by private tutors.

Russell's influence in analytic philosophy, and especially logic, is tremendous. He studied mathematics and philosophy at Trinity College, Cambridge, where he was influenced by the mathematician and philosopher Alfred North Whitehead. In 1910, Russell and Whitehead published the first volume of *Principia Mathematica*, where they championed the view that mathematics is reducible to logic. He went on to publish hundreds of books, essays and political pamphlets. In 1950, he won the Nobel Prize for literature.

Russell's was deeply entrenched in politics and social activism. During World War I he was arrested and sent to prison for six months due to pacifist activities and protest. While in prison, he was able to write and read, and claims to have found the experience "quite agreeable." He remained a pacifist throughout his life, and was again incarcerated for attending a nuclear disarmament rally in 1961. He also survived a plane crash in 1948, where the only survivors were those sitting in the smoking section. As such, Russell claimed that he owed his life to smoking. Russell was married four times, but had a reputation for carrying on extra-marital affairs. He died on February 2, 1970 at the age of 97 in Penrhyndeudraeth, Wales.

**Further Reading** Russell wrote an autobiography in three parts, spanning his life from 1872–1967 (Russell, 1967, 1968, 1969). The Bertrand Russell Research Centre at McMaster University is home of the Bertrand Russell archives. See their website at Duncan (2015), for information on the volumes of his collected works (including searchable indexes), and archival projects. Russell's paper *On Denoting* (Russell, 1905) is a classic of 20th century analytic philosophy.

The Stanford Encyclopedia of Philosophy entry on Russell (Irvine, 2015) has sound clips of Russell speaking on Desire and Political theory. Many video interviews with Russell are available online. To see him talk about smoking and being involved in a plane crash, e.g., see Russell (n.d.). Some of Russell's works, including his *Introduction to Mathematical Philosophy* are available as free audiobooks on LibriVox (n.d.).

## 52.9 Alfred Tarski

Alfred Tarski was born on January 14, 1901 in Warsaw, Poland (then part of the Russian Empire). Often described as "Napoleonic," Tarski was boisterous, talkative, and intense. His energy was often reflected in his lectures—he once set fire to a wastebasket while disposing of a cigarette during a lecture, and was forbidden from lecturing in that building again.

Figure 52.9: Alfred Tarski

Tarski had a thirst for knowledge from a young age. Although later in life he would tell students that he studied logic because it was the only class in which he got a B, his high school records show that he got A's across the board—even in logic. He studied at the University of Warsaw from 1918 to 1924. Tarski first intended to study biology, but became interested in mathematics, philosophy, and logic, as the university was the center of the Warsaw School of Logic and Philosophy. Tarski earned his doctorate in 1924 under the supervision of Stanisław Leśniewski.

Before emigrating to the United States in 1939, Tarski completed some of his most important work while working as a secondary school teacher in Warsaw. His work on logical consequence and logical truth were written during this time. In 1939, Tarski was visiting the United States for a lecture tour. During his visit, Germany invaded Poland, and because of his Jewish heritage,

Tarski could not return. His wife and children remained in Poland until the end of the war, but were then able to emigrate to the United States as well. Tarski taught at Harvard, the College of the City of New York, and the Institute for Advanced Study at Princeton, and finally the University of California, Berkeley. There he founded the multidisciplinary program in Logic and the Methodology of Science. Tarski died on October 26, 1983 at the age of 82.

**Further Reading** For more on Tarski's life, see the biography *Alfred Tarski: Life and Logic* (Feferman and Feferman, 2004). Tarski's seminal works on logical consequence and truth are available in English in (Corcoran, 1983). All of Tarski's original works have been collected into a four volume series, (Tarski, 1981).

## 52.10 Alan Turing

Alan Turing was born in Mailda Vale, London, on June 23, 1912. He is considered the father of theoretical computer science. Turing's interest in the physical sciences and mathematics started at a young age. However, as a boy his interests were not represented well in his schools, where emphasis was placed on literature and classics. Consequently, he did poorly in school and was reprimanded by many of his teachers.

Turing attended King's College, Cambridge as an undergraduate, where he studied mathematics. In 1936 Turing developed (what is now called) the Turing machine as an attempt to precisely define the notion of a computable function and to prove the undecidability of the decision problem. He was beaten to the result by Alonzo Church, who proved the result via his own lambda calculus. Turing's paper was still published with reference to Church's result. Church invited Turing to Princeton, where he spent 1936–1938, and obtained a doctorate under Church.

Figure 52.10: Alan Turing

Despite his interest in logic, Turing's earlier interests in physical sciences remained prevalent. His practical skills were put to work during his service with the British cryptanalytic department at Bletchley Park during World War II. Turing was a central figure in cracking the cypher used by German Naval communications—the Enigma code. Turing's expertise in statistics and cryptography, together with the in-

troduction of electronic machinery, gave the team the ability to crack the code by creating a de-crypting machine called a "bombe." His ideas also helped in the creation of the world's first programmable electronic computer, the Colossus, also used at Bletchley park to break the German Lorenz cypher.

Turing was gay. Nevertheless, in 1942 he proposed to Joan Clarke, one of his teammates at Bletchley Park, but later broke off the engagement and confessed to her that he was homosexual. He had several lovers throughout his lifetime, although homosexual acts were then criminal offences in the UK. In 1952, Turing's house was burgled by a friend of his lover at the time, and when filing a police report, Turing admitted to having a homosexual relationship, under the impression that the government was on their way to legalizing homosexual acts. This was not true, and he was charged with gross indecency. Instead of going to prison, Turing opted for a hormone treatment that reduced libido. Turing was found dead on June 8, 1954, of a cyanide overdose—most likely suicide. He was given a royal pardon by Queen Elizabeth II in 2013.

**Further Reading** For a comprehensive biography of Alan Turing, see Hodges (2014). Turing's life and work inspired a play, *Breaking the Code*, which was produced in 1996 for TV starring Derek Jacobi as Turing. *The Imitation Game*, an Academy Award nominated film starring Bendict Cumberbatch and Kiera Knightley, is also loosely based on Alan Turing's life and time at Bletchley Park (Tyldum, 2014).

Radiolab (2012) has several podcasts on Turing's life and work. BBC Horizon's documentary *The Strange Life and Death of Dr. Turing* is available to watch online (Sykes, 1992). (Theelen, 2012) is a short video of a working LEGO Turing Machine—made to honour Turing's centenary in 2012.

Turing's original paper on Turing machines and the decision problem is Turing (1937).

## 52.11 Ernst Zermelo

Ernst Zermelo was born on July 27, 1871 in Berlin, Germany. He had five sisters, though his family suffered from poor health and only three survived to adulthood. His parents also passed away when he was young, leaving him and his siblings orphans when he was seventeen. Zermelo had a deep interest in the arts, and especially in poetry. He was known for being sharp, witty, and critical. His most celebrated mathematical achievements include the introduction of the axiom of choice (in 1904), and his axiomatization of set theory (in 1908).

Zermelo's interests at university were varied. He took courses in physics, mathematics, and philosophy. Under the supervision of Hermann Schwarz, Zermelo completed his dissertation *Investigations in the Calculus of Variations* in 1894 at the University of Berlin. In 1897, he decided to pursue more studies

at the University of Göttigen, where he was heavily influenced by the foundational work of David Hilbert. In 1899 he became eligible for professorship, but did not get one until eleven years later—possibly due to his strange demeanour and "nervous haste."

Zermelo finally received a paid professorship at the University of Zurich in 1910, but was forced to retire in 1916 due to tuberculosis. After his recovery, he was given an honourary professorship at the University of Freiburg in 1921. During this time he worked on foundational mathematics. He became irritated with the works of Thoralf Skolem and Kurt Gödel, and publicly criticized their approaches in his papers. He was dismissed from his position at Freiburg in 1935, due to his unpopularity and his opposition to Hitler's rise to power in Germany.

Figure 52.11: Ernst Zermelo

The later years of Zermelo's life were marked by isolation. After his dismissal in 1935, he abandoned mathematics. He moved to the country where he lived modestly. He married in 1944, and became completely dependent on his wife as he was going blind. Zermelo lost his sight completely by 1951. He passed away in Günterstal, Germany, on May 21, 1953.

**Further Reading** For a full biography of Zermelo, see Ebbinghaus (2015). Zermelo's seminal 1904 and 1908 papers are available to read in the original German (Zermelo, 1904, 1908). Zermelo's collected works, including his writing on physics, are available in English translation in (Ebbinghaus et al., 2010;

Ebbinghaus and Kanamori, 2013).

# Photo Credits

Georg Cantor, p. 703: Portrait of Georg Cantor by Otto Zeth courtesy of the Universitätsarchiv, Martin-Luther Universität Halle–Wittenberg. UAHW Rep. 40-VI, Nr. 3 Bild 102.

Alonzo Church, p. 704: Portrait of Alonzo Church, undated, photographer unknown. Alonzo Church Papers; 1924–1995, (C0948) Box 60, Folder 3. Manuscripts Division, Department of Rare Books and Special Collections, Princeton University Library. © Princeton University. The Open Logic Project has obtained permission to use this image for inclusion in non-commercial OLP-derived materials. Permission from Princeton University is required for any other use.

Gerhard Gentzen, p. 705: Portrait of Gerhard Gentzen playing ping-pong courtesy of Ekhart Mentzler-Trott.

Kurt Gödel, p. 706: Portrait of Kurt Gödel, ca. 1925, photographer unknown. From the Shelby White and Leon Levy Archives Center, Institute for Advanced Study, Princeton, NJ, USA, on deposit at Princeton University Library, Manuscript Division, Department of Rare Books and Special Collections, Kurt Gödel Papers, (C0282), Box 14b, #110000. The Open Logic Project has obtained permission from the Institute's Archives Center to use this image for inclusion in non-commercial OLP-derived materials. Permission from the Archives Center is required for any other use.

Emmy Noether, p. 708: Portrait of Emmy Noether, ca. 1922, courtesy of the Abteilung für Handschriften und Seltene Drucke, Niedersächsische Staats- und Universitätsbibliothek Göttingen, Cod. Ms. D. Hilbert 754, Bl. 14 Nr. 73. Restored from an original scan by Joel Fuller.

Rózsa Péter, p. 709: Portrait of Rózsa Péter, undated, photographer unknown. Courtesy of Béla Andrásfai.

Julia Robinson, p. 710: Portrait of Julia Robinson, unknown photographer, courtesy of Neil D. Reid. The Open Logic Project has obtained permission to use this image for inclusion in non-commercial OLP-derived materials. Permission is required for any other use.

Bertrand Russell, p. 712: Portrait of Bertrand Russell, ca. 1907, courtesy of the William Ready Division of Archives and Research Collections, McMaster University Library. Bertrand Russell Archives, Box 2, f. 4.

Alfred Tarski, p. 713: Passport photo of Alfred Tarski, 1939. Cropped and restored from a scan of Tarski's passport by Joel Fuller. Original courtesy of Bancroft Library, University of California, Berkeley. Alfred Tarski Papers, Banc MSS 84/49. The Open Logic Project has obtained permission to use this image for inclusion in non-commercial OLP-derived materials. Permission from Bancroft Library is required for any other use.

Alan Turing, p. 714: Portrait of Alan Mathison Turing by Elliott & Fry, 29 March 1951, NPG x82217, © National Portrait Gallery, London. Used under a Creative Commons BY-NC-ND 3.0 license.

Ernst Zermelo, p. 716: Portrait of Ernst Zermelo, ca. 1922, courtesy of the Abteilung für Handschriften und Seltene Drucke, Niedersächsische Staats- und Universitätsbibliothek Göttingen, Cod. Ms. D. Hilbert 754, Bl. 6 Nr. 25.

# Bibliography


Andrásfai, Béla. 1986. Rózsa (Rosa) Péter. *Periodica Polytechnica Electrical Engineering* 30(2-3): 139–145. URL http://www.pp.bme.hu/ee/article/view/4651.

Aspray, William. 1984. The Princeton mathematics community in the 1930s: Alonzo Church. URL http://www.princeton.edu/mudd/finding_aids/mathoral/pmc05.htm. Interview.

Baaz, Matthias, Christos H. Papadimitriou, Hilary W. Putnam, Dana S. Scott, and Charles L. Harper Jr. 2011. *Kurt Gödel and the Foundations of Mathematics: Horizons of Truth*. Cambridge: Cambridge University Press.

Church, Alonzo. 1936a. A note on the Entscheidungsproblem. *Journal of Symbolic Logic* 1: 40–41.

Church, Alonzo. 1936b. An unsolvable problem of elementary number theory. *American Journal of Mathematics* 58: 345–363.

Corcoran, John. 1983. *Logic, Semantics, Metamathematics*. Indianapolis: Hackett, 2nd ed.

Csicsery, George. 2016. Zala films: Julia Robinson and Hilbert's tenth problem. URL http://www.zalafilms.com/films/juliarobinson.html.

Dauben, Joseph. 1990. *Georg Cantor: His Mathematics and Philosophy of the Infinite*. Princeton: Princeton University Press.

Davis, Martin, Hilary Putnam, and Julia Robinson. 1961. The decision problem for exponential Diophantine equations. *Annals of Mathematics* 74(3): 425–436. URL http://www.jstor.org/stable/1970289.

Dick, Auguste. 1981. *Emmy Noether 1882–1935*. Boston: Birkhäuser.

du Sautoy, Marcus. 2014. A brief history of mathematics: Georg Cantor. URL http://www.bbc.co.uk/programmes/b00ss1j0. Audio Recording.

Duncan, Arlene. 2015. The Bertrand Russell Research Centre. URL http://russell.mcmaster.ca/.

Ebbinghaus, Heinz-Dieter. 2015. *Ernst Zermelo: An Approach to his Life and Work*. Berlin: Springer-Verlag.

Ebbinghaus, Heinz-Dieter, Craig G. Fraser, and Akihiro Kanamori. 2010. *Ernst Zermelo. Collected Works*, vol. 1. Berlin: Springer-Verlag.

Ebbinghaus, Heinz-Dieter and Akihiro Kanamori. 2013. *Ernst Zermelo: Collected Works*, vol. 2. Berlin: Springer-Verlag.

Enderton, Herbert B. forthcoming. Alonzo Church: Life and Work. In *The Collected Works of Alonzo Church*. Cambridge: MIT Press.

Feferman, Anita and Solomon Feferman. 2004. *Alfred Tarski: Life and Logic*. Cambridge: Cambridge University Press.

Feferman, Solomon. 1994. Julia Bowman Robinson 1919–1985. *Biographical Memoirs of the National Academy of Sciences* 63: 1–28. URL `http://www.nasonline.org/publications/biographical-memoirs/memoir-pdfs/robinson-julia.pdf`.

Feferman, Solomon, John W. Dawson Jr., Stephen C. Kleene, Gregory H. Moore, Robert M. Solovay, and Jean van Heijenoort. 1986. *Kurt Gödel: Collected Works. Vol. 1: Publications 1929–1936*. Oxford: Oxford University Press.

Feferman, Solomon, John W. Dawson Jr., Stephen C. Kleene, Gregory H. Moore, Robert M. Solovay, and Jean van Heijenoort. 1990. *Kurt Gödel: Collected Works. Vol. 2: Publications 1938–1974*. Oxford: Oxford University Press.

Frey, Holly and Tracy V. Wilson. 2015. Stuff you missed in history class: Emmy Noether, mathematics trailblazer. URL `http://www.missedinhistory.com/podcasts/emmy-noether-mathematics-trailblazer/`. Podcast audio.

Gentzen, Gerhard. 1935a. Untersuchungen über das logische Schließen I. *Mathematische Zeitschrift* 39: 176–210. English translation in Szabo (1969), pp. 68–131.

Gentzen, Gerhard. 1935b. Untersuchungen über das logische Schließen II. *Mathematische Zeitschrift* 39: 176–210, 405–431. English translation in Szabo (1969), pp. 68–131.

Gödel, Kurt. 1929. Über die Vollständigkeit des Logikkalküls [On the completeness of the calculus of logic]. Dissertation, Universität Wien. Reprinted and translated in Feferman et al. (1986), pp. 60–101.

Gödel, Kurt. 1931. über formal unentscheidbare Sätze der *Principia Mathematica* und verwandter Systeme I [On formally undecidable propositions of *Principia Mathematica* and related systems I]. *Monatshefte für Mathematik*

*und Physik* 38: 173–198. Reprinted and translated in Feferman et al. (1986), pp. 144–195.

Grattan-Guinness, Ivor. 1971. Towards a biography of Georg Cantor. *Annals of Science* 27(4): 345–391.

Hodges, Andrew. 2014. *Alan Turing: The Enigma*. London: Vintage.

Institute, Perimeter. 2015. Emmy Noether: Her life, work, and influence. URL `https://www.youtube.com/watch?v=tNNyAyMRsgE`. Video Lecture.

Irvine, Andrew David. 2015. Sound clips of Bertrand Russell speaking. URL `http://plato.stanford.edu/entries/russell/russell-soundclips.html`.

Jacobson, Nathan. 1983. *Emmy Noether: Gesammelte Abhandlungen—Collected Papers*. Berlin: Springer-Verlag.

John Dawson, Jr. 1997. *Logical Dilemmas: The Life and Work of Kurt Gödel*. Boca Raton: CRC Press.

LibriVox. n.d. Bertrand Russell. URL `https://librivox.org/author/1508?primary_key=1508&search_category=author&search_page=1&search_form=get_results`. Collection of public domain audiobooks.

Linsenmayer, Mark. 2014. The partially examined life: Gödel on math. URL `http://www.partiallyexaminedlife.com/2014/06/16/ep95-godel/`. Podcast audio.

MacFarlane, John. 2015. Alonzo Church's JSL reviews. URL `http://johnmacfarlane.net/church.html`.

Matijasevich, Yuri. 1992. My collaboration with Julia Robinson. *The Mathematical Intelligencer* 14(4): 38–45.

Menzler-Trott, Eckart. 2007. *Logic's Lost Genius: The Life of Gerhard Gentzen*. Providence: American Mathematical Society.

O'Connor, John J. and Edmund F. Robertson. 2014. Rózsa Péter. URL `http://www-groups.dcs.st-and.ac.uk/~history/Biographies/Peter.html`.

Péter, Rózsa. 1935a. Über den Zusammenhang der verschiedenen Begriffe der rekursiven Funktion. *Mathematische Annalen* 110: 612–632.

Péter, Rózsa. 1935b. Konstruktion nichtrekursiver Funktionen. *Mathematische Annalen* 111: 42–60.

Péter, Rózsa. 1951. *Rekursive Funktionen*. Budapest: Akademiai Kiado. English translation in (Péter, 1967).

Péter, Rózsa. 1967. *Recursive Functions*. New York: Academic Press.

Péter, Rózsa. 2010. *Playing with Infinity*. New York: Dover. URL https://books.google.ca/books?id=6V3wNs4uv_4C&lpg=PP1&ots=BkQZaHcR99&lr&pg=PP1#v=onepage&q&f=false.

Radiolab. 2012. The Turing problem. URL http://www.radiolab.org/story/193037-turing-problem/. Podcast audio.

Reid, Constance. 1986. The autobiography of Julia Robinson. *The College Mathematics Journal* 17: 3–21.

Reid, Constance. 1996. *Julia: A Life in Mathematics*. Cambridge: Cambridge University Press. URL https://books.google.ca/books?id=lRtSzQyHf9UC&lpg=PP1&pg=PP1#v=onepage&q&f=false.

Robinson, Julia. 1949. Definability and decision problems in arithmetic. *Journal of Symbolic Logic* 14(2): 98–114. URL http://www.jstor.org/stable/2266510.

Robinson, Julia. 1996. *The Collected Works of Julia Robinson*. Providence: American Mathematical Society.

Rose, Daniel. 2012. A song about Georg Cantor. URL https://www.youtube.com/watch?v=QUP5Z4Fb5k4. Audio Recording.

Russell, Bertrand. 1905. On denoting. *Mind* 14: 479–493.

Russell, Bertrand. 1967. *The Autobiography of Bertrand Russell*, vol. 1. London: Allen and Unwin.

Russell, Bertrand. 1968. *The Autobiography of Bertrand Russell*, vol. 2. London: Allen and Unwin.

Russell, Bertrand. 1969. *The Autobiography of Bertrand Russell*, vol. 3. London: Allen and Unwin.

Russell, Bertrand. n.d. Bertrand Russell on smoking. URL https://www.youtube.com/watch?v=80oLTiVW_lc. Video Interview.

Segal, Sanford L. 2014. *Mathematicians under the Nazis*. Princeton: Princeton University Press.

Sigmund, Karl, John Dawson, Kurt Mühlberger, Hans Magnus Enzensberger, and Juliette Kennedy. 2007. Kurt Gödel: Das Album–The Album. *The Mathematical Intelligencer* 29(3): 73–76.

Smith, Peter. 2013. *An Introduction to Gödel's Theorems*. Cambridge: Cambridge University Press.

Sykes, Christopher. 1992. BBC Horizon: The strange life and death of Dr. Turing. URL `https://www.youtube.com/watch?v=gyusnGbBSHE`.

Szabo, Manfred E. 1969. *The Collected Papers of Gerhard Gentzen*. Amsterdam: North-Holland.

Takeuti, Gaisi, Nicholas Passell, and Mariko Yasugi. 2003. *Memoirs of a Proof Theorist: Gödel and Other Logicians*. Singapore: World Scientific.

Tamassy, Istvan. 1994. Interview with Róza Péter. *Modern Logic* 4(3): 277–280.

Tarski, Alfred. 1981. *The Collected Works of Alfred Tarski*, vol. I–IV. Basel: Birkhäuser.

Theelen, Andre. 2012. Lego turing machine. URL `https://www.youtube.com/watch?v=FTSAiF9AHN4`.

Turing, Alan M. 1937. On computable numbers, with an application to the "Entscheidungsproblem". *Proceedings of the London Mathematical Society, 2nd Series* 42: 230–265.

Tyldum, Morten. 2014. The imitation game. Motion picture.

Wang, Hao. 1990. *Reflections on Kurt Gödel*. Cambridge: MIT Press.

Zermelo, Ernst. 1904. Beweis, daß jede Menge wohlgeordnet werden kann. *Mathematische Annalen* 59: 514–516. English translation in (Ebbinghaus et al., 2010, pp. 115-119).

Zermelo, Ernst. 1908. Untersuchungen über die Grundlagen der Mengenlehre I. *Mathematische Annalen* 65(2): 261–281. English translation in (Ebbinghaus et al., 2010, pp. 189-229).