

MathematicalLogic

Richard Grandy

Michael Barkasi

Joshua Reagan

Draft 0.71

Copyright © 2021 Richard Grandy, Michael Barkasi, and Joshua Reagan

This work is licensed under the Creative Commons Attribution-NonCommercial-NoDerivs 3.0 Unported License.

To view a copy of the license, visit <http://creativecommons.org/licenses/by-nc-nd/3.0/>.

You are free to Share – to copy, distribute and transmit the work,
under the following conditions:

Attribution – You must attribute the work to the authors (but not in any way that suggests that they endorse you or your use of the work);

Noncommercial – You may not use this work for commercial purposes;

No Derivative Works – You may not alter, transform, or build upon this work.

This book is set in Latin Modern (for text and math fonts, released under the GUST Font License) and Adobe Source Sans Pro (for headings, released by Adobe under the OFL).

Edition 0.7 (draft), 2020

Contents

Preface	v
Acknowledgments	ix
1 Introduction	1
1.1 What is Logic?	1
1.2 Natural and Formal Languages	3
1.3 Some Distinctions	7
1.4 MathEnglish	8
1.5 Exercises	13
2 Sentential Logic	15
2.1 The Language SL	15
2.2 Models	21
2.3 Entailment and other Relations	31
2.4 Recursive Proofs	38
2.5 Recursive Proofs in SL	41
2.6 Disjunctive Normal Form	45
2.7 Truth Functional Expressiveness	50
2.8 Exercises	53
3 Quantifier Language I	61
3.1 The Language QL1	61
3.2 Models	68
3.3 Entailment and other Relations	79
3.4 Exercises	82
4 Quantifier Language II	85
4.1 The Language QL	85
4.2 Models	88
4.3 The Dragnet Theorem	94
4.4 Exercises	103

5 Translations	107
5.1 SL Applications	107
5.2 QL Applications	119
5.3 Exercises	123
6 Derivations	127
6.1 Introduction	127
6.2 The Basic System SD	128
6.3 Shortcut Rules for SD	148
6.4 QD	164
6.5 Exercises	176
7 Soundness and Completeness	181
7.1 Introduction	181
7.2 Soundness	182
7.3 Completeness	188
7.4 Completeness of QD	194
7.5 Strong Completeness and Other Results	205
7.6 Decidability and Church's Theorem	209
7.7 Löwenheim-Skolem and Compactness	211
7.8 Exercises	212
8 Further Directions	213
8.1 Many-Valued Logic	213
8.2 Modal Sentential Logic	217
8.3 Quantifier Logic with Identity	223
8.4 Exercises	229
Appendix A: List of Theorems	233
Appendix B: List of Derivation Rules	241
Works Cited	247
Index	255

Preface

What is mathematical logic?

Logic is the study of logical consequence and, by extension, logical truth. Logical consequence is a relation between a sentence Φ and a set of sentences Δ . A common way to define logical consequence, modeled on Aristotle's discussion of syllogisms, is:

Φ is a *logical consequence* of Δ if and only if it is not possible for the members of Δ to be true and Φ to be false.

Likewise, a sentence Φ is a logical truth if and only if it cannot possibly be false. We can think of logical truth as a special case: a sentence Φ is a *logical truth* if and only if it is a logical consequence of the empty set; so, i.e., $\Delta = \emptyset$. However, the sense of 'possible' at work in this definition needs clarification. Usually it's thought that a logical consequence holds in virtue of the meaning of "logical" terms and the overall form of the sentences. For example, 'Fran is a Wallaby' is a logical consequence of 'If Fran was born last year, then Fran is a Wallaby' and 'Fran was born last year'. That's because, in virtue of the meaning of 'If ... then', any sentence Ψ must be true whenever both of the sentences 'If Φ then Ψ ,' and Φ are true. Because 'meaning' is a notoriously difficult concept to define precisely, we use a formal semantics, giving an exact specification for logical terms.

In *mathematical* logic we use mathematical methods to study logical consequence and logical truth. In this textbook in particular, we define abstract, formal languages which have as sentences strings of basic symbols. For sentences of these formal languages, we define (i) the relation of entailment and (ii) formal derivation systems. Put roughly, both entailment and formal derivations provide a mathematical approximation of logical consequence. The goal of this textbook is simply to give the reader an introduction to those mathematical methods. Questions about how logical consequence relates to entailment and derivation are largely set aside.

What material is covered here?

The main subject of this text is classical first-order logic (or as we call it, quantificational logic), with sentential logic (sometimes called 'propositional' logic) introduced first. We focus mostly on a quantificational logic that doesn't include identity and function symbols. Our conjunction and disjunction operators are of arbitrary (but

finite) arity because that more closely mirrors typical English use. Many-valued logic, modal logic, and the identity operator are briefly introduced in the last chapter. The main purpose there is to show how sentential and quantificational logic can be extended to treat other logical operators and to cover a wider range of applications.

The main goal of the text is to prove that for quantificational logic, whether one uses formal derivations or proofs of logical entailment, one always gets the same results. There are various ways to define the semantics for quantificational logic.¹ We use Benson Mates’ modification of the standard Tarskian model-theoretic semantics.² The derivation system in this text is a natural deduction style system.³ In the system used here only conditional elimination can discharge assumptions; this tends to make the other rules simpler. We use Fitch-style derivations,⁴ but following Donald Kalish and Richard Montague⁵ we discharge assumptions by drawing a box around each completed subproof. Rather than using the more commonly found Henkin-style proofs⁶ to prove strong completeness, we prove it constructively. We believe this makes the connection between syntactic and semantic methods more intuitive. In Chapter 7 we provide an algorithm (“The Method”) which, for any sentence Φ entailed by some set of sentences Δ , produces a derivation of Φ from Δ . The proof used here owes much to Willard Quine’s completeness proof.⁷

How can I effectively use this book?

Reading mathematical texts is difficult, especially for those not accustomed to the style of prose typical of the genre. Following a mathematical argument requires significant cognitive effort and concentration, so prepare yourself. The most fruitful way to approach this text is with pencil and paper. As you read, work out the examples, write the proofs, and find counterexamples on your own. There’s no reason to stick to the text—look for your own proofs and work out new examples. Effective use of the text requires active engagement.

We have tried our best to prepare a helpful and clear text. A large stock of examples is included and most intratext references include a page number. Many of the proofs in the text, especially the recursive proofs, leave steps for the reader to complete. These are always steps (or whole proofs) for which the reader will have seen something similar before. Many of the examples have been worked through in detail. Our hope is that the examples show clearly all the “moving parts”—that they show

¹For example, game-theoretic semantics and substitutional quantifiers; see [Dunn and Belnap 1968](#), [Stevenson 1973](#), [Hintikka 1996](#), [Leblanc 2001](#), [Westerståhl 2001](#), [Jacquette 2002](#).

²Mates 1972

³This style of system was first developed by Gentzen in (1934), [Hodges 2001b](#): 26. If you are curious, derivation systems roughly divide into four types: natural deduction, Hilbert-style axiomatic systems, Gentzen-style sequent calculi, and proof tableaux; see [Hodges 2001a](#): 24 for an overview.

⁴Fitch 1952

⁵Kalish and Montague 1964

⁶Henkin 1949. For one of many modern treatments, see [Bergmann et al. 2003](#): ch. 11.4.

⁷Quine 1982

how the answers to problems follow directly from the definitions and theorems. You probably won't learn much logic by reading the text once or twice, so work through the examples and theorems.

Issues in the Philosophy of Logic

In mathematical logic, one begins with rigorous mathematical definitions of “sentence”, truth, logical truth, entailment, and derivation and then proves theorems about them. A primary concern in logic is how to make sense of reasoning in natural languages and the notion of logical consequence. How the two—the rigorously defined mathematical concepts like entailment and “pretheoretic” concepts like logical consequence—relate is a natural question, and one that's received a great deal of attention by philosophers.⁸

Because the aim of this text is to introduce mathematical logic and not philosophical logic, we largely set these questions aside. In some places, usually for the sake of exposition or motivation, we make claims that raise issues squarely in the domain of philosophy of logic.⁹ Given the aims and scope of this text, however, we aren't able to point out all the relevant philosophical concerns, let alone do much to motivate our potentially contentious claims.

History of Logic

Logic has a long history going back to Aristotle. Aristotle claims that he was the first to work out any systematic treatment of logical consequence (*Sophistical Refutations*, 34, 183b34–36; citation from [Smith 1995](#): 27), and as Robin Smith notes, “we have no reason to dispute this” ([1995](#): 27). Aristotle's work gave rise to a tradition of work in logic that extends through the Stoics, Byzantine commentators, early Islamic philosophy, and European Scholasticism. Modern mathematical logic has roots in this Aristotelian tradition as well (though it has roots elsewhere too, e.g. foundational work in mathematics). However, the Aristotelian tradition is beyond the expertise the authors, and so we have not attempted to cite relevant work from it.

⁸See Blanchette's ([2001](#)) and Shapiro's ([2005](#)) for an introduction.

⁹For example, we say in section [1.2.2](#) that formal languages can be thought of as models of natural languages; and much of the discussion on identity in section [8.3](#) raises issues as well.

Acknowledgments

This book began as a series of lecture handouts written for the yearly mathematical logic class in the philosophy department at Rice University. The handouts were primarily written by Prof. Richard Grandy from 2007 to 2011, with both Stan Husi and Jacob Mills (TAs for the class during that time) contributing. Michael Barkasi began the process of typesetting these handouts in \LaTeX for the fall of 2012 (at which time he was the TA). By next summer he finished this process, and over the next year he compiled and reorganized the handouts into a draft textbook, adding in material as needed. Joshua Reagan (TA from 2014-17) has been collaborating with Prof. Grandy since that time, significantly revising and adding new material to the text.

As with most other textbooks, no theorems are original and their presentation is, for the most part, what you'll find elsewhere. The proofs for all the theorems—along with the actual writing and presentation—have been done from scratch by one or more of the authors; but the general methods and approaches used are just those the authors learned from others. Thus this textbook owes just as much to previous work in logic as most other modern textbooks. It passes on to students a body of results and a general conceptual framework developed by many logicians over the years. It does so in roughly the style and presentation in which those results were passed to the authors themselves, with some changes they think are improvements. We will note these as they come up.

A few more specific acknowledgments are called for. At the time the first author was preparing the original handouts he was using Merrie Bergmann, James Moor, and Jack Nelson's textbook *The Logic Book* (2003); hence some of the terminology and organization in this book reflects that. Benson Mates' classic *Elementary Logic* (1972), the previous text, and Church's *Introduction to Elementary Logic* (1956) also influenced the first author's approach. The second author has relied heavily on Wilfred Hodges's rich survey article "Elementary Predicate Logic" (2001b) for historical citations. The name "Dragnet" for theorem 4.9 comes from Michael Smith, another former TA for the first author. The problems in exercise 5.3.3 (on page 124) comes from Howard Pospesel & David Marans, *Arguments: Deductive Logic Exercises* (1978), which Pospesel and Marans have generously released to the public.

Chapter 1

Introduction

1.1 What is Logic?

1.1.1 Rational Creatures

Humans are rational creatures. As such, we reason about our circumstances and the world in general so we may understand them better. An improved understanding of the world helps us make informed decisions, which (ideally) tend to bring about preferable outcomes. If you find yourself lost in a labyrinth and hunted by a hungry minotaur, clever reasoning could help you escape. Poor reasoning could get you eaten!

At least some of our reasoning is *discursive*. Discursive reasoning is an iterative process: start with a set of initial claims and then infer a result from them, perhaps repeating the process until a certain conclusion is reached. Such a chain of inferences can be written down or otherwise recorded as an argument.

Knowledge of logic helps us exploit a particularly reliable kind of discursive reasoning. It enables us to judge an argument from any source according to independent, principled criteria. Not only can we assess the strength of arguments presented to us; we can construct rigorous arguments of our own to share with others. Rationality thus has an important social aspect. By sharing arguments with each other we can, to some extent, coordinate beliefs and behaviors, and thereby complete tasks beyond the ability of any one person.

The social role of argumentation provides a clue about the subject matter of logic. Arguments are shared by way of language. One can give an argument by writing it on paper, typing it in an e-mail, stating it in a speech, etc. Once inscribed or encoded in one linguistic medium or another, it can then be assessed by others. Accordingly, logicians must attend to certain public features of language.

1.1.2 Logical Consequence

Logic is the study of *logical consequence* and *logical truth*. When we say that one sentence ‘logically follows’ from another, we mean that the first sentence is a logical consequence of the other. But how can we identify when some claim is a logical consequence of another? As suggested previously, we must attend to certain features of the language in which the argument is given.

One sentence ψ is a logical consequence of another sentence ϕ if and only if the truth of ϕ guarantees, in virtue of its logical structure, the truth of ψ .

Three points are worth making about this provisional definition. First, don't be afraid of the Greek letters: they're just variables for sentences. Second, the phrase 'if and only if' is one we use throughout the text, sometimes abbreviated as 'iff'. ' ϕ if and only if ψ ' is equivalent to 'If ϕ then ψ and if ψ then ϕ '. Third, a solid understanding of this definition depends on having a clear notion of *logical structure*. One of the central goals of this textbook is to provide such a notion.

Consider the following two sentences:

1. George Washington was in the Continental Army and Nathanael Green was in the Continental Army.
2. Nathanael Green was in the Continental Army.

The second sentence is a logical consequence of the first—if the first is true the second must be too. The next two sentences share the same structure as the last two:

3. Benjamin Franklin was a delegate to the Continental Congress and Thomas Jefferson was a delegate to the Continental Congress.
4. Thomas Jefferson was a delegate to the Continental Congress.

As above, the second sentence is a logical consequence of the first. We can show the common logical structure of the preceding pairs of sentences using a *schema*:

5. ϕ and ψ .
6. ψ .

We have replaced the non-logical content of each sentence with Greek letters and kept the logical word 'and'. The Greek letters serve as variables which stand for clauses of the English language. Any substitution of appropriate English clauses for the letters ϕ and ψ in the above schema generates a pair of sentences in which the latter follows from the former.¹

A sentence can also be the logical consequence of a set of sentences. For example, given the sentences,

7. The sun is shining.
8. The birds are chirping.

the following is a logical consequence:

9. The sun is shining and the birds are chirping.

We can represent the logical structure of these sentences in schematic form:

¹We will say more about what counts as an appropriate substitution later in the chapter.

- 10. ϕ .
- 11. ψ .
- 12. ϕ and ψ .

Any substitution of appropriate English clauses for the letters ϕ and ψ in this schema generates three sentences in which the last follows logically from the first two.

1.1.3 Logical Truth

A sentence is a *logical truth* if and only if the logical structure of the sentence guarantees its truth. For example,

- 13. If the sun is shining then the sun is shining.

This sentence must be true, regardless of whether the sun is shining. Contrast that with sentence 8; if no birds exist, then 8 can't be true. There is nothing special about the non-logical content of 13. We could replace 'the sun is shining' with 'it's raining outside' and get another logical truth:

- 14. If it's raining outside then it's raining outside.

The schema of these two logical truths is:

- 15. If ϕ then ϕ .

The non-logical content is replaced with ϕ and the words 'if...then' are retained. Any substitution of an appropriate clause of English for ϕ generates a logical truth. More generally, logically true sentences have a structure such that, whatever appropriate clause(s) we substitute for the non-logical parts, the result is also a logical truth.

To construct general schemas from particular examples, as we did above, requires distinguishing between the logical structure and the non-logical content of a sentence. We take for granted that certain "logical" words play a special role in constituting the logical structure of an English sentence—e.g., words such as 'and', 'or', 'if...then', 'not', 'all', 'some', among many others.

Logical consequence and logical truth are closely related concepts. In fact, each can be defined in terms of the other, though we won't specify the connection until we give a mathematically precise definition of each. Unfortunately, natural languages such as English have features that make these concepts difficult (or even impossible) to define adequately.

1.2 Natural and Formal Languages

1.2.1 Natural Languages

Calling English a *natural* language is only meant to indicate that it developed gradually and informally over time, and wasn't the product of, say, scholars officially defining the

grammar and vocabulary from scratch. English, Greek, German, Russian, Spanish, Mandarin, and Hindi are all natural in this sense.

Let's examine some of the obstacles to achieving mathematical precision in natural language. First, it's indeterminate which collections of words and letters are genuine sentences. The following strings are, perhaps, neither clearly sentences nor clearly not sentences of English:²

16. Colorless green ideas sleep furiously.
17. Furiously sleep ideas green colorless.
18. Seventeen is purple.
19. Green is a prime number.
20. Someone someone admires admires someone.

There are no principled, universally accepted rules that determine whether each is officially a sentence of English. Without a clear distinction between sentences and non-sentences, we cannot be certain which are appropriate for substitution in logical schemas like those given in the previous section.

A second problem with natural languages is that they contain paradoxical sentences. A *paradoxical* sentence is one that's true if and only if it's false. The most famous examples are those that engender the liar paradox, often called *liar* sentences.³ The following are two examples.

21. Sentence 21 is false.
22. The second sentence on this page with exactly twelve words is false.

Paradoxical sentences seem to be simultaneously both true and false. Try assuming that 21 is false, and it seems to follow that it's also true. Try assuming that 21 is true, and it comes out false. Both outcomes are impossible because they're each contradictory. Paradoxical sentences frustrate systematic and coherent logical analysis, so logicians prefer to exclude them altogether.

A third problem is that natural languages contain *ungrounded* sentences. The following pair is a good example:

23. Sentence 24 is true.
24. Sentence 23 is true.

These sentences seem to be unhinged from reality. We can assume consistently that both are true, and we can assume consistently that both are false, regardless of any substantive fact about the world. Most logicians prefer to exclude ungrounded sentences along with the paradoxical ones.

²Sentences 16 and 17 are from Chomsky 2002 [1957].

³See the following: Tarski 1983 [1933], 1944, Kripke 1975, Barwise and Etchemendy 1987, and Gupta 2001.

An important note is in order. Paradoxical and ungrounded sentences are different from *contradictory* (or “self-contradictory”) ones. The following is a contradiction, not a paradox:

25. The door is open and it isn’t open.

Unlike a paradoxical sentence, which seems to take both truth values, a contradictory sentence must be false. There is nothing wrong with contradictory sentences from a logician’s point of view. They are perfectly legitimate relatives of logical truths. Just as a logical truth must be true, a contradiction (i.e., a logical falsehood) must be false. Contradictory sentences are self-refuting, but they take a single truth value and so are acceptable for our purposes.

A fourth problem with natural language is that many, if not most, sentences of natural languages have ambiguous or context-dependent meanings. Consider the following.

- 26. Wealthy Americans flee south in record numbers.
- 27. We gave the bananas to the monkeys because they were hungry.
- 28. We gave the bananas to the monkeys because they were tasty.
- 29. We gave the bananas to the monkeys because they were there.
- 30. Paul insulted George because he thought he was someone else.
- 31. Sally’s car is red.
- 32. He is tall.
- 33. He is the tallest one here.
- 34. You are tall.
- 35. Sally went to the bank.

We can often resolve ambiguous sentence meaning by considering the particular circumstances in which a sentence is spoken or written. For instance, the ambiguity may come from an indexical term like ‘you’ (e.g., 34), in which case the intended referent is clear once we figure out who is speaking or writing to whom. Sometimes the meaning of a term is context-dependent, as with ‘tall’ (e.g., 32). Five feet would be tall for a seven-year old, but not for an adult. Other cases are more complicated. Does ‘Sally’s car’ in 31 mean the car Sally owns, leases, or something else? Is ‘red’ the color of the car’s exterior or interior? Is it light red, dark red, or something in between?

Pronoun reference can be difficult to resolve, even in context (e.g., ‘he’ in 30, 32, 33). Finally, sometimes a word in the sentence has multiple distinct meanings (e.g., ‘bank’ in 35) or the overall syntax of the sentence is ambiguous (e.g., 26–29). As a result the intended meaning of the sentence may not be recoverable from the context without additional information.

Ambiguous sentences can make careful logical analysis difficult or impossible. The following looks like a logical truth:

36. If she is tall then she is tall.

But what if the first ‘she’ refers to one person and the second refers to another?

1.2.2 Formal Languages

The logician avoids these difficulties by using carefully constructed formal languages.⁴ A *formal* language has the following features:

- (1) There is an explicit list of permitted symbols.
- (2) There is a definition of which strings of symbols count as sentences.

A *string* of symbols is just a row or sequence of symbols. For example, ‘fq3H7’ is a string consisting of ‘f’ followed by ‘q’, ‘3’, ‘H’, and ‘7’. The order of the characters is important; ‘f7q’ is not the same string as ‘7fq’.

Formal languages have a precisely defined syntax, but no fixed semantics. Definitions of sentences in formal languages make no reference to meanings of the sentences or their parts. These definitions make reference only to the *form*, or shape of the symbols and their arrangements in strings. We should think of the symbols and sentences of a formal language as lacking inherent meaning.

In a well-constructed formal language we can check whether any given string is a genuine sentence, solving the indeterminacy problem faced by natural languages. We also exclude the features of natural languages that lead to paradoxical and ungrounded sentences. Sentences of formal languages have no inherent meaning, but there are ways to assign meaning to them while avoiding the difficulties of natural language. In chapter 5 we provide methods of translating English sentences into constructed formal languages.

By turning to formal languages we are using a familiar (and often successful) strategy of replacing a hard messy problem with a more tractable mathematical one. For example, geometry doesn’t deal with points or lines in the physical world; the points and lines of geometry have zero area and zero width, respectively. Nevertheless, geometry provides useful models of the physical world when physical points and lines approximate geometrical ones sufficiently. When the logical features of our formal language sufficiently reflect the logical features of our natural language, mathematical idealization is a productive strategy.⁵

How do our formal languages relate to natural languages like English? We think of the formal languages as abstract models of important parts of their natural counterparts. We cover several formal languages in this text: SL (chapter 2), QL1 (chapter 3), QL (chapter 4), and later MSL and QLI (chapter 8), will serve as progressively

⁴Poorly (or deviously) constructed formal languages can have the same problems as natural ones.

⁵Historically the move to formal languages in the study of logic (and mathematics) has been fruitful. One example, important for the development of modern logic, comes from set theory. By using formal methods logicians found that the naïve axioms of set theory are inconsistent (see Demopoulos and Clark 2005 for a quick overview of Russell’s paradox, or Smullyan and Fitting 2010: ch 1 for a more careful discussion).

more detailed models of English. After learning how to define these formal languages and how to translate between them and English, the logic student should have a better understanding of the logical structure of English and of logical consequence more generally. The student should keep in mind that these formal languages cannot perfectly capture all the desirable logical features of English. There is a limit to both the range of English sentences they model and the accuracy with which they model them.

1.3 Some Distinctions

1.3.1 Use and Mention

There are three distinctions that are important in the modern study of logic. The first is the use/mention distinction. For any word or expression (string of words), we can either *use* the word or expression, or instead *mention* it. *Using* words and expressions is what we normally do, while *mentioning* a word or expression is one way to talk about that word or expression. For example, you could say (i) that a cow is a four-legged bovine, or you could instead say (ii) that ‘cow’ has three letters. In case (i) we are talking about a particular kind of animal, while in case (ii) we are talking about the *word* for that animal. In this textbook words that are mentioned are put in single quotes. Here are some examples:

- 37. Houston has over 2 million inhabitants. [True]
- 38. Houston has over 10 million inhabitants. [False]
- 39. ‘Houston’ has more than 2 million inhabitants. [False]
- 40. ‘Houston’ has more than 4 letters. [True]
- 41. Austin has more than 4 letters. [False]
- 42. ‘Austin’ has exactly 6 letters. [True]

1.3.2 Type and Token

The next distinction is the type/token distinction. How many letters does ‘Houston’ have? The answer depends on whether the asker means letter tokens or letter types. A *token* is a physical object or event, while a *type* is an abstract *kind* of physical object or event. Tokens are located in space and time, while types presumably are not. The word ‘Houston’ has 7 letter tokens and 6 letter types, because the letter ‘o’ occurs twice. In the sentence ‘Ron went on and on’ there are five word tokens and four word types.⁶ In ‘radar’ there are five letter tokens and three letter types.

⁶In the sentence ‘Ron went on and on’, there are three occurrences of the string ‘on’ with one of them in the name ‘Ron’. However, this instance doesn’t count as a word token in English, because it’s only a piece of a word, not a complete word. Other cases in English are more ambiguous. In the compound word ‘footnote’ does ‘note’ count as a word token?

1.3.3 Object Language and Metalanguage

The third distinction is that between an object language and a metalanguage. Returning to the use/mention distinction, if one mentions a word, then the *object language* is the language of the word being mentioned and the *metalanguage* is the language in which that word is mentioned. For example, if I say “The German word for dog is ‘Hund’,” I’ve used English (the metalanguage) to talk about German (the object language). In this text the metalanguage will be English plus some mathematical tools and symbols. The object language will be some formal language.

Usually there’s a little more to being a metalanguage than the mere fact that it’s the language used to talk about another. Often there’s some specific purpose. For example, the metalanguage might be used to give definitions of words in the object language. Or, as in our case, the metalanguage might be used to define when a sentence of the object language is true, or to describe when the logical consequence relationship holds between sentences of the object language.

1.4 MathEnglish

As stated previously, the various formal languages in this text are to be object languages. However, before we can define our first formal language (SL), we need some mathematical tools for our metalanguage. The metalanguage will contain a mix of English, additional symbols (e.g., the Greek letters), and other jargon. We will call our metalanguage *MathEnglish*.

1.4.1 Metavariables

One important tool for MathEnglish is the *metavariable*, which we define as a variable for strings of the object language. We will use lowercase Greek letters as our metavariables, as we did in the schemas given at the beginning of the chapter. The three we’ll use most commonly are ‘ ϕ ’ (phi), pronounced “fie”, ‘ θ ’ (theta), pronounced “theta”, and ‘ ψ ’ (psi), pronounced “sigh”. These Greek letters are not in the object language, but instead are used in the metalanguage for talking about the object language.

An example will make their role more clear. Let’s say that MathEnglish is our metalanguage and (regular) English is our object language.⁷ Recall the logically true sentence from the beginning of the chapter:

If the sun is shining then the sun is shining.

We noticed that we could replace ‘the sun is shining’ with any other assertion and get another logically true sentence. With this in mind, consider the following claim of MathEnglish:

⁷As discussed earlier, in future chapters we will use a formal language and not English as our object language, but for the purpose of illustration we temporarily ignore the problems of natural language.

All English sentences of the form ‘If ϕ then ϕ ’ are logical truths.

Here we have introduced the character ‘ ϕ ’ as a metavariable. The use of ‘ ϕ ’ helps us talk about *all* English sentences of a certain form as logically true, or at least all English sentences of a certain kind. As logicians we are concerned with truth and truth-preservation, so we are only going to address strings with plausible truth values, i.e. declarative sentences. A declarative sentence is one that makes an assertion. By restricting the substitutions to declaratives, we rule out substitutions such as ‘Ergle bergle barfle narfle,’ ‘nef34fwjh9ufh32,’ ‘Kentucky fried chicken,’ and ‘Are you cooking?’. We cannot legitimately affirm truth or falsity of these non-assertions. In the rest of the text, when we refer to ‘all’ sentences of a natural language we only mean the declarative ones but won’t repeat it explicitly.

In any case, metavariables are indispensable tools for making general claims about object languages.

1.4.2 Sets

MathEnglish also makes use of a set theory. A *set* is just some collection of items, which we’ll call *elements*. These items can be anything, like animals, people, planets, cartoon characters, or even abstract objects such as numbers. We can indicate a set using curly brackets, as in the following set of integers from 1 to 4:

$$\{1, 2, 3, 4\}$$

The order of elements in set notation doesn’t matter, so the following set is identical to the last:

$$\{2, 3, 1, 4\}$$

Furthermore, repetition of an element in set notation is to be disregarded—an element can only be in the set once. Accordingly, the following sets are actually just the same set as the last two:

$$\begin{aligned} &\{1, 2, 2, 3, 3, 3, 4, 4, 4, 4\} \\ &\{2, 2, 3, 1, 4, 2, 3, 1, 4, 4, 3\} \end{aligned}$$

Sets don’t have to be finite. Consider the set of positive integers:

$$\{1, 2, 3, 4, \dots, 1002, 1003, 1004, \dots\}$$

There is one set that doesn’t have any elements, and it’s called either the *null set* or the *empty set*. Symbolically we can express the empty set as either $\{\}$ or \emptyset .

To express set membership, we use the ‘ \in ’ symbol. Let the Greek letters Δ and Γ stand for some arbitrary sets. (They are pronounced ‘delta’ and ‘gamma’, respectively.) We may assert that 7 is an element of set Δ and 3 is an element of Γ as follows: $7 \in \Delta$, $3 \in \Gamma$. On this notation,

$$2 \in \{1, 2, 3\}$$

is true and,

$$4 \in \{1, 2, 3\}$$

is false.

We use lowercase Greek letters α and β (pronounced ‘alpha’ and ‘beta’, respectively) as metavariables for objects. For example, we can affirm that if α and β are odd numbers and Δ is the set of even numbers, $\alpha + \beta \in \Delta$.

Definition 1.1. For any two sets Δ and Γ , Δ is a *subset* of Γ iff for each α such that $\alpha \in \Delta$, $\alpha \in \Gamma$.

For example, $\{1, 2, 3\}$ is a subset of $\{1, 2, 3, 4\}$. The set $\{1, 2\}$ is a subset of each of the two previous sets. We use the symbol ‘ \subseteq ’ to stand for the subset relation. So when Δ is a subset of Γ , $\Delta \subseteq \Gamma$. From the definition of subset it follows that every set is a subset of itself.

Definition 1.2. For any two sets Δ and Γ , Δ is a *proper subset* of Γ iff

- (1) $\Delta \subseteq \Gamma$ and
- (2) there is some α such that $\alpha \notin \Delta$ and $\alpha \in \Gamma$.

As you can probably guess, the ‘ \notin ’ symbol means ‘is not an element of’. The symbolic notation for ‘ Δ is a proper subset of Γ ’ is $\Delta \subset \Gamma$. No set is a proper subset of itself.

Take care to avoid confusing the membership, subset, and proper subset relations.

- 43. $2 \in \{1, 2, 3\}$ [True]
- 44. $2 \subseteq \{1, 2, 3\}$ [False]
- 45. $\{2, 3\} \subseteq \{1, 2, 3\}$ [True]
- 46. $\{2, 3\} \in \{1, 2, 3\}$ [False]
- 47. $\{2, 3\} \subset \{1, 2, 3\}$ [True]
- 48. $\{1, 2, 3\} \subset \{1, 2, 3\}$ [False]

1.4.3 More on Sets: Union and Intersection

Sometimes we’ll want to define a set by reference to other sets. Say we have two sets, Γ_1 and Γ_2 , and there is a third set, Δ , that contains all the members of the first two, but nothing else. Under these conditions we say that Δ is the *union* of Γ_1 and Γ_2 . Let’s give a strict definition:

Definition 1.3. $\Delta = \Gamma_1 \cup \Gamma_2$ iff

- (1) for each $\alpha \in \Gamma_1$, $\alpha \in \Delta$ and

- (2) for each $\alpha \in \Gamma_2$, $\alpha \in \Delta$ and
- (3) for each $\alpha \in \Delta$, $\alpha \in \Gamma_1$ or $\alpha \in \Gamma_2$ (or both).

The symbol ‘ \cup ’ means ‘union’. Some examples:

$$\{\text{Mercury, Jupiter, Mars, Saturn}\} = \{\text{Mercury, Jupiter}\} \cup \{\text{Mars, Saturn}\}.$$

$$\{\text{Newton, Bacon, Boyle}\} = \{\text{Newton, Bacon}\} \cup \{\text{Bacon, Boyle}\}.$$

Another useful concept is *intersection*. The set Δ is the intersection of two sets Γ_1 and Γ_2 iff Δ contains all the elements shared by both Γ_1 and Γ_2 but nothing else. More strictly:

Definition 1.4. $\Delta = \Gamma_1 \cap \Gamma_2$ iff

- (1) for each α such that $\alpha \in \Gamma_1$ and $\alpha \in \Gamma_2$, $\alpha \in \Delta$ and
- (2) for each $\alpha \in \Delta$, $\alpha \in \Gamma_1$ and $\alpha \in \Gamma_2$.

The symbol ‘ \cap ’ means ‘intersection’. The intersection of $\{\text{Mercury, Jupiter}\}$ and $\{\text{Mars, Saturn}\}$ is the empty set, $\{\}$, because the former two sets have no members in common.

$$\{\text{Mercury, Jupiter}\} \cap \{\text{Mars, Saturn}\} = \{\}.$$

By contrast, the intersection of $\{1, 5, 6, 9\}$ and $\{2, 3, 5, 6\}$ is $\{5, 6\}$.

$$\{1, 5, 6, 9\} \cap \{2, 3, 5, 6\} = \{5, 6\}.$$

For another example:

$$\{\text{Carnap, Quine}\} = \{\text{Anscombe, Quine, Carnap}\} \cap \{\text{Carnap, Lewis, Quine}\}.$$

1.4.4 Ordered Pairs and Ordered n-tuples

What if we want to describe a collection in which, unlike sets, the order does matter? For that purpose we have *ordered pairs* and *ordered n-tuples*. An ordered pair is a two-place collection in which the order matters. To differentiate ordered pairs from sets, we indicate them using angled brackets rather than curly brackets. So, while the sets $\{1, 2\}$ and $\{2, 1\}$ are identical, the ordered pairs $\langle 1, 2 \rangle$ and $\langle 2, 1 \rangle$ are not.

If we want to put together an ordered collection having more than two places, we can use an ordered n -tuple, where n is the number of places needed. So, $\langle \text{Mercury, Venus, Earth} \rangle$ is a 3-tuple, $\langle \text{Jupiter, Saturn, Uranus, Neptune} \rangle$ is a 4-tuple, and so on.

Also unlike sets, for ordered pairs and n -tuples, repetition makes a difference. For example, $\langle 1, 2, 3 \rangle$ is different from $\langle 1, 2, 3, 3, 3 \rangle$.

1.4.5 Mathematical Proofs

We use proofs to establish that something has certain mathematical properties. A proof works from one or more definitions to some result, which we sometimes call a *theorem*. If we want to prove an intermediate result that isn't by itself particularly interesting but which will be handy to have for one or more later proofs, we sometimes call it a *lemma*.

Let's illustrate with an easy proof.

Theorem 1.5. *If $\Delta_1 \subseteq \Delta_2$ and $\Delta_2 \subseteq \Delta_3$ then $\Delta_1 \subseteq \Delta_3$.*

Proof: This is a conditional statement. To prove statements like these, assume first part and try to show that the second part follows. In this case we assume that there are some sets Δ_1 , Δ_2 , and Δ_3 such that $\Delta_1 \subseteq \Delta_2$ and $\Delta_2 \subseteq \Delta_3$.

Next we use the definition of subset to show that $\Delta_1 \subseteq \Delta_3$ follows from the assumption. By the definition of \subseteq : Any α in Δ_1 is also in Δ_2 , and any α in Δ_2 is also in Δ_3 . It's clear that if $\alpha \in \Delta_1$ then $\alpha \in \Delta_3$. Thus, by the definition of \subseteq , $\Delta_1 \subseteq \Delta_3$. ■

We have proved that ' \subseteq ' is transitive. Having achieved this result, we permit ourselves to use it throughout the rest of the text without proving it again. The end of a completed proof is indicated with a black box: ■.

Later in the book we will show how to prove more complicated (and more interesting) theorems.

1.4.6 Recursive Definitions

In later chapters we use MathEnglish to define concepts such as *sentence*, *truth*, and *entailment* for a given object language. Many of these definitions are recursive. Recursive definitions can be understood as defining which objects are in a given set.

Definition 1.6. *A recursive definition of Δ has three clauses:*

- (1) the *base clause(s)*, which specifies one or more objects which are elements of Δ ,
- (2) the *generating clause(s)*, which specifies one or more ways of generating (or finding) other objects that are elements of Δ , and
- (3) the *closure clause*, which specifies that something is in Δ only if it can be shown to be so by applications of the first two clauses.

One concept that we can define recursively is *natural number*.

Definition 1.7. The set \mathbb{N} of natural numbers:

- (1) Base Clause: Let 0 be a natural number. I.e. $0 \in \mathbb{N}$.
- (2) Generating Clause: If $n \in \mathbb{N}$, then $n + 1 \in \mathbb{N}$.
- (3) Closure Clause: Nothing else is in \mathbb{N} .

This definition unambiguously identifies infinitely many numbers as natural numbers. Obviously 0 is a natural number. Is 7 a natural number? It is. We know 0 is a natural number because the base clause says so. Thus, according to the generating clause, $0 + 1$ (i.e., 1) is also a natural number. Apply the generating clause again to show that $1 + 1$ (i.e., 2) is natural number. One can continue applying the generating clause until 7 is reached. Is -3 a natural number? No. -3 is less than 0 and the generating clause only defines larger numbers as natural. The closure clause rules out the inclusion of anything else.

Let's use a recursive definition to identify which people are ancestors of French mathematician Blaise Pascal.

- (1) Base Clause: Blaise Pascal's mother and father are ancestors of Blaise Pascal.
- (2) Generating Clause: If x is an ancestor of Blaise Pascal and y is a parent of x , then y is also an ancestor of Blaise Pascal.
- (3) Closure Clause: No one else is an ancestor of Blaise Pascal.

This definition picks out the mother and father of Blaise Pascal, their respective mothers and fathers, the mothers and fathers of the latter, and so on. It excludes everyone else. Recursive definitions are powerful, because in a few lines we can fix an unambiguous collection that is arbitrarily large. The closure clause is usually relatively trivial (e.g., "Nothing else is an x ").

1.4.7 Conclusion

Now we have most of the mathematical tools we need to define our first formal language, *Sentential Logic*. Anything else we need will be given along the way.

1.5 Exercises

1.5.1 Sets

1. True or false: $\{2, 3\} \subseteq \{1, 2, 3\}$
2. True or false: $\{1, 2, 3\} \subset \{1, 2, 3\}$
3. True or false: $\{2, 3\} \in \{1, 2, 3\}$
4. True or false: $\{2, 3\} \in \{1, \{2, 3\}\}$
5. True or false: $2 \in \{1, \{2, 3\}\}$
6. True or false: $\{1, 2, 3\} = \{1, \{2, 3\}\}$
7. True or false: $\{1, 2, 3\} = \{3, 3, 1, 2, 3, 2, 2, 1\}$

Chapter 2

Sentential Logic

2.1 The Language SL

2.1.1 Sentences of SL

Our first task is to define the syntax of the basic formal language SL.¹ The language we're starting with is variously called *sentential logic* or *propositional logic*. We use the former name, because it's easier to get a grasp on what a sentence is. What a proposition is is a matter of intense philosophical debate.²

As stated in the previous chapter, each formal language requires (1) a list of basic symbols, and (2) a definition of which sequences of those symbols count as sentences. Think of these as determining the proper grammar of the language.

Definition 2.1. The *basic symbols* of SL are of three kinds:³

- (1) Logical Connectives: $\sim, \wedge, \vee, \rightarrow, \leftrightarrow$
- (2) Punctuation Symbols: $(,)$
- (3) Sentence Letters: $A, B, \dots, T, A_1, B_1, \dots, T_1, A_2, B_2, \dots$

Logical connectives are sometimes called logical symbols, logical operators, logical terms, or logical functors.⁴ One could draw important distinctions between symbols, terms, and operators, but the names are just as often used interchangeably. The *sentence letters* are italicized capital letters of the Roman alphabet from 'A' to 'T'. To give ourselves an infinite supply, any of these letters concatenated with a subscripted positive integer is also a sentence letter; e.g. C and a subscripted 7 can be combined to get the new sentence letter C_7 . Keep in mind that C_7 and C are different sentences.

¹Work on this sentential logic originated with George Boole (1854) and Augustus De Morgan (1847; 1860). As Christine Ladd-Franklin notes (1883: 17) before giving her own, variations quickly followed from William S. Jevons, Ernst Schröder, Hugh McColl, and Charles S. Peirce.

²For historical and contemporary discussions of propositions, see Frege 1892, Russell 1996 [1903]: 13,47, Church 1956: 26, Quine 1986: ch. 1, Schiffer 1987: ch. 3, Grandy 1993, Bealer 1998, King 2007, Soames 2010.

³The commas and ellipses are *not* symbols of SL.

⁴In other textbooks there are sometimes different symbols used for these connectives. Along with \sim, \neg and $-$ are used for negation, $\&$ and \cdot for conjunction, \supset and \Rightarrow for conditionals, and \equiv for biconditionals. \vee is almost universally the symbol for disjunction.

Definition 2.2. The *sentences of SL* are given by the following recursive definition:

Base Clause: Every sentence letter is a sentence.

Generating Clauses:

- (1) If ϕ is a sentence, then so is $\sim\phi$.⁵
- (2) If ϕ and θ are sentences, then so are both $(\phi \rightarrow \theta)$ and $(\phi \leftrightarrow \theta)$.
- (3) If all of $\phi_1, \phi_2, \dots, \phi_n$ are sentences (where n is an integer ≥ 2), then so are $(\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n)$ and $(\phi_1 \vee \phi_2 \vee \dots \vee \phi_n)$.

Closure Clause: No string is an SL sentence unless it counts as such according to the above base and generating clauses.

The sentences specified by the base clause, i.e. lone sentence letters, are called *atomic* sentences. In generating clause (2), sentences of the first form are called *conditionals*, and those of the second are *biconditionals*. The left-hand side of the conditional is traditionally called the *antecedent* and the right-hand side the *consequent*. We often use the alternative terminology LHS (left-hand side) and RHS (right-hand side).

Generating clause (3) makes our version of SL slightly different from what you'll find in other logic texts.⁶ Sentences of the first form of clause (3) are called *conjunctions* and their component sentences *conjuncts*. Sentences of the second form are called *disjunctions* and their component sentences *disjuncts*. Many logic books treat conjunction and disjunction as binary (2-place), e.g., ' $(A \vee B)$ '. According to our clause (3) ' $(A \vee B \vee C)$ ' is also a perfectly good sentence. Textbooks that treat disjunctions as binary require an extra set of parentheses to generate an equivalent sentence: e.g., ' $((A \vee B) \vee C)$ '.⁷ This is also an acceptable sentence of SL, but the extra parentheses don't add interesting information. We prefer the definitions of conjunction and disjunction given above because they're closer to natural English and they allow us to avoid unnecessary parentheses.

The closure clause excludes strings that the other clauses don't define as sentences. This is needed to prevent nonsense such as ' $(B \rightarrow A$ ' from counting as an SL sentence. There is no way to generate the string from repeated applications of the base and generating clauses, so it is ruled out.

Additionally, ' $\sim\phi$ ', ' $(\phi \rightarrow \theta)$ ', etc. in the generating clauses are *sentence schemas*. They aren't SL sentences because the Greek letters aren't given as symbols of the language. The substitution of an SL sentence for each Greek letter in a schema results in a SL sentence, however. For example, the substitution $\phi = 'A'$, $\theta = '(C \leftrightarrow D)'$ into

⁵Remember from Chapter 1 that ϕ and θ are used as metavariables. In this definition they stand for sentences of SL.

⁶Some linguists have used the definition we use, including Gleitman 1965.

⁷We'll explain which sentences count as 'equivalent' later in the chapter.

the schema ' $(\phi \rightarrow \theta)$ ' results in the sentence ' $(A \rightarrow (C \leftrightarrow D))$ '.⁸

Example 2.3. $((A \vee (D \rightarrow B)) \wedge \sim G)$ is a sentence of SL. Proof: The base clause of definition 2.2 shows that A , D , B , and G are all sentences; they are all sentence letters. From D and B , and by generating clause (2), we know $(D \rightarrow B)$ is a sentence. From G and generating clause (1), $\sim G$ is a sentence. From A and $(D \rightarrow B)$, and generating clause (3), $(A \vee (D \rightarrow B))$ is a sentence. And, finally, from $(A \vee (D \rightarrow B))$ and $\sim G$, and generating clause (3), we see that $((A \vee (D \rightarrow B)) \wedge \sim G)$ is a sentence.

Unlike English, for any sequence of SL symbols it is possible to *prove* whether it is a sentence.

2.1.2 Official and Unofficial Sentences of SL

Definition 2.2 is the definition of an *official* sentence of SL. For convenience' sake we often work with *unofficial* sentences.

Definition 2.4. A string of symbols is an *unofficial* sentence iff we can obtain it from an official sentence by

- (1) deleting outer parentheses, or
- (2) replacing one or more pairs of official round parentheses () with square brackets [] or curly brackets { }.

Thus ' $A \wedge B \wedge C$ ' is an unofficial sentence, as are

- | | |
|---|---------------------------------|
| 1. $\sim([A \wedge B] \vee [C \wedge D])$ | 5. $\sim A \vee C$ |
| 2. $(A \wedge B) \wedge C$ | 6. $[(A \wedge B) \wedge C]$ |
| 3. $(A \wedge B) \rightarrow C$ | 7. $\{A \wedge B\} \wedge C$ |
| 4. $(A \wedge [B \rightarrow C])$ | 8. $A \wedge [B \rightarrow C]$ |

From an unofficial sentence we can unambiguously reconstruct the related official sentence. Throughout the rest of this text we usually drop outer parentheses, but we will consistently use the standard parentheses (). The reader should feel free to use brackets [] or curly parentheses { } as is helpful.

2.1.3 A Comment on Use and Mention

Most of the time when you see Greek letters in the text, as in definition 2.2 (on the previous page), we are *using* them, not mentioning them. Thus it's appropriate that in definition 2.2 we did not put them in single quotes. By contrast, we generally *mention* official and unofficial SL sentences rather than use them. Because we mention SL

⁸Often when Greek letters ' ϕ ', ' ψ ', ' θ ', etc. are used it will tacitly be assumed that they are ranging over SL sentences and not mere strings of SL symbols. Of course, we can still use them when needed to range over all strings of SL symbols as we did in the definition just given of SL sentences.

sentences so often it would be tedious to put them in quotes. Therefore we refrain from doing so unless there is some special reason to do so. We are also less strict when mentioning the basic symbols of SL.⁹

2.1.4 Other Properties of Sentences

Next we define four important properties and related features of sentences: subsentence, order, main connective, and construction tree.

Definition 2.5. The following clauses define when one sentence is a *subsentence* of another:

- (1) Every sentence is a subsentence of itself.
- (2) ϕ is a subsentence of $\sim\phi$.
- (3) ϕ and θ are subsentences of $(\phi \rightarrow \theta)$ and $(\phi \leftrightarrow \theta)$.
- (4) Each of $\phi_1, \phi_2, \dots, \phi_n$ is a subsentence of $(\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n)$ and $(\phi_1 \vee \phi_2 \vee \dots \vee \phi_n)$.
- (5) (Transitivity) If ϕ is a subsentence of θ and θ is a subsentence of ψ , then ϕ is a subsentence of ψ .
- (6) That's all.

Example 2.6. The sentence $(B \rightarrow C)$ has 3 subsentences:

- (1) $(B \rightarrow C)$
- (2) B
- (3) C

Subsentences are counted by token, not type. Hence the similar $(B \rightarrow B)$ also has three subsentences; the two tokens of B are counted separately.

Example 2.7. $(A \vee (D \rightarrow B)) \wedge \sim G$ has 8 subsentences:

- | | |
|--|--------------|
| (1) $(A \vee (D \rightarrow B)) \wedge \sim G$ | (5) D |
| (2) $A \vee (D \rightarrow B)$ | (6) B |
| (3) $D \rightarrow B$ | (7) $\sim G$ |
| (4) A | (8) G |

Definition 2.8. A sentence ϕ is a *proper subsentence* of ψ iff ϕ is a subsentence of, but isn't identical to ψ .

⁹This is the usual convention, e.g. [Hodges 2001b](#): 7.

Thus, while each sentence is a subsentence of itself, no sentence is a proper subsentence of itself.

Definition 2.9. The following clauses define the *order* of every SL sentence.¹⁰ Let $\text{ORD}\phi$ be the order of ϕ . Then:

- (1) If ϕ is an atomic sentence (a sentence letter), then $\text{ORD}\phi = 1$.
- (2) For any sentence ϕ , $\text{ORD}\sim\phi = \text{ORD}\phi + 1$.
- (3) For any sentences ϕ and θ , $\text{ORD}(\phi \rightarrow \theta)$ is one greater than the max of $\text{ORD}\phi$ and $\text{ORD}\theta$. Likewise, $\text{ORD}(\phi \leftrightarrow \theta)$ is one greater than the max of $\text{ORD}\phi$ and $\text{ORD}\theta$.
- (4) For any sentences ϕ_1, \dots, ϕ_n , $\text{ORD}(\phi_1 \wedge \dots \wedge \phi_n)$ is one greater than the max of $\text{ORD}\phi_1, \dots, \text{ORD}\phi_n$.
- (5) For any sentences ϕ_1, \dots, ϕ_n , $\text{ORD}(\phi_1 \vee \dots \vee \phi_n)$ is one greater than the max of $\text{ORD}\phi_1, \dots, \text{ORD}\phi_n$.
- (6) That's all.

Example 2.10. We'll give the order for all the subsentences of $(A \vee (D \rightarrow B)) \wedge \sim G$. The order of an atomic sentence is 1, so A, D, B, G each have order 1. The order of $\phi \rightarrow \psi$ is 1 plus the maximum of the orders of ϕ and ψ ; thus the order of $D \rightarrow B$ is 2. Because $\text{ORD}\sim\phi = \text{ORD}\phi + 1$, $\text{ORD}(\sim G) = \text{ORD}(G) + 1 = 2$. Because the order of a disjunction $\phi_1 \vee \dots \vee \phi_n$ is 1 plus the maximum order of the disjuncts, the order of $A \vee (D \rightarrow B)$ is 3. Similarly for conjunctions, the order of $(A \vee (D \rightarrow B)) \wedge \sim G$ is 4.

Definition 2.11. The *main connective* is the connective token (or tokens) that occur(s) in the sentence but in no proper subsentence.

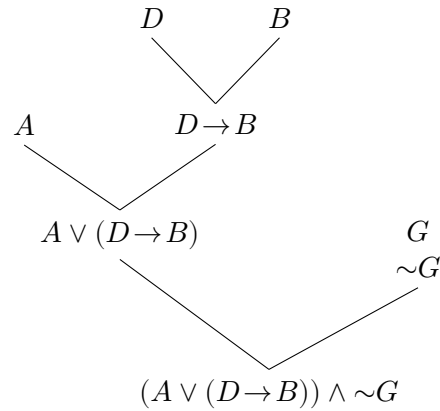
Example 2.12. The main connective in each of the following sentences has been underlined.

- | | |
|---|---|
| (1) $(A \vee (D \rightarrow B)) \underline{\wedge} \sim G$ | (4) $\underline{\simeq}(L \vee K \vee H)$ |
| (2) $L \underline{\vee} K \underline{\vee} H$ | (5) $((\underline{((D \rightarrow E) \vee A)} \underline{\wedge} (\sim B \wedge \sim(C \leftrightarrow H))))$ |
| (3) $L \underline{\vee} (A \rightarrow B) \underline{\vee} H$ | (6) $(\sim B \underline{\wedge} \sim(C \leftrightarrow H))$ |

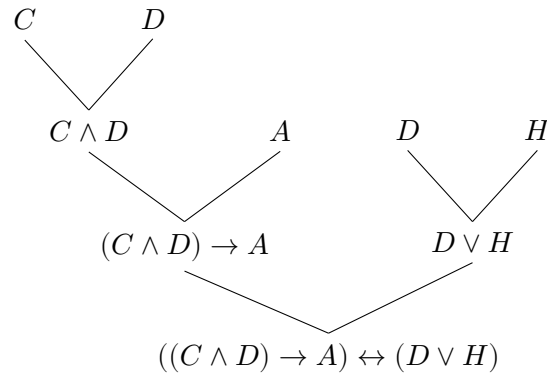
Definition 2.13. The *construction tree* for a sentence is a diagram of how the sentence is generated through the recursive clauses of the definition of SL sentences. We put atomic sentences as leaves at the top, and the generating clauses specify how we can join nodes of the tree together (starting with the leaves at the top) into new nodes. The complete sentence is the node at the base of the tree.

Example 2.14. Give the construction tree for $(A \vee (D \rightarrow B)) \wedge \sim G$.

¹⁰(Post 1921, Hodges 2001b: 11)

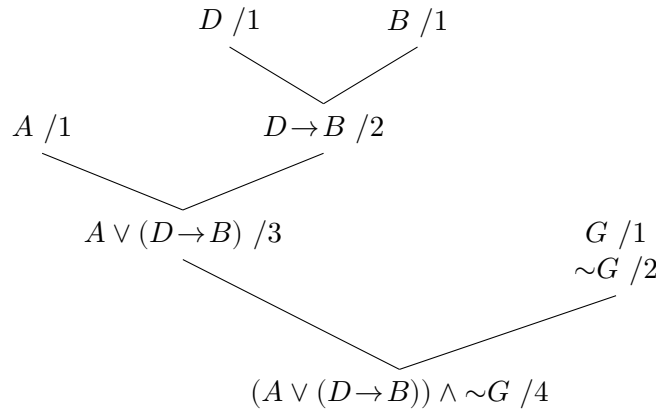


Example 2.15. Give the construction tree for $((C \wedge D) \rightarrow A) \leftrightarrow (D \vee H)$.



There are some helpful relationships between the construction tree of a sentence, its order, its subsentences, and its main connective. The subsentences of a sentence are the nodes in the sentence's construction tree. The order of a sentence is the number of nodes of its longest branch from root to leaf, i.e., the height of the tree. The main connective of a sentence is the connective added last (at the bottom) of the construction tree.

Example 2.16. Consider again the construction tree for $(A \vee (D \rightarrow B)) \wedge \sim G$. Find the order of the sentence by counting the height of the branches of the tree. (We count up as we work our way down the branches.) Note that the answer we get agrees with that computed in example 2.10 (on the previous page).



2.1.5 How Many SL Sentences are There?

There are at least as many sentence letters as there are natural numbers (countably infinite), and the sentence letters are a proper subset of the set of sentences. Are there more sentences than natural numbers? Below we prove there are not by matching up sentences with natural numbers.

Theorem 2.17. *The number of SL sentences is equal to the number of natural numbers.*

Proof: First, assign each sentence letter a natural number that only contains the digit ‘1’, for example:

A	B	C	D	...
1	11	111	1111	...

Next, assign numbers to the other symbols of SL, for example:

\sim	\wedge	\vee	\rightarrow	\leftrightarrow	$($	$)$
2	3	4	5	6	7	8

Given any sentence, replace its symbols with the associated numbers. For example, $\sim(A \wedge B \wedge \sim D)$ gets mapped to 2713113211118. For any sentence of SL, there is a unique natural number defined by this process. For any natural number we can determine if it represents an SL sentence, and if so which one. ■

This is not the most efficient way of representing sentences with numbers, but it is a simple one that avoids the use of special properties (e.g., being a prime number).

2.2 Models

SL is a formal language and as was mentioned in section 1.2.2 sentences of formal languages are mere strings of symbols. Because they lack inherent meanings—they don’t “say anything” about the world—sentences of SL lack a truth value. That

is, they are neither true nor false. Even so nothing stops us from *assigning* truth values to SL sentences. In this section we explain how to do that consistently. You might ask why we don't first assign SL sentences meanings, then determine whether they are true or false based on those meanings. There are difficulties associated with assigning meanings that would complicate our project unnecessarily.¹¹ The point of giving formal definitions of an SL sentence and model is to provide a rigorous basis for studying logical truth and logical consequence. One of the most important discoveries in logic was that for SL only truth values matter; all other details of meaning are irrelevant.

2.2.1 Truth in a Model

The truth value of a sentence depends on a model.

Definition 2.18. A *model of a SL sentence ϕ* is an assignment of a truth value, either true or false, to each sentence letter in ϕ .

In mathematical terms, a model of ϕ is a function from the set of sentence letters of ϕ to the set of truth values: $\{T, F\}$. We use the letter ' \mathbf{m} ' (a fraktur-style ' \mathbf{m} ') as the symbol for a model. If a model \mathbf{m} assigns a sentence letter ψ the value true, we write $\mathbf{m}(\psi) = T$. For the value false, we instead write $\mathbf{m}(\psi) = F$.

To illustrate, a model for $(A \vee B \vee C)$ must make a truth value assignment to each of the sentence letters A , B , and C . Any model for $(A \vee B \vee C)$ is therefore also a model for $(A \wedge B \wedge C)$ and $((A \rightarrow B) \wedge \sim C)$; they all have the same sentence letters.

We often speak informally of *models* without making reference to any particular SL sentence. It's useful to talk this way because any given model of ϕ is a model of any other SL sentence with the same sentence letters, or a subset of them. If \mathbf{m} makes assignments to A , B , and C , then \mathbf{m} is a model of all the sentences that only contain sentence letters from that list. Accordingly, we define a model for a *set* of SL sentences.

Definition 2.19. \mathbf{m} is a *model of a set of sentences Δ* iff \mathbf{m} is a model for each sentence in Δ .

Consider a model \mathbf{m} that makes a truth value assignment to every sentence letter of SL. No sentence letter lacks an assignment, so it follows that \mathbf{m} is a model for the set of all SL sentences. The following definition characterizes such models as *models of SL*.

Definition 2.20. \mathbf{m} is a *model of SL* iff \mathbf{m} is a model of every sentence of SL.

Models can be uniform; for example, there is a model that assigns true to every sentence letter. Or they can be systematic and mildly complicated, such as the model that alternately assigns true or false to a list of sentence letters. A model can assign truth values in any other pattern you can think of, or even random assignments. *Every* possible function from sentence letters to truth values is a model.

¹¹Later we will *interpret* SL sentences as having meanings (chapter 5).

Our models assign one of two truth values to each sentence letter, but that's not the only way to define them. Other definitions of 'model' assign more than two. Our assumption that there are only two truth values simplifies analysis and is widely shared, but whether two is enough is a matter of intense philosophical debate. Nevertheless, even if our definition of a model is a simplification, it is a historically fruitful one and helps us better understand logical consequence. In chapter 8 we discuss formal languages with additional truth values.

A model of ϕ only assigns truth values to the sentence letters of ϕ . It does not directly assign a truth value to any of the non-atomic sentences of SL. Thus, we must specify how each of the logical connectives operates on lower-order truth values in order to fix a truth value, in some given model, for non-atomic SL sentences. While the truth assignments to the sentence letters vary by model, the truth functions of the connectives do not.¹²

Definition 2.21. The following clauses define whether an SL sentence θ is *true* or *false* on a model \mathbf{m} for θ . The relevant clause is determined by which main connective θ has, if any:

- (1) θ is a sentence letter. θ is true on \mathbf{m} iff \mathbf{m} assigns true to it, i.e. $\mathbf{m}(\theta) = \top$.
- (2) θ is of the form $\sim\phi$ (i.e. is a negation). θ is true on \mathbf{m} iff ϕ is false on \mathbf{m} .
- (3) θ is of the form $(\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n)$ (i.e. is a conjunction). θ is true on \mathbf{m} iff each of the conjuncts $\phi_1, \phi_2, \dots, \phi_n$ is true on \mathbf{m} .
- (4) θ is of the form $(\phi_1 \vee \phi_2 \vee \dots \vee \phi_n)$ (i.e. is a disjunction). θ is true on \mathbf{m} iff at least one of the disjuncts $\phi_1, \phi_2, \dots, \phi_n$ is true on \mathbf{m} .
- (5) θ is of the form $\phi \rightarrow \psi$ (i.e. is a conditional). θ is true on \mathbf{m} iff the LHS ϕ is false or the RHS ψ is true on \mathbf{m} (or both).
- (6) θ is of the form $\phi \leftrightarrow \psi$ (i.e., is a biconditional). θ is true on \mathbf{m} iff ϕ and ψ have the same truth value on \mathbf{m} .
- (7) A sentence is false on \mathbf{m} iff it's not true on \mathbf{m} .

For every model \mathbf{m} for sentence ϕ , the above definition (i.e., the definition of truth) fixes a unique truth value for ϕ .¹³

Although there are an infinite number of SL sentences letters to which a model can assign truth values, only what the model assigns to a finite number of those sentence letters matters for determining the value of any given sentence. Unsurprisingly, only sentence letters that appear in the sentence matter. (We prove this later in the chapter.) For example, when assessing the value of $A \rightarrow B$ on \mathbf{m} only the assignments to A and B are relevant; other assignments may be disregarded. It follows that if two models \mathbf{m}_1 and \mathbf{m}_2 assign the same truth values to A and B , respectively, then they fix the same truth value for $A \rightarrow B$.

¹²We discuss *truth functions* further in section 2.2.2.

¹³If a model \mathbf{m} is *not* a model for some SL sentence ϕ , then \mathbf{m} does *not* fix a truth value for ϕ .

If \mathbf{m} makes assignments to A and B but no other sentence letters, we say that \mathbf{m} is a *minimal model* for $A \rightarrow B$. More generally:

Definition 2.22. \mathbf{m} is a *minimal model* of ϕ iff \mathbf{m} makes assignments to every sentence letter in ϕ but to no other sentence letters.

There are only four minimal models for the sentence $A \rightarrow B$, because there are only 4 combinations of truth values that can be assigned to A and B . The number of distinct minimal models for a sentence ϕ is 2^n , where n is the number of sentence letter (types) in ϕ .

There are several different ways to compute the truth value of an SL sentence in a model. We demonstrate some informal ones in the following examples, and then develop a systematic method in section 2.2.4 (on page 28).

Example 2.23. Give the truth value of $A \vee \sim B$ on a model \mathbf{m} such that $\mathbf{m}(A) = \text{F}$ and $\mathbf{m}(B) = \text{F}$.

One way to compute the truth value of this sentence in \mathbf{m} is to read off the values of all the subsentences, using definition 2.21 (the definition of truth in SL), until we finally get to the sentence itself.

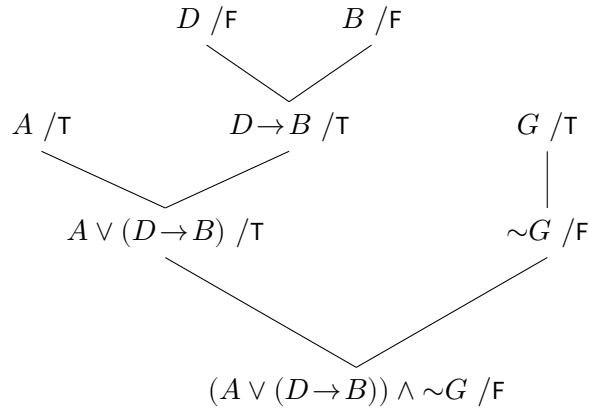
Proof: From the negation clause of the definition of truth and the fact that $\mathbf{m}(B) = \text{F}$, it follows that $\sim B$ is true on \mathbf{m} . According to the disjunction clause of the definition of truth $A \vee \sim B$ is true on \mathbf{m} iff at least one of the disjuncts is true, i.e., A and $\sim B$. So because $\sim B$ is true on \mathbf{m} , $A \vee \sim B$ is too. ■

Example 2.24. Give the truth value of $(A \vee (D \rightarrow B)) \wedge \sim G$ on a model \mathbf{m} such that $\mathbf{m}(A) = \text{F}$, $\mathbf{m}(D) = \text{T}$, $\mathbf{m}(B) = \text{T}$, and $\mathbf{m}(G) = \text{T}$.

When computing the truth value for a complicated sentence it can help to be strategic about which subsentences to look at first. Often you don't need to determine the value of every subsentence. The main connective can be an important clue about where to start. The sentence $(A \vee (D \rightarrow B)) \wedge \sim G$ is a conjunction, which is false on a model if any one of its conjuncts is.

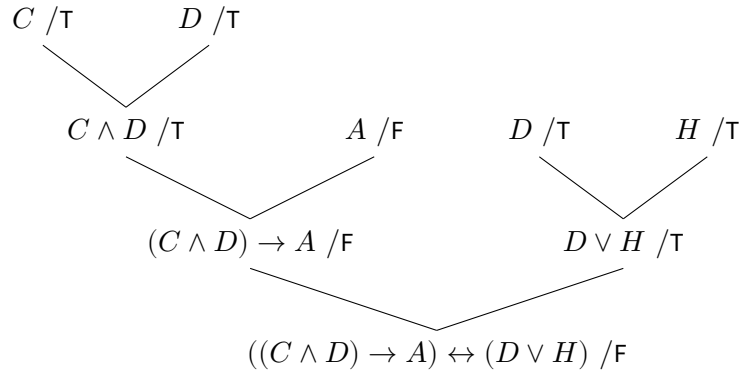
Proof: From the negation clause of the definition of truth and the fact that $\mathbf{m}(G) = \text{T}$, it follows that $\sim G$ is false on \mathbf{m} . From the conjunction clause of the definition of truth, $(A \vee (D \rightarrow B)) \wedge \sim G$ is true on \mathbf{m} iff each conjunct is. But $\sim G$ is false on \mathbf{m} , so $(A \vee (D \rightarrow B)) \wedge \sim G$ is too. ■

One way to make sure we compute the truth values of the subsentences in an appropriate order is by following the branches in the construction tree of the sentence. The idea is to start at the top of the construction tree (the truth values of which are given by the model) and work our way down the branches. We can even use the tree itself as an aid to computing truth values, writing the truth value of each subsentence next to it on the tree as we go. Let's illustrate with $(A \vee (D \rightarrow B)) \wedge \sim G$ on \mathbf{m} :



Example 2.25. Compute the truth value of the sentence $((C \wedge D) \rightarrow A) \leftrightarrow (D \vee H)$ on a model \mathbf{m} such that $\mathbf{m}(C) = \text{T}$, $\mathbf{m}(D) = \text{T}$, $\mathbf{m}(A) = \text{F}$, and $\mathbf{m}(H) = \text{T}$.

We again use a construction tree to illustrate how to work toward the answer.



2.2.2 Truth Functions and Truth Tables

A truth function is any function $f : \{\text{T}, \text{F}\} \times \dots \times \{\text{T}, \text{F}\} \Rightarrow \{\text{T}, \text{F}\}$, i.e., from sequences of truth values to truth values. The middle clauses of the definition of truth in a model—[2.21](#) (on page [23](#))—associate each logical connective of SL with a truth function. Often these truth functions are given by a truth table, though they can be written out explicitly in other ways. For example, the truth function associated with conditionals in the definition of truth can be represented as:

$$\begin{aligned}
 f(\text{T}, \text{T}) &= \text{T} \\
 f(\text{T}, \text{F}) &= \text{F} \\
 f(\text{F}, \text{T}) &= \text{T} \\
 f(\text{F}, \text{F}) &= \text{T}
 \end{aligned}$$

Or, for short:

$$9. \quad f(v_1, v_2) = \begin{cases} F & \text{if } v_1 = T \text{ and } v_2 = F \\ T & \text{otherwise} \end{cases}$$

The *truth table* for the conditional is:

$$10. \quad \begin{array}{c|cc} \rightarrow & T & F \\ \hline T & T & F \\ F & T & T \end{array}$$

or can be written alternatively as:

$$11. \quad \begin{array}{ccc} \phi & \psi & \phi \rightarrow \psi \\ \hline T & T & T \\ T & F & F \\ F & T & T \\ F & F & T \end{array}$$

The truth functions for \sim and \wedge are good translations of ‘not’ and ‘and’ in English, respectively. The symbol \vee is a good translation of the inclusive sense of ‘or’ in which the overall sentence is true iff at least one component is true. When we come to translations in chapter 5, we will discuss the exclusive sense of ‘or’ for which the overall sentence is true just in case exactly one component is true. Our truth function for \rightarrow is the best truth-functional translation of ‘if...then...’ in English, but its adequacy is a matter of controversy. The debate began more than 2,000 years ago, when the ancient philosophers Diodorus Cronus and Philo of Megara argued about whether conditionals could adequately be captured by truth-functional semantics. Perhaps the next 2,000 years will bring the controversy to a close.

2.2.3 Logical Truth: TFT, TFF, & TFC

A randomly chosen sentence will probably be true on some models and false on others. However, some special sentences are true on all models. One example of such is the sentence $A \vee \sim A$. Others are false on all models, e.g. $A \wedge \sim A$.

Definition 2.26. A sentence ϕ of SL is *truth functionally true* (TFT) iff it is true on all models for ϕ .

Definition 2.27. A sentence ϕ of SL is *truth functionally false* (TFF) iff it is false on all models for ϕ .

Definition 2.28. A sentence ϕ of SL is *truth functionally contingent* (TFC) iff it is true on one model for ϕ and false on another.

Example 2.29. Prove that $A \wedge \sim A$ is TFF.

Proof: Any model \mathbf{m} for $A \wedge \sim A$ has to assign either T or F to A . If it assigns T to A , then $\sim A$ will be false on \mathbf{m} . So $A \wedge \sim A$ is false on \mathbf{m} . But if \mathbf{m} assigns F to A , then

again $A \wedge \sim A$ is false on \mathbf{m} . Either way, the sentence is false on \mathbf{m} . This holds in all models \mathbf{m} , so $A \wedge \sim A$ is TFF. ■

Example 2.30. Similar reasoning shows that any sentence of the form $\phi \wedge \sim \phi$ is TFF. By definition 2.21 (on page 23), ϕ is either true or false on any model \mathbf{m} . If ϕ is false on \mathbf{m} , then $\phi \wedge \sim \phi$ is false on \mathbf{m} . If ϕ is true on \mathbf{m} , then $\sim \phi$ is false on \mathbf{m} and the conjunction $\phi \wedge \sim \phi$ is false on \mathbf{m} . Either way, $\phi \wedge \sim \phi$ is false on \mathbf{m} , and this holds for all models.

Example 2.31. Similar reasoning will show that any sentence of the form $\phi \vee \sim \phi$ is TFT. We leave it to the reader to adapt the argument.

Example 2.32. In examples 2.23 (on page 24) and 2.25 (on page 25) we saw that the sentence $(A \vee (D \rightarrow B)) \wedge \sim G$ is false in some models. To see that it's true in some models, consider a model that's just like the one used in example 2.23 (on page 24), but assigns F to G instead of T. In that model the sentence is true. Hence the sentence is TFC.

Example 2.33. The sentence $B \rightarrow (C \rightarrow B)$ is TFT. Any model \mathbf{m} will assign either T or F to B . If it assigns F to B , then $B \rightarrow (C \rightarrow B)$ is true on \mathbf{m} , because, according to the def. of truth for \rightarrow , a conditional is true if the LHS is false. If \mathbf{m} assigns T to B , then $C \rightarrow B$ is true on \mathbf{m} , again because of the def. of truth for \rightarrow . But if $C \rightarrow B$ is true on \mathbf{m} , it follows that $B \rightarrow (C \rightarrow B)$ is true on \mathbf{m} .

Example 2.34. The sentence $\sim C \wedge ((B \rightarrow C) \wedge B)$ is TFF. There are a few ways we can show this, but here we'll use a different approach than we used in the previous examples. Assume that the sentence *isn't* TFF. Then there is some model \mathbf{m} that makes it true. By def. of truth for \wedge , it follows that both conjuncts $\sim C$ and $(B \rightarrow C) \wedge B$ are true on this \mathbf{m} . If $\sim C$ is true on \mathbf{m} , then C is false on \mathbf{m} ; so \mathbf{m} assigns F to C . If $(B \rightarrow C) \wedge B$ is true on \mathbf{m} , then both conjuncts are true; so both $B \rightarrow C$ and B are true on \mathbf{m} . Thus C is true on \mathbf{m} too; so \mathbf{m} assigns T to C . But \mathbf{m} can't assign both F and T to C , so there can't be any model \mathbf{m} that makes $\sim C \wedge ((B \rightarrow C) \wedge B)$ true.

The method of demonstration used in example 2.34 is called *Indirect Proof* or *reductio ad absurdum* (or sometimes just '*reductio*' or 'RAA'). On this method, we assume the opposite of our desired conclusion and then work our way to a contradiction. Because we know that a contradiction can't be true, we then infer that the original assumption must be false. For many problems and theorems, RAA is the easiest method to use. It is called '*reductio ad absurdum*' because it 'reduces' the initial assumption to a contradiction, an absurdity.

Although the definitions of TFT, TFF, and TFC are specific to SL (after all, the models to which each definition refers are models of SL), we can use essentially the same definitions for *any* formal language for which we have some notion of a model. Broadly speaking, we can think of sentences which fit these definitions as

being (respectively) *logically true*, *logically false*, and *logically contingent*. Hereafter we'll sometimes use the more general term 'logical truth' instead of 'truth functional truth' when it's clear we're talking about SL. A sentence that's a logical truth is sometimes said to be *valid*, or said to be a *tautology*. We will avoid these terms in this context.

2.2.4 Procedures for Testing TFT, TFF, & TFC

In the above examples (2.29–2.34) we used the definitions of TFT, TFF, & TFC (definitions 2.26–2.28) and the definition for truth (definition 2.21 (on page 23)) to show whether a given SL sentence was TFT, TFF, or TFC. But there are methods or procedures for systematically testing whether a given SL sentence is TFT, TFF, or TFC.

Perhaps the most well known method is using truth tables.¹⁴ Later in this chapter we'll prove theorem 2.68 (on page 42), which says that when looking at a given SL sentence ϕ we only need to think about the *minimal* models that assign truth values to the sentence letters appearing in ϕ . One way to test whether ϕ is TFT, TFF, or TFC is to write down all the possible assignments of truth values to the sentence letters of ϕ , then for each compute the truth value of ϕ on that assignment. The number of possible assignments (minimal models) are finite. In fact if there are n sentence letters in ϕ there will be 2^n possible assignments. If ϕ turns out true in all these possible assignments, then ϕ is TFT. If it turns out false in all of them, then ϕ is TFF. And if it is true in some assignments and false in others, then ϕ is TFC.

We can diagrammatically represent this procedure by arranging all the possible assignments of truth values to sentences letters appearing in the given sentence ϕ and the truth value of ϕ on those assignments in rows. We will use $(A \vee (D \rightarrow B)) \wedge \sim G$ as an example. If ϕ has n sentence letters, there will be 2^n rows. Because this sentence has 4 sentence letters, its table will have 16 rows. We write the 4 sentence letters on a top row and then fill out the assignments by starting at the far right sentence letter (G), putting T and F alternating below it for the 16 rows. (See table 2.1.) Next, move to the second-to-the-right sentence letter (D) and write 2 T's and 2 F's alternating below it for 16 rows. Then move to the sentence letter to the left of the last one (B) and write 4 T's and 4 F's alternating below it until all the rows are filled. Finally, move to the last sentence letter (A) and write 8 T's and 8 F's alternative below it until all the rows are filled. This will fill out the 16 rows in such a way that each row gives a distinct possible assignment of truth values to sentence letters and no possible assignment is missed. In general the pattern is to start at the far right column alternating T and F, then move to the left doubling the number of T's and F's that appeared in the previous column.

With all the possible assignments to the sentence letters filled out, we write the sentence itself to the right of the sentence letters and under it put, in the respective

¹⁴On a historical note, Hodges (2001b: 5) says that Peirce (1902) was the first to use truth tables, but his student Ladd-Franklin (1883: 62) had something similar.

A	B	D	G	$((A \vee (D \rightarrow B)) \wedge \sim G)$
T	T	T	T	F
T	T	T	F	T
T	T	F	T	F
T	T	F	F	T
T	F	T	T	F
T	F	T	F	T
T	F	F	T	F
T	F	F	F	T
F	T	T	T	F
F	T	T	F	T
F	T	F	T	F
F	T	F	F	T
F	F	T	T	F
F	F	T	F	F
F	F	F	T	F
F	F	F	F	T

Table 2.1: Sample Truth Table

rows, its truth value on that assignment. The truth value of the sentence on the given assignment can be computed in any number of ways, e.g., using trees as was done in example 2.24 (on page 24). Once the truth value of the sentence has been computed for all the rows it's a trivial matter to read off the table whether the sentence is TFT, TFF, or TFC. If T is under the sentence in every row, then it's TFT. If F is in every row, then it's TFF. And if each T and F appear in at least one row, then it's TFC. By looking at table 2.1 we can see that $((A \vee (D \rightarrow B)) \wedge \sim G)$ is TFC.

The process of filling out a truth table is entirely 'mechanical.' That is, one can carry out the process by following purely formal rules. Once we have a truth table for some SL sentence, we can again use purely formal rules to determine whether it's TFT, TFF, or TFC. Therefore, no creativity is necessary to determine whether any given sentence is, e.g., TFT.

Example 2.35. We already saw in example 2.29 (on page 26) that $A \wedge \sim A$ is TFF. The following truth table confirms this.

A	$A \wedge \sim A$
T	F
F	F

Example 2.36. In example 2.33 (on page 27) we saw that $B \rightarrow (C \rightarrow B)$ is TFT. This is confirmed by the following truth table.

B	C	$B \rightarrow (C \rightarrow B)$
T	T	T
T	F	T
F	T	T
F	F	T

Example 2.37. We saw in example 2.34 (on page 27) that the sentence $\sim C \wedge ((B \rightarrow C) \wedge B)$ is TFF. The following truth table confirms this.

B	C	$\sim C \wedge ((B \rightarrow C) \wedge B)$
T	T	F
T	F	F
F	T	F
F	F	F

The various formal derivation systems used in logic provide another class of procedures for testing SL sentences for TFT and TFF (not all derivation systems provide a procedure for testing for TFC, but semantic tableaux do). In chapter 6 we develop one such derivation system, which is a specific variant of what's called a natural deduction system. Some derivation systems, in particular variants of semantic tableaux, provide more or less direct ways of testing for TFT (logical truths), TFF (logical falsehoods), and TFC (logical contingencies).¹⁵ Other derivation systems, such as variants of natural deduction systems, Hilbert-style axiomatic systems, and Gentzen-style sequent calculi, don't on their own provide direct means of testing for them. But with these specific algorithms or ways of using the derivation systems can be developed (as we do in chapter 7) that provide a testing procedure.

While truth tables are a convenient method for answering many questions about SL sentences, two warnings are in order. First, for complex sentences the size of the truth table can be very large. Remember, the number of rows needed for a truth table is 2^n where n is the number of different sentence letters. Second, while truth tables are useful in SL, there is nothing comparable for our later formal languages. The sooner you learn to analyze sentences directly, rather than relying on a truth table, the better. For example, consider the sentence $(A \rightarrow (B \wedge \sim(C \vee D))) \vee (E \rightarrow A)$. You *could* evaluate this sentence with a 32 line truth table. Or, you can instead note that if A is true in a model, then $(E \rightarrow A)$ is also true (def. of truth for \rightarrow), and so the whole sentence is true; and if A is false on the model, then $(A \rightarrow (B \wedge \sim(C \vee D)))$ is also true (def. of truth for \rightarrow), and the whole sentence is true. Every model must make A either true or false, so the whole sentence is true either way. Therefore it is TFT.

¹⁵A comprehensive introductory treatment of semantic tableaux (called truth trees by the author) is given in Nicholas J.J. Smith's (2012) textbook *Logic: The Laws of Truth*. Smith also gives a more detailed and thorough introduction to truth tables.

2.3 Entailment and other Relations

2.3.1 Entailment

We've discussed what it is for a sentence of SL to be true in a model and the classification of TFT, TFF, and TFC sentences. Now we turn to the notion of entailment. Entailment is a 2-place relation that holds between sets of sentences and individual sentences. The symbol ' \models ', called the double turnstile, is used to represent entailment. We write that Δ entails θ as follows: ' $\Delta \models \theta$ '. (The ' \models ' is *not* a symbol of SL. Like the Greek letters it is a symbol of MathEnglish.)

Definition 2.38. If Δ is a set of SL sentences and θ is an SL sentence, then the following are equivalent ways to define when Δ entails θ :

- (1) $\Delta \models \theta$ iff every model for Δ and θ that makes all sentences in Δ true also makes θ true.
- (2) $\Delta \models \theta$ iff every model for Δ and θ either makes at least one sentence in Δ false or makes θ true.

Δ can be the empty set or an infinite set. If Δ is the empty set then we simply write ' $\models \theta$ '. The following are a consequence of the definition of entailment:

- (1) $\phi_1, \phi_2, \phi_3, \dots, \phi_n \models \theta$ iff every model for $\phi_1, \phi_2, \phi_3, \dots, \phi_n$, and θ that makes all of $\phi_1, \phi_2, \phi_3, \dots, \phi_n$ true also makes θ true.
- (2) $\phi_1, \phi_2, \phi_3, \dots, \phi_n \models \theta$ iff every model for $\phi_1, \phi_2, \phi_3, \dots, \phi_n$, and θ either makes at least one of $\phi_1, \phi_2, \phi_3, \dots, \phi_n$ false or makes θ true.

And for an even more narrow consequence:

- (1) A sentence ϕ entails another sentence θ iff every model for ϕ and θ that makes ϕ true also makes θ true.
- (2) A sentence ϕ entails another sentence θ iff there is no model for ϕ and θ that makes ϕ true and θ false.

Above we mentioned that $\Delta \models \theta$ may hold even in cases where Δ is the empty set: $\models \theta$. If $\models \theta$, then every model for θ must make a sentence to the left of the turnstile false, or make θ true. But there are no sentences to the left of the turnstile for a model to make false. So, every model for θ must make θ true. Therefore:

Theorem 2.39. *For all SL sentences θ , $\models \theta$ iff θ is TFT.*

Before moving on to examples, it's worth noting that the definition of entailment given here only makes sense for sentences of SL. It can't be applied directly to meaningful English sentences. But as we discuss in section 5.1 (on page 107), through translations

between SL and English we can begin to talk about entailment between English sentences. And this will be at least a start to elucidating the notion of logical consequence in English.

Example 2.40. $(A \wedge B) \models B$, that is, any model that makes $(A \wedge B)$ true makes B true as well.

Proof: We must show that for all models \mathbf{m} , if \mathbf{m} makes $(A \wedge B)$ true, then it also makes B true. Assume a model \mathbf{m} makes $(A \wedge B)$ true. By the definition of truth for \wedge —clause 3, 2.21 (on page 23)—both A and B are true in \mathbf{m} as well. ■

Example 2.41. $B \models (A \vee B)$. We leave the proof to the reader.

Example 2.42. For all ϕ , for some θ , $\phi \models \theta$. That is, every SL sentence entails at least one SL sentence.

Proof: We need to show that no matter what SL sentence ϕ you pick, there's always some SL sentence θ which is entailed by it, i.e. always some θ such that $\phi \models \theta$. But this is easy: for every SL sentence ϕ you pick, just let $\theta = \phi$. Every SL sentence entails itself. Thus, no matter what SL sentence you pick, there's always some sentence entailed by it. ■

Example 2.43. For some θ , for all ϕ , $\phi \models \theta$. That is, there's some SL sentence that's entailed by all SL sentences.

Proof: We need to find some θ such that all SL sentences entail it. Let θ be $(A \vee \sim A)$. This sentence is TFT, so no model makes it false. No matter what sentence ϕ we pick, there is no model that makes ϕ true and θ false. So, by the definition of \models (2.38), ϕ entails θ . ■

Example 2.44. For some ϕ , for all θ , $\phi \models \theta$. We leave the proof to the reader.

2.3.2 Procedures for Testing Entailment

In the previous examples (2.40–2.44) we used the definition of entailment (definition 2.38) to reason about whether a given entailment holds. But there are various procedures for mechanically checking whether a given entailment holds. Truth tables provide the simplest way.

To test whether a set of sentences Δ entails a sentence θ , we first write down all the sentence letters that appear in θ and that appear in the sentences in Δ . Again we put these on the left side of the truth table and under them (following the procedure outlined in section 2.2.4, on page 28) we write all the possible assignments of truth values. Then to the right of the sentence letters we write each of the sentences from Δ (in separate columns) and to the right of those we write θ . Under θ and all the sentences from Δ we write the truth value of that sentence based on the assignment

of truth values listed in that row. Then Δ entails θ iff there is no row in the truth table in which all the sentences in Δ are true and θ is false. Any rows of the truth table that do not make all of the LHS sentences true are irrelevant. After marking one of the LHS sentences as false we can omit the rest of the row.

Example 2.45. In example 2.40 we said that $(A \wedge B) \models B$. We can show this with a truth table.

A	B	$(A \wedge B)$	B
T	T	T	T
T	F	F	F
F	T	F	T
F	F	F	F

Example 2.46. In example 2.41 we said that $B \models (A \vee B)$. The following truth table shows this.

A	B	B	$(A \vee B)$
T	T	T	T
T	F	F	T
F	T	T	T
F	F	F	F

Example 2.47. We conclude with a more complicated example. Here we want to show that $A \vee B, \sim C \rightarrow \sim A, B \rightarrow C \models C$.

A	B	C	$A \vee B$	$\sim C \rightarrow \sim A$	$B \rightarrow C$	C
T	T	T	T	T	T	T
T	T	F	T	F	F	F
T	F	T	T	T	T	T
T	F	F	T	F	T	F
F	T	T	T	T	T	T
F	T	F	T	T	F	F
F	F	T	F	T	T	T
F	F	F	F	T	T	F

2.3.3 Basic Results on Entailment

A simple, but important theorem involves moving sentences from one side of the turnstile to the other.

Theorem 2.48. SL Exportation Theorem: *For all SL sentences ϕ and θ , $\phi \models \theta$ iff $\models \phi \rightarrow \theta$.*

In order to prove an iff-statement like this one (i.e., a biconditional), we need to prove two things. First, we have to prove that if the left-hand side—in this example, $\phi \models \theta$ —is true, then the right-hand side—in this example, $\models \phi \rightarrow \theta$ —must also be true. Second, we have to prove that if the right-hand side is true, then the left-hand side must also be true. Proofs of a biconditional generally involve these two parts.¹⁶ The convention is that the first part of the proof (the one that shows that if the left-hand side is true, then the right-hand side is true) will be marked with a left-to-right arrow ‘ \Rightarrow ’, and the other part of the proof will be marked with a right-to-left arrow ‘ \Leftarrow ’.

Proof: (\Rightarrow) To begin, we assume that the left-hand side, $\phi \models \theta$, is true. We want to show, without using any additional information, that the right-hand side, $\models \phi \rightarrow \theta$, is also true. Given the definition of \models and the truth of $\phi \models \theta$, it follows that on every model on which ϕ is true, θ is also true.

There are two possibilities for the truth value of ϕ : either (a) ϕ is true or (b) ϕ is false. (a) Assume a model \mathbf{m} such that ϕ is true. Because $\phi \models \theta$, that means that \mathbf{m} makes θ true. By the definition of truth for \rightarrow , \mathbf{m} makes $(\phi \rightarrow \theta)$ true. (If a model makes the RHS of a conditional true, it makes the whole conditional true.) Remember this result! This is close to our goal.

Now for the second possibility. (b) Assume a model \mathbf{m} that makes ϕ false. According to the definition of truth for \rightarrow , \mathbf{m} makes $(\phi \rightarrow \theta)$ true. (If a model makes the LHS of a conditional false, it makes the whole conditional true.)

No matter what model we pick, ϕ must be true or false. Either way, we see—in (a) and (b)—that the model must make $(\phi \rightarrow \theta)$ true. In other words, all models make $(\phi \rightarrow \theta)$ true. Which in turn means that $(\phi \rightarrow \theta)$ is TFT (def. of TFT, 2.26, on page 26). And according to theorem 2.39 (on page 31), it follows that the entailment $\models (\phi \rightarrow \theta)$ holds.

(\Leftarrow) Now assume that $\models (\phi \rightarrow \theta)$ is true, with the goal of showing that $\phi \models \theta$ is also true. By the definition of \models , the truth of $\models (\phi \rightarrow \theta)$ implies that $(\phi \rightarrow \theta)$ is TFT. By the definition of TFT, this means that $(\phi \rightarrow \theta)$ is true on every model \mathbf{m} .

By the definition of truth of \rightarrow , this means that for every model \mathbf{m} , either the LHS, ϕ , is false or the RHS, θ , is true. Hence there is no model that makes ϕ true and θ false. So, by the definition of \models , that means that $\phi \models \theta$. ■

There are a number of other theorems that are similar to, and expand upon, theorem 2.48. Here we state them and leave the proofs to the reader.

Theorem 2.49.

(1) If $\phi_1, \phi_2, \dots, \phi_n$ and ψ are SL sentences, then

$$\begin{aligned} \phi_1, \phi_2, \dots, \phi_n \models \psi &\text{ iff } \phi_2, \dots, \phi_n \models (\phi_1 \rightarrow \psi) \\ \phi_1, \phi_2, \dots, \phi_n \models \psi &\text{ iff } \phi_1, \phi_3, \dots, \phi_n \models (\phi_2 \rightarrow \psi) \end{aligned}$$

¹⁶There are other ways of proving a biconditional that don’t involve explicitly carrying out these steps, but the method we use here is often the most straightforward.

- $$\vdots$$
- $$\phi_1, \phi_2, \dots, \phi_n \models \psi \text{ iff } \phi_1, \dots, \phi_{n-1} \models (\phi_n \rightarrow \psi)$$
- (2) If $\phi_1, \phi_2, \dots, \phi_n$ and ψ are SL sentences, then
- $$\phi_1, \phi_2, \dots, \phi_n \models \psi \text{ iff } \phi_1, \dots, \phi_{n-1} \models (\phi_n \rightarrow \psi)$$
- $$\phi_1, \phi_2, \dots, \phi_n \models \psi \text{ iff } \phi_1, \dots, \phi_{n-2} \models (\phi_{n-1} \rightarrow (\phi_n \rightarrow \psi))$$
- $$\vdots$$
- $$\phi_1, \phi_2, \dots, \phi_n \models \psi \text{ iff } \models (\phi_1 \rightarrow (\dots \rightarrow (\phi_{n-1} \rightarrow (\phi_n \rightarrow \psi))))$$
- (3) If ϕ and ψ are SL sentences and Δ is a set of SL sentences containing ϕ (i.e., $\phi \in \Delta$) and Δ^* is Δ with ϕ removed, then $\Delta \models \psi$ iff $\Delta^* \models (\phi \rightarrow \psi)$.
- (4) If ϕ and ψ are SL sentences, then $\phi \models \psi$ iff $\phi, \sim\psi \models (A \wedge \sim A)$.
- (5) If ϕ_1, \dots, ϕ_n and ψ_1, \dots, ψ_n are all SL sentences, then
- $$\phi_1, \dots, \phi_n \models (\psi_1 \vee \dots \vee \psi_n) \text{ iff } \phi_1, \dots, \phi_n, \sim\psi_1 \models (\psi_2 \vee \dots \vee \psi_n)$$
- $$\phi_1, \dots, \phi_n \models (\psi_1 \vee \dots \vee \psi_n) \text{ iff } \phi_1, \dots, \phi_n, \sim\psi_2 \models (\psi_1 \vee \psi_3 \vee \dots \vee \psi_n)$$
- $$\vdots$$
- $$\phi_1, \dots, \phi_n \models (\psi_1 \vee \dots \vee \psi_n) \text{ iff } \phi_1, \dots, \phi_n, \sim\psi_n \models (\psi_1 \vee \dots \vee \psi_{n-1})$$

2.3.4 Other Relations

We can define a number of other important relations between SL sentences in terms of entailment:

Definition 2.50. Two sentences θ and ϕ are *truth functionally equivalent* (TFE) iff all models for θ and ϕ assign them the same truth value, which is the same as saying they entail each other: both $\theta \models \phi$ and $\phi \models \theta$.

Definition 2.51. Two sentences θ and ϕ are *truth functionally contradictory* iff all models for θ and ϕ assign them opposite truth values, which is the same as saying that each sentence is TFE to the negation of the other.

Definition 2.52. Two sentences θ and ϕ are *truth functionally contrary* iff they cannot both be true in the same model \mathbf{m} . (This is the same as saying that each entails the negation of the other, or $\theta \wedge \phi \models A \wedge \sim A$.)

Definition 2.53. Two sentences θ and ϕ are *truth functionally subcontrary* iff they cannot both be false in the same model \mathbf{m} . (This is the same as saying that the negation of each entails the other, or $\models \theta \vee \phi$.)

Definition 2.54. Two sentences θ and ϕ are *truth functionally independent* iff none of the above hold (including entailments), i.e. iff there are four models:

- (1) A model in which both θ and ϕ are true;
- (2) A model in which both θ and ϕ are false;
- (3) A model in which θ is true and ϕ is false; and
- (4) A model in which θ is false and ϕ is true.

Example 2.55. Each pair of contradictory sentences is also contrary (this follows trivially from the definitions). But sentences can be contrary without being contradictory.

Proof: $C \wedge D$ and $C \wedge \sim D$ are contrary but not contradictory.

(Contrary:) If $C \wedge D$ is true in a model \mathbf{m} , then C and D are each true on \mathbf{m} . So $\sim D$ is false in \mathbf{m} , and $C \wedge \sim D$ is false in \mathbf{m} . If $C \wedge \sim D$ is true in \mathbf{m} , then both C and $\sim D$ are true on \mathbf{m} . D is false in \mathbf{m} , and hence $C \wedge D$ is false in \mathbf{m} . Thus, by def. 2.52, the pair is contrary.

(Not Contradictory:) Any model \mathbf{m} that assigns F to C makes both $C \wedge D$ and $C \wedge \sim D$ false. By def. 2.51, the pair is not contradictory. ■

Example 2.56. Contradictory sentences are also subcontrary, but sentences can be subcontrary without being contradictory.

Proof: D and $C \vee \sim D$ are subcontrary but not contradictory.

(Subcontrary:) Assume that D is false on a model \mathbf{m} . It follows that $\sim D$ is true on \mathbf{m} and so is $C \vee \sim D$. Alternatively, assume that $C \vee \sim D$ is false on \mathbf{m} . Then both C and $\sim D$ are false on \mathbf{m} . So D is true on \mathbf{m} . By def. 2.53, the pair is subcontrary.

(Not Contradictory:) Any model that assigns T to both D and C makes both D and $C \vee \sim D$ true. So by def. 2.51, the pair isn't contradictory. ■

Example 2.57. If two sentences are both contrary and subcontrary, they are contradictory.

Proof: If two sentences are contrary, then by definition 2.52 any model \mathbf{m} that makes one true makes the other false. If two sentences are subcontrary, then by definition 2.53 any model \mathbf{m} that makes one false makes the other true. Because every model either makes a sentence true or makes it false, no model assigns two sentences that are contrary and subcontrary the same truth value. So by definition 2.51, two sentences that are contrary and subcontrary are also contradictory. ■

Example 2.58. $A \vee (B \wedge D)$ and $(A \vee B) \wedge (A \vee D)$ are TFE.

Proof: Assume that $A \vee (B \wedge D)$ is true on some \mathbf{m} . By the def. of truth of \vee , either A is true on \mathbf{m} or $B \wedge D$ is true on \mathbf{m} . If A is true on \mathbf{m} , then both $A \vee B$ and $A \vee D$ are true on \mathbf{m} . So, $(A \vee B) \wedge (A \vee D)$ is true on \mathbf{m} . If, alternatively, $B \wedge D$ is true on \mathbf{m} , then then both B and D are true on \mathbf{m} , and so both $A \vee B$ and $A \vee D$ are true on

m. Hence, $(A \vee B) \wedge (A \vee D)$ is true on **m**. In either case, $(A \vee B) \wedge (A \vee D)$ is true on **m**. Thus, $A \vee (B \wedge D) \models (A \vee B) \wedge (A \vee D)$.

We leave it to the reader to show that $(A \vee B) \wedge (A \vee D) \models A \vee (B \wedge D)$. It then follows by definition 2.50 that the two sentences are TFE. ■

Example 2.59. For any SL sentences ϕ and θ , $\phi \rightarrow \theta$ and $\sim\phi \vee \theta$ are TFE.

Proof: Assume that $\phi \rightarrow \theta$ is true on some model **m**. Then on **m** either ϕ is false or θ is true. Hence either $\sim\phi$ or θ is true on **m**. It follows that $\sim\phi \vee \theta$ is true on **m**. Hence by the definition of \models , $\phi \rightarrow \theta \models \sim\phi \vee \theta$.

We leave it to the reader to show that $\sim\phi \vee \theta \models \phi \rightarrow \theta$. It then follows by definition 2.50 that the two sentences are TFE. ■

Example 2.60. For any SL sentence ϕ , $\sim\sim\phi$ and ϕ are TFE.

Proof: The Proof is left to the reader. ■

2.3.5 Procedures for Testing Other Relations

As before we can use truth tables to test for the following relationships: truth-functional equivalence, truth-functional contradictory, truth-functional contrary, truth-functional subcontrary, and truth-functional independence. By definition, two sentences ϕ and θ are truth functionally equivalent iff they have the same truth value on every assignment. So we can test for truth-functional equivalence by putting both ϕ and θ in a truth table and seeing if they have the same truth value in every row.

Example 2.61. In example 2.58 (on the previous page) we said that $A \vee (B \wedge D)$ and $(A \vee B) \wedge (A \vee D)$ are TFE. We can show this using truth tables.

A	B	D	$A \vee (B \wedge D)$	$(A \vee B) \wedge (A \vee D)$
T	T	T	T	T
T	T	F	T	T
T	F	T	T	T
T	F	F	T	T
F	T	T	T	T
F	T	F	F	F
F	F	T	F	F
F	F	F	F	F

Example 2.62. In example 2.59 (on the current page) we said that for any SL sentences ϕ and θ , $\phi \rightarrow \theta$ and $\sim\phi \vee \theta$ are TFE. This can be shown using truth tables.

ϕ	θ	$\phi \rightarrow \theta$	$\sim\phi \vee \theta$
T	T	T	T
T	F	F	F
F	T	T	T
F	F	T	T

In this example we're not looking at actual SL sentences; instead we have sentence schemas. But for reasons to be discussed below this procedure still works: this truth table shows that for any two sentences ϕ and θ , $\phi \rightarrow \theta$ and $\sim\phi \vee \theta$ are TFE.

While truth tables provide guaranteed answers to these questions, students should not become reliant on this method. For one thing, this method doesn't work for more complex languages, and it is important to practice with the simpler SL structures the reasoning skills and methods needed in later chapters. Secondly, truth tables can quickly become very large and tedious, and often simple reasoning suffices. More generally, truth tables give an answer but often don't give insight.

For example, the truth table for the sentence $((A \rightarrow B) \wedge (B \rightarrow C)) \rightarrow (A \rightarrow C)$ requires 8 lines. But we can reason that the only way for this sentence to be false is if $((A \rightarrow B) \wedge (B \rightarrow C))$ is true in \mathbf{m} and $(A \rightarrow C)$ is false. $(A \rightarrow C)$ can only be false in \mathbf{m} if A is true and C is F in \mathbf{m} . Now we look at B and ask what \mathbf{m} should assign it to make the sentence false; we see that if B is T, $(B \rightarrow C)$ is false and the LHS is false so the whole sentence is true; if B is F in \mathbf{m} , then $(A \rightarrow B)$ is false, the LHS is false and the whole sentence is true. Therefore *there is no model that makes the sentence F* and it is TFT. We have discovered a pattern. And if we consider a similar but more complex sentence in which A , B and C are systematically replaced by complex sentences, the size of the truth-table blows up, but the reasoning above remains simple.

2.4 Recursive Proofs

2.4.1 The Method of Recursive Proof

Many logic concepts are characterized by recursive definitions. To prove a theorem about recursively defined concept, we usually want to employ a method called *recursive proof*. The structure of a recursive proof mirrors the structure of a recursive definition.

Definition 2.63. Let Δ be some set whose members are defined recursively. To prove that all members of Δ have some property ϕ we use a *recursive proof*, which works as follows:

Base Step: Show that everything identified by the base clause of the recursive definition has ϕ .

Inheritance Step: Show that ϕ is inherited; i.e., show that if the previous objects from which new objects are generated (or found) by the generating clause have ϕ then the new ones have ϕ too.

Closure Step: Finally, show that completing the base and inheritance steps is sufficient to show that all members of Δ have ϕ .

The inheritance step usually has two parts. The first part is a *recursive assumption*. We will *assume* that some previous objects—the objects from which new ones are generated by the generating clause—already have the property in question. Generally we make this assumption by selecting metavariables to represent the previous objects. We are entitled to this assumption because we know with certainty that there are some such previous objects. At the very least, the objects previously identified in the base clause have the property. Second, we prove that the new, generated objects must also have the property in question.

How much we actually write down in the closure step will vary from proof to proof. Sometimes we will simply note that the closure condition (“that’s all”) ensures nothing has been left out by our proof. Other times, if the proof is more complicated, we might also reiterate what we have just shown.

Throughout the book we signify the end of a completed proof with a black box: ■.

2.4.2 Recursive Proof and Mathematical Induction

Many important mathematical concepts are defined recursively. We already saw the recursive definition of natural numbers. If we want to *prove* something about all natural numbers, we can use mathematical induction. *Mathematical induction* is a method of proof in which one shows (i) that some property holds of the first natural number, and (ii) that for any natural number n having that property, the successor of n also has that property. Mathematical induction is one sort of recursive proof:

Base Step: 0 has property ϕ .

Inheritance Step: Whenever n has property ϕ , its successor, $n + 1$, also has ϕ .

Therefore, we can conclude that

Closure Step: all natural numbers have property ϕ .

Let’s go through an example of mathematical induction. This first proof isn’t very exciting but it illustrates how recursive proofs work.

Example 2.64. Let’s prove that there is no largest natural number n , i.e., that there are an infinity of natural numbers.

Proof:

- (1) Base Step: The number 0 isn’t the largest; 1 is larger.
- (2) Inheritance Step: First we’ll make our recursive assumption. Assume that n isn’t the largest natural number.

We want to show that the *successor* of n —i.e., $n + 1$ —isn't the largest natural number either. This is easy. $n + 1$ can't be the largest because $n + 1 < n + 2$.

So, we've shown that the natural number $n + 1$ can't be the largest.

- (3) Closure Step: So, no natural number n is the largest.

■

Let's look at another, slightly more difficult recursive proof.

Example 2.65. For every natural number n , there is a set that has exactly n elements.

Proof:

- (1) Base Step: The empty set, \emptyset , has 0 elements.
- (2) Inheritance Step: Recursive Assumption: Assume that for natural number n there is a corresponding set Δ containing only the natural numbers i such that $0 < i \leq n$. This means that Δ has exactly n elements. (To illustrate, if $n = 1$, then $\Delta = \{1\}$; if $n = 2$, then $\Delta = \{1, 2\}$; etc.)

We want to show that for the successor of n , $n + 1$, there is another set Δ' with $n + 1$ elements. We can construct such a set by defining Δ' as $\Delta \cup \{n + 1\}$. The set $\{n + 1\}$ has exactly one element, and this element isn't in Δ . Because Δ has n elements and $\Delta' = \Delta \cup \{n + 1\}$, it follows that Δ' has one more element than Δ , i.e., that it has $n + 1$ elements.

We've shown that for the natural number $n + 1$ there is a corresponding set Δ' with $n + 1$ elements.

- (3) Closure Step: So, for every natural number n there is a corresponding set with n elements.

■

Two things can give people trouble with the structure of recursive proofs (aside from intrinsically hard problems). One is that the base case is often easy or trivial. Make sure you have done it correctly but don't worry if it seems too easy.

Second, some students initially think that the recursive assumption in the inheritance step assumes what we are trying to prove, but that isn't so. Think of the recursive assumption this way: *if* some group of objects has property X, then from that we can prove that another group of objects also has X. We already know that some objects have the property in question: the object(s) identified in the base step. The recursive assumption lets us give a label to these previously identified objects bearing the property in question. In the last example, we used the variable n to stand for numbers that we already know have a corresponding set. That allowed us to show that other objects—in this example, designated by $n + 1$ —also have the same property.

2.5 Recursive Proofs in SL

2.5.1 Recursive Proof Examples in SL

Let's illustrate recursive proofs further by establishing two simple results about SL sentences.

We'll use the method to prove more surprising and substantial results throughout the rest of this book.

Example 2.66. In this example we prove that every (official) sentence has exactly as many left parentheses as right.

Proof:

Base Step: All atomic sentences (i.e., sentence letters) have zero left parentheses and zero right parentheses. And, of course, $0 = 0$.

Inheritance Step: Suppose ϕ and $\theta, \theta_1, \theta_2, \dots, \theta_n$ each have exactly as many left parentheses as right, and are of order k or less. (This is our recursive assumption.) We want to show that sentences of order $k + 1$ have the same property.

New sentences can be generated as $(\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n)$, $(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n)$, $(\phi \rightarrow \theta)$, and $(\phi \leftrightarrow \theta)$. In all these cases the resulting parentheses are all those of ϕ and θ plus the new matching outer ones; so, the resulting sentence still has exactly as many left parentheses as right. The only other way new sentences can be generated is as $\sim\phi$, in which case the resulting parentheses are just those of ϕ and that, by the recursive assumption, has exactly as many left parentheses as right.

Closure Step: Because the inheritance step covers all the ways of generating an SL sentence, for all SL sentences the number of left parentheses is the same as the number of right parentheses. ■

Example 2.67. In this example we prove that in every official SL sentence, the number of left parentheses is greater than or equal to the number of arrows.

Proof: Let $LP\phi$ be the number of left parentheses in ϕ and $AR\phi$ be the number of arrows in ϕ .

Base Step: In the base case ϕ is atomic, so $LP\phi = 0$ and $AR\phi = 0$. Thus, $LP\phi = AR\phi$ and so $LP\phi \geq AR\phi$. This holds for *all* atomic ϕ .

Inheritance Step:

Recursive Assumption: Assume the above property holds for some $\theta, \theta_1, \theta_2, \dots, \theta_n$, which are all of order k or less. That is, assume that $LP\theta \geq AR\theta$, $LP\theta_1 \geq AR\theta_1$, $LP\theta_2 \geq AR\theta_2$, ... $LP\theta_n \geq AR\theta_n$. Let's show that the sentences of order $k + 1$ have the same property.

Negation: $LP\sim\theta = LP\theta$ and $AR\sim\theta = AR\theta$. By assumption $LP\theta \geq AR\theta$, so $LP\sim\theta \geq AR\sim\theta$ too.

Conditional: $LP(\theta_1 \rightarrow \theta_2) = LP\theta_1 + LP\theta_2 + 1$ and $AR(\theta_1 \rightarrow \theta_2) = AR\theta_1 + AR\theta_2 + 1$. Because $LP\theta_1 \geq AR\theta_1$ and $LP\theta_2 \geq AR\theta_2$, clearly $LP\theta_1 + LP\theta_2 + 1 \geq AR\theta_1 + AR\theta_2 + 1$ too. So $LP(\theta_1 \rightarrow \theta_2) \geq AR(\theta_1 \rightarrow \theta_2)$.

Biconditional: $LP(\theta_1 \leftrightarrow \theta_2) = LP\theta_1 + LP\theta_2 + 1$ and $AR(\theta_1 \leftrightarrow \theta_2) = AR\theta_1 + AR\theta_2$. Because $LP\theta_1 \geq AR\theta_1$ and $LP\theta_2 \geq AR\theta_2$, clearly $LP\theta_1 + LP\theta_2 + 1 \geq AR\theta_1 + AR\theta_2$ too. So $LP(\theta_1 \leftrightarrow \theta_2) \geq AR(\theta_1 \leftrightarrow \theta_2)$.

Disjunction: We have that $LP(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n) = LP\theta_1 + LP\theta_2 + \dots + LP\theta_n + 1$ and $AR(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n) = AR\theta_1 + AR\theta_2 + \dots + AR\theta_n$. Because $LP\theta_1 \geq AR\theta_1$, $LP\theta_2 \geq AR\theta_2$, ... $LP\theta_n \geq AR\theta_n$, we have that $LP\theta_1 + LP\theta_2 + \dots + LP\theta_n + 1 \geq AR\theta_1 + AR\theta_2 + \dots + AR\theta_n$. So, $LP(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n) \geq AR(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n)$ too.

Conjunction: The argument is literally the same as that for disjunction, just erase every token of ‘ \vee ’ and replace it with a token of ‘ \wedge ’.

Closure Step: These are all the ways of constructing SL sentences, and in every case, if the number of left parentheses is greater than or equal to the number of arrows in the component sentences, then that still holds in the new sentences generated from those components.

■

In both examples we are trying to prove something about all SL sentences. In both cases our recursive assumption is something like: assume that the property in question already holds, in particular, for some SL sentences ϕ_1, \dots, ϕ_m of order k or less. We do this because in the inheritance step we’re trying to show that if you build new SL sentences (using the logical connectives) out of old ones for which the property holds, then the property also holds for the new ones. One way to do this—the one just used here—is to assume that the property holds for some particular sentences ϕ_1, \dots, ϕ_m and then show that it holds for those built from those sentences using the logical connectives.

2.5.2 Minimal Model Theorem

Earlier we claimed that the only sentence letter assignments that matter for the truth value of some SL sentence ϕ are those actually in ϕ . We’re now ready to prove it.

Theorem 2.68. *If two models for ϕ , \mathbf{m}_1 and \mathbf{m}_2 , make the same assignments to all the sentence letters contained in ϕ , then ϕ is true on \mathbf{m}_1 iff ϕ is true on \mathbf{m}_2 .*

Proof:

Base Step: Let θ be a SL sentence of order 1. It follows that θ must be a lone sentence letter. If \mathbf{m}_1 and \mathbf{m}_2 make the same assignments for all the sentence letters, then θ , which is just one sentence letter, is true on \mathbf{m}_1 iff θ is true on \mathbf{m}_2 .

Inheritance Step: Recursive Assumption: Assume that for each sentence θ of order n or less, θ is true on \mathbf{m}_1 iff θ is true on \mathbf{m}_2 .

Negation: Say that ψ is of the form $\sim\theta$. $\sim\theta$ is true on \mathbf{m}_1 iff θ is false on \mathbf{m}_1 (definition of truth, \sim). And $\sim\theta$ is true on \mathbf{m}_2 iff θ is false on \mathbf{m}_2 . By our recursive assumption (RA), θ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 . It follows that $\sim\theta$ is true on \mathbf{m}_1 iff $\sim\theta$ is true on \mathbf{m}_2 . I.e., ψ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Conditional: Say that ψ is of the form $\theta_1 \rightarrow \theta_2$. $\theta_1 \rightarrow \theta_2$ is true on \mathbf{m}_1 iff either θ_1 is false or θ_2 is true on \mathbf{m}_1 (definition of truth, \rightarrow). The same holds on model \mathbf{m}_2 . By RA, θ_1 is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 ; and θ_2 is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 . Therefore, there are four possibilities: (i) θ_1 is true on both \mathbf{m}_1 and \mathbf{m}_2 , and θ_2 is true on both too; (ii) θ_1 is false on both \mathbf{m}_1 and \mathbf{m}_2 , and θ_2 is false on both too; (iii) θ_1 is false on both \mathbf{m}_1 and \mathbf{m}_2 , and θ_2 is true on both models; and (iv) θ_1 is true on both \mathbf{m}_1 and \mathbf{m}_2 , and θ_2 is false on both models. In cases (i) through (iii), $\theta_1 \rightarrow \theta_2$ is true on both \mathbf{m}_1 and \mathbf{m}_2 . In case (iv), $\theta_1 \rightarrow \theta_2$ is false on both \mathbf{m}_1 and \mathbf{m}_2 . Thus, ψ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Biconditional: Say that ψ is of the form $\theta_1 \leftrightarrow \theta_2$. $\theta_1 \leftrightarrow \theta_2$ is true on \mathbf{m}_1 iff θ_1 and θ_2 have the same truth value on \mathbf{m}_1 (definition of truth, \leftrightarrow). The same holds on model \mathbf{m}_2 . By RA, θ_1 is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 ; and θ_2 is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 . There are two possibilities: (i) θ_1 and θ_2 share the same truth value on both \mathbf{m}_1 and \mathbf{m}_2 ; and (ii) θ_1 and θ_2 have differing truth values on both \mathbf{m}_1 and \mathbf{m}_2 . In case (i), $\theta_1 \leftrightarrow \theta_2$ is true on both \mathbf{m}_1 and \mathbf{m}_2 . In case (ii), $\theta_1 \leftrightarrow \theta_2$ is false on both \mathbf{m}_1 and \mathbf{m}_2 . Thus, ψ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Disjunction: Say that ψ is of the form $(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n)$. $(\theta_1 \vee \theta_2 \vee \dots \vee \theta_n)$ is true on \mathbf{m}_1 iff at least one disjunct is true on \mathbf{m}_1 (definition of truth, \vee). As before, the same holds on \mathbf{m}_2 . By RA, for every disjunct θ_k , θ_k is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 . So, either one disjunct (at least) is true on both \mathbf{m}_1 and \mathbf{m}_2 , or none of the disjuncts is true on either of \mathbf{m}_1 and \mathbf{m}_2 . Either way, ψ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Conjunction: Say that ψ is of the form $(\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n)$. $(\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n)$ is true on \mathbf{m}_1 iff every conjunct is true on \mathbf{m}_1 (definition of truth, \wedge). Again, the same holds on \mathbf{m}_2 . By RA, for every conjunct θ_k , θ_k is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 . Thus, either every conjunct is true on both \mathbf{m}_1 and \mathbf{m}_2 , or at least one conjunct is false on both \mathbf{m}_1 and \mathbf{m}_2 . Either way, ψ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Closure Step: There is no other way to form a sentence of SL ϕ , so the above clauses are sufficient to prove that if \mathbf{m}_1 and \mathbf{m}_2 make the same assignments for all the sentence letters, then ϕ is true on \mathbf{m}_1 iff ϕ is true on \mathbf{m}_2 . ■

2.5.3 Main Connective Theorem

Earlier (2.11) we defined the main connective of a sentence ϕ as the connective that isn't in any proper subsentence of ϕ . Atomic sentences don't have any connectives, and so don't have main connectives. Most non-atomic sentences have just one main connective, but there is an exception: conjunctions (or disjunctions) with n tokens of ' \wedge ' (or ' \vee ') that are all the main connectives, where $n > 1$. In such sentences, $n + 1$ is the number of conjuncts (disjuncts). For example, all three of the ' \wedge ' tokens in the following sentence are the main connectives: $(B \wedge A \wedge H \wedge G)$. Let's call sentences that have multiple tokens of ' \wedge ' as main connectives 'extended conjunctions', and those that have multiple tokens of ' \vee ' as main connectives 'extended disjunctions'. With this new terminology, let's turn to another recursive proof.

Theorem 2.69. Main Connective Theorem: *For every SL sentence ϕ , one of the following holds: (i) ϕ has no main connective; (ii) ϕ has exactly one main connective token; or (iii) ϕ is an extended conjunction (or disjunction) with $n - 1$ main connective tokens, where n is the number of conjuncts (disjuncts).*

Proof of Thm. 2.69, Main Connective Theorem:

Base Step: Every SL sentence of order 1 is atomic (i.e., a lone sentence letter) and has no connectives at all. Therefore it has no main connective.

Inheritance Step: Assume that the theorem holds for every sentence with order i or less, and that ϕ is of order $i + 1$, where $i \geq 1$. (This is the recursive assumption.)

By the definition of an SL sentence (2.2), ϕ will have to have one of the following forms: $\sim\psi$, $(\theta_1 \rightarrow \theta_2)$, $(\theta_1 \leftrightarrow \theta_2)$, $(\theta_1 \wedge \dots \wedge \theta_n)$, or $(\theta_1 \vee \dots \vee \theta_n)$. For each of these possible sentence schemas for ϕ , there is at least one connective that isn't in any proper subsentence of ϕ . It follows that ϕ must have *at least* one main connective token.

Negation: Assume (for *reductio*) that ϕ has more than one main connective token, and that the leftmost of these tokens is a negation. Given that the leftmost is a ' \sim ' token, ϕ must be of the form $\sim\psi$ (by def. of SL sentence, 2.2). If this is so, then the other main connective tokens of ϕ must be in ψ . But ψ is a proper subsentence of ϕ , so any connective in ψ can't be a main connective of ϕ (def. of main connective). Therefore, contrary to our assumption for *reductio*, there must be exactly one main connective token in ϕ : the ' \sim '.

Conditional: Assume (again, for *reductio*) that ϕ has more than one main connective token, and that the leftmost of these is a ' \rightarrow '. By the definition of SL sentence (2.2), it follows that ϕ must be of the form $\theta_1 \rightarrow \theta_2$. Accordingly, any other main connective tokens of ϕ must be in θ_2 . However, θ_2 is a proper subsentence of ϕ , and thus can't contain a main connective of ϕ . So, contrary to our assumption for *reductio*, there must be exactly one main connective token for ϕ : the ' \rightarrow '.

Biconditional: This clause is the same as the ‘Conditional’ clause above, except with ‘ \leftrightarrow ’ in place of ‘ \rightarrow ’.

Conjunction: Assume that the leftmost main connective token in ϕ is a ‘ \wedge ’. Assume (for *reductio*) that there is some main connective of ϕ other than ‘ \wedge ’. According to the definition of SL sentence (2.2), ϕ must be of the form $(\theta_1 \wedge \theta_2 \wedge \theta_3 \wedge \dots \wedge \theta_n)$, where n is the number of conjuncts. It follows that the non-‘ \wedge ’ main connective token must be in one of the conjuncts of ϕ . But the conjuncts of ϕ are proper subsentences of ϕ , and so can’t contain any main connective tokens. So, contrary to our assumption for *reductio*, all of the main connective tokens of ϕ are ‘ \wedge ’.

Given that ϕ is of the form $(\theta_1 \wedge \theta_2 \wedge \theta_3 \wedge \dots \wedge \theta_n)$, n is the number of conjuncts. We want to show that the number of main connective tokens, k , is $n - 1$. Assume that $k > n - 1$ ¹⁷. But if $k > n - 1$ then there must be two ‘ \wedge ’ tokens adjacent to each other; ϕ is an SL sentence, so this is impossible. Assume instead that $k < n - 1$ ¹⁸. But then there must be two conjuncts adjacent to each other, not separated by a connective. Again, ϕ is an SL sentence, so this is impossible. Therefore $k = n - 1$. So, when n is the number of conjuncts in ϕ , $n - 1$ is the number of main connective tokens.

There must be at least two conjuncts in ϕ (definition of SL sentence). Assuming only two conjuncts in ϕ , there is exactly one main connective token. Alternatively, when the number of conjuncts, n , is greater than two, there are $n - 1$ main connective tokens and ϕ is an extended conjunction.

Disjunction: This clause is the same as the ‘Conjunction’ clause above, except with ‘ \vee ’ in place of ‘ \wedge ’.

Closure Step: We’ve shown that sentences with order 1 have no main connective, and that every sentence of order of 2 or greater either has exactly one main connective token, or is an extended conjunction (or disjunction) with n main connective tokens, where $n + 1$ is the number of conjuncts (disjuncts). There is no other way to construct an SL sentence than the ways described in the previous two clauses.

■

2.6 Disjunctive Normal Form

Which sentences below are easy to evaluate? Which are difficult?

¹⁷I.e., that the number of main connective tokens is greater than (the number of conjuncts, minus one)....

¹⁸I.e., that the number of main connective tokens is fewer than (the number of conjuncts, minus one)....

12. $\sim(A \rightarrow \sim(C \wedge B)) \rightarrow (A \rightarrow C)$

14. $\sim(\sim(A \rightarrow \sim R) \rightarrow (A \rightarrow R))$

13. $A \wedge R \wedge A \wedge \sim R$

15. $(\sim A \vee \sim C \vee \sim B) \vee (\sim A \vee C)$

If we have the truth values of the sentence letters, then clearly 13 and 15 are much simpler to figure out than 12 and 14. In general, we find that the truth values for certain sentences are easier to calculate than others. (Some sentences, we might say, are more “transparent”). Say that we want to figure out the truth value of a difficult sentence and we know that it’s TFE to an easy sentence. Then we could figure out the truth value of the difficult one by figuring out the truth value of the easy one. After all, they’re equivalent! In the above list of sentences, 12 is TFE to 15 and 13 is TFE to 14.

If a sentence is fairly complicated we might not know whether it’s TFE to a simpler, more transparent sentence. To get around this we can simplify the complicated sentence by a systematic series of steps, each of which replaces some subsentence with a simpler sentence that we know is TFE to the subsentence being replaced.

2.6.1 The TFE Replacement Theorem

First we have to verify that equivalence transformation is legitimate, so we will prove:

Theorem 2.70. Truth Functional Equivalence Replacement: *Let ϕ be a subsentence of θ . If ϕ and ϕ^* are truth functionally equivalent, and θ^* is the result of replacing one occurrence of ϕ by ϕ^* in θ , then θ and θ^* are truth functionally equivalent.*

Before turning to the proof, recall from section 2.3.4, definition 2.50 (on page 35) that two sentences ϕ and ϕ^* are truth functionally equivalent iff all models assign them the same truth value. This is the same as saying they entail each other, i.e., that $\phi^* \models \phi$ and $\phi \models \phi^*$.

Proof:

Base Step: Suppose that θ is an atomic sentence. In this case θ and ϕ are the same. So, θ^* and ϕ^* are the same. Thus, by hypothesis (the hypothesis being that ϕ and ϕ^* are truth functionally equivalent), θ^* and θ are truth functionally equivalent.

Inheritance Step: For this proof we must consider each connective separately.

Negation: Suppose θ is a negation $\sim\psi$, for some sentence ψ which either is identical to ϕ , or of which ϕ is a subsentence.

Assume that ψ and ψ^* (ψ^* the result of replacing at least one occurrence of ϕ with ϕ^* in ψ) are truth functionally equivalent, i.e. have the same truth value on every model, and are of order k or less. (This is our recursive assumption.) Now, by supposition θ^* is the same sentence as $(\sim\psi)^*$, which with a little thought one can see is the same sentence as $\sim(\psi^*)$.

Next, note that by the definition of true in a model, $\sim(\psi^*)$ is true in a model \mathbf{m} iff ψ^* is false in \mathbf{m} iff ψ is false in \mathbf{m} iff $\sim\psi$ is true in \mathbf{m} . So, θ^* is true in \mathbf{m} iff $\sim\psi$ is true in \mathbf{m} .

So, θ^* is true in \mathbf{m} iff θ is true in \mathbf{m} , so by definition θ and θ^* are truth functionally equivalent.

Conditional: Suppose θ is a conditional $(\psi_1 \rightarrow \psi_2)$, where at least one of ψ_1 and ψ_2 has ϕ as a subsentence, or is identical to ϕ . So θ^* is $(\psi_1 \rightarrow \psi_2)^*$, which with some thought one can see is either (case 1) $(\psi_1^* \rightarrow \psi_2)$ or (case 2) $(\psi_1 \rightarrow \psi_2^*)$.

Suppose it is the first case, and assume that ψ_1 is truth functionally equivalent to ψ_1^* , and that both are of order k or less. (This is our recursive assumption.)

We know that $(\psi_1^* \rightarrow \psi_2)$ is true in a model \mathbf{m} iff ψ_1^* is false in \mathbf{m} or ψ_2 is true in \mathbf{m} . But that holds iff ψ_1 is false in \mathbf{m} or ψ_2 is true in \mathbf{m} , and that holds iff $(\psi_1 \rightarrow \psi_2)$ is true in \mathbf{m} .

In the first case, θ^* is true in \mathbf{m} iff θ is true in \mathbf{m} , so by definition θ and θ^* are truth functionally equivalent.

Showing that θ and θ^* are truth functionally equivalent in case 2 is left to the reader.

Biconditional: Showing that θ and θ^* are truth functionally equivalent when θ is a biconditional of the form $(\psi_1 \leftrightarrow \psi_2)$ is left to the reader.

Conjunction: Suppose that θ is a conjunction $(\psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \dots \wedge \psi_n)$, where at least one of $\psi_1, \psi_2, \psi_3, \dots, \psi_n$ has ϕ as a subsentence, or is identical to ϕ .

So θ^* is $(\psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \dots \wedge \psi_n)^*$. Since θ^* is the result of replacing *one* occurrence of ϕ in θ with ϕ^* , it's not hard to see that $(\psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \dots \wedge \psi_n)^*$ can *only* one of either $(\psi_1^* \wedge \psi_2 \wedge \dots \wedge \psi_n)$ or $(\psi_1 \wedge \psi_2^* \wedge \dots \wedge \psi_n)$ or ... or $(\psi_1 \wedge \psi_2 \wedge \dots \wedge \psi_n^*)$. But, clearly, there's no difference which it is. So, without loss of generality say that θ^* is $(\psi_1 \wedge \psi_2^* \wedge \psi_3 \wedge \dots \wedge \psi_n)$.

As before, assume that ψ_2 and ψ_2^* are truth functionally equivalent, and are of order k or less. (This is our recursive assumption.) So, for any model \mathbf{m} , θ is true in \mathbf{m} iff $(\psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \dots \wedge \psi_n)$ is true in \mathbf{m} iff $(\psi_1 \wedge \psi_2^* \wedge \psi_3 \wedge \dots \wedge \psi_n)$ is true in \mathbf{m} iff $(\psi_1 \wedge \psi_2 \wedge \psi_3 \wedge \dots \wedge \psi_n)^*$ is true in \mathbf{m} iff θ^* is true in \mathbf{m} .

Therefore θ and θ^* are truth functionally equivalent.

Disjunction: The disjunction case is similar and it is left to the reader.

Closure Step: Those are the only ways SL sentences can be formed; hence the theorem is proved. ■

Example 2.71. We can use this theorem to show that 12 and 15 above are equivalent, as are 13 and 14. Consider 12 and 15. (We leave 13 and 14 to the reader.) For any formulas ϕ and θ ,

16. $(\phi \rightarrow \theta)$ and $(\sim \phi \vee \theta)$ are TFE (See ex. 2.59, on page 37)

- 17. $\sim\sim\phi$ and ϕ are TFE (See ex. 2.60, on page 37)
- 18. $\sim(\theta \wedge \phi)$ and $(\sim\theta \vee \sim\phi)$ are TFE (See 4 and 12, section 2.8.6)
- 19. $(\phi \vee \theta \vee \psi)$ and $(\phi \vee (\theta \vee \psi))$ are TFE (Obvious)

By making substitutions starting with 12 that the theorem says result in successive truth functionally equivalent sentences, we can get from 12 to 15 and thereby have shown that 15 is truth functionally equivalent to 12.

- 20. $\sim(A \rightarrow \sim(C \wedge B)) \rightarrow (A \rightarrow C)$ [12]
- 21. $\sim\sim(A \rightarrow \sim(C \wedge B)) \vee (A \rightarrow C)$ [16]
- 22. $(A \rightarrow \sim(C \wedge B)) \vee (A \rightarrow C)$ [17]
- 23. $(\sim A \vee \sim(C \wedge B)) \vee (\sim A \vee C)$ [16]
- 24. $(\sim A \vee (\sim C \vee \sim B)) \vee (\sim A \vee C)$ [18]
- 25. $(\sim A \vee \sim C \vee \sim B) \vee (\sim A \vee C)$ [19]

2.6.2 Disjunctive Normal Form

Sentences in disjunctive normal form are especially easy to evaluate.

Definition 2.72. A SL sentence is in *disjunctive normal form* (DNF) iff

- (1) it contains no conditional (\rightarrow) or biconditional (\leftrightarrow),
- (2) negations (\sim) only govern sentence letters, and
- (3) no conjunction (\wedge) contains a disjunction (\vee) as a subsentence.

A typical example of a sentence in DNF is $(Q \wedge \sim R) \vee (\sim P \wedge R)$. The truth conditions for this sentence are easy to see. The sentence $(Q \wedge \sim R) \vee (\sim P \wedge R)$ is true on a model \mathbf{m} when either $\mathbf{m}(Q) = \mathbf{T}$ and $\mathbf{m}(R) = \mathbf{F}$, or $\mathbf{m}(P) = \mathbf{F}$ and $\mathbf{m}(R) = \mathbf{T}$. Some less typical examples are:

- 26. Q
- 27. $\sim R$
- 28. $Q \wedge \sim R$
- 29. $\sim Q \vee R$

DNF is important because we can prove the following theorem.

Theorem 2.73. The Disjunctive Normal Form Theorem: *Every sentence of SL is truth functionally equivalent to an SL sentence which is in DNF.*

Proof: The proof relies on three lemmas, each of which can be established rigorously by recursive proof. We leave the details of these lemmas to the reader.

Here we provide a process showing how to turn any given sentence into one that's in DNF. We proceed in three stages, corresponding to the three lemmas that are necessary for a proof.

Step A: If a subsentence of ϕ has a conditional or biconditional as its main connective, i.e., is of the form $(\psi \rightarrow \theta)$ or $(\theta \leftrightarrow \psi)$, replace the subsentence by $(\sim\psi \vee \theta)$ or $(\psi \wedge \theta) \vee (\sim\psi \wedge \sim\theta)$ respectively. Repeat as necessary to obtain a sentence ϕ' without conditionals or biconditionals.

Step B:

- (1) Replace any subsentence of the form $\sim\sim\psi$ in ϕ' with ψ .
- (2) Replace any subsentence of the form $\sim(\psi \wedge \theta)$ in ϕ' with $(\sim\psi \vee \sim\theta)$.
- (3) Replace $\sim(\psi \vee \theta)$ in ϕ' with $(\sim\psi \wedge \sim\theta)$.

Repeat as necessary to obtain ϕ'' in which negations govern nothing but sentence letters.

Step C: The only thing that could prevent ϕ'' from being in DNF is that some conjunctions govern some disjunctions, i.e., there is a subsentence $\theta \wedge (\psi_1 \vee \psi_2 \vee \dots \vee \psi_n)$, or the reverse $(\psi_1 \vee \psi_2 \vee \dots \vee \psi_n) \wedge \theta$. Those subsentences can be replaced by the equivalent $(\psi_1 \wedge \theta) \vee (\psi_2 \wedge \theta) \vee \dots \vee (\psi_n \wedge \theta)$. Repeat as necessary.

■

A recursive proof would be more rigorous—it would have a clause for each SL connective, and would explain in each clause how to construct from each subsentence another TFE subsentence that is in DNF.

DNF sentences allow us see some of the advantages of formal languages. For instance, we can construct a simple, mechanical process that will tell us when a DNF sentence is TFF.

Let ϕ be some DNF sentence, and let $\theta_1, \theta_2, \theta_3, \dots$, and θ_n each be the disjuncts of ϕ . We can see that ϕ is TFF iff every disjunct θ_i is TFF. Because each θ_i is a conjunction with negated and unnegated sentence letters as the conjuncts, there is only one way that it can be TFF. A θ_i is TFF iff it has some sentence letter ψ as one conjunct and $\sim\psi$ as another.

It follows that we can use the following process to determine whether a DNF sentence ϕ is TFF. Check every disjunct of ϕ to see if it has some ψ and $\sim\psi$ as conjuncts. If so, then ϕ is TFF; otherwise it isn't. For example, consider the following DNF sentence:

$$(Q \wedge R \wedge \sim Q) \vee (\sim Q \wedge R \wedge R) \vee (O \wedge R \wedge \sim O)$$

The first disjunct, $(Q \wedge R \wedge \sim Q)$, is TFF because it has Q and $\sim Q$ as conjuncts; the third disjunct, $(O \wedge R \wedge \sim O)$, is also TFF. But the second disjunct, $(\sim Q \wedge R \wedge R)$, isn't TFF. So, the whole sentence isn't TFF.

If we were to replace the second disjunct with $(\sim Q \wedge R \wedge \sim R)$, so that the new whole sentence is:

$$(Q \wedge R \wedge \sim Q) \vee (\sim Q \wedge R \wedge \sim R) \vee (O \wedge R \wedge \sim O)$$

...then the result *is* TFF, because each disjunct has a sentence letter and its negation as conjuncts.

We can now construct a mechanical method that can determine whether *any* SL sentence ϕ is TFF. First, we use the process in 2.73 to construct an equivalent DNF sentence, ϕ^* . Then we use the method given above to determine whether ϕ^* is TFF. Because they are equivalent, ϕ^* is TFF iff ϕ is TFF. No creativity is needed to apply this method—each individual step requires nothing more than following simple instructions.

We can even extend this method to determine whether any SL sentence is TFT. We will take advantage of the fact that if you put a negation in front of a TFT sentence ϕ , the resulting sentence, $\sim\phi$, is TFF. So, all that is necessary to see whether ϕ is TFT is to negate ϕ , put $\sim\phi$ into DNF, and then to see whether the final result is TFF. If so, then ϕ is TFT! If not, then ϕ isn't. As before, this process is entirely mechanical or *formal*. Note that, along with the truth table method in section 2.2.4, we now have two rather different formal methods for determining logical truth in SL. In later chapters our methods will be closer to the DNF approach.

Any given sentence of SL is truth functionally equivalent to more than one DNF sentence. A sentence has DNFs that differ slightly for at least two reasons. First, for any sentence ϕ and sentence letter ψ , if ϕ is in DNF, then $\phi \vee (\psi \wedge \sim\psi)$ is TFE to ϕ and also in DNF. Second, sometimes a sentence in DNF can be simplified. Thus

$$30. (Q \wedge R \wedge O) \vee (Q \wedge R \wedge N) \vee (Q \wedge R \wedge \sim O)$$

can be simplified to the DNF

$$31. (Q \wedge R) \vee (Q \wedge R \wedge N)$$

and further to

$$32. (Q \wedge R).$$

2.7 Truth Functional Expressiveness

The definition of truth in a model (def. 2.21, on page 23) associates each of the logical connectives of SL with a truth function.¹⁹ We can think of the logical connectives of SL as truth functions—i.e., as having a meaning that's exhausted by the definition of truth. One might ask whether the five logical connectives of SL cover all the possible logical connectives.

In one sense it's obvious that they do not. For example, we could introduce a new connective, %, choose some number of places for it, and give some clause that describes how the truth value of a sentence with % as the main connective depends on the truth value of the component parts. As long as this clause differs from any of those in the definition of truth, % is distinct from the five in SL.

¹⁹See section 2.2.2 for more details.

But although the five logical connectives of SL obviously do not exhaust all the possible connectives, there's still a sense in which they might indirectly cover them all. Even if $\%$ is distinct from all the connectives of SL, maybe there's still some sentence schema that is truth functionally equivalent to $\%$, and that uses only (but not necessarily all of) the five connectives of SL. For example, say $\%$ is a 3-place connective, such that a sentence $\theta_1 \% \theta_2 \% \theta_3$ is true on a model \mathbf{m} iff at least two of the θ are true on \mathbf{m} . So, the sentence is true if θ_1 and θ_2 are true, or if θ_1 and θ_3 are true, or if θ_2 and θ_3 are true. We can express this without ' $\%$ ', using \wedge for 'and' and \vee for 'or': $(\theta_1 \wedge \theta_2) \vee (\theta_1 \wedge \theta_3) \vee (\theta_2 \wedge \theta_3)$. Even though $\%$ is a connective that isn't in our language, we can use the connectives of SL to construct a truth functionally equivalent sentence.

A logical connective $\%$ is *definable* in terms of some set of other logical connectives Δ iff there's some sentence schema using only connectives from Δ that's truth functionally equivalent to $\%$. With this in mind, we can ask whether every logical connective is definable in terms of the five connectives of SL. We can also ask if any of the connectives of SL are definable in terms of the others. We answer the first of these questions with theorem 2.78, on the following page. The second we consider now in the following examples.²⁰

Example 2.74. We can define conjunction using negation and disjunction.

Proof: Any sentence $\phi_1 \wedge \dots \wedge \phi_n$ is TFE to the sentence $\sim(\sim\phi_1 \vee \dots \vee \sim\phi_n)$. ■

Example 2.75. We can define disjunction using negation and conjunction.

Proof: Any sentence $\phi_1 \vee \dots \vee \phi_n$ is TFE to the sentence $\sim(\sim\phi_1 \wedge \dots \wedge \sim\phi_n)$. ■

Example 2.76. We can define conditional using negation and disjunction.

Proof: Any sentence $\phi_1 \rightarrow \phi_2$ is TFE to the sentence $\sim\phi_1 \vee \phi_2$. ■

Example 2.77. The pairs \sim and \wedge , and \sim and \vee are each adequate to define the remaining connectives in SL.

Proof: By example 2.75, with \sim and \wedge we can define \vee . By example 2.59, $\phi \rightarrow \theta$ and $\sim\phi \vee \theta$ are TFE. So we can define \rightarrow with \sim and \wedge . As the reader can check, $\phi \leftrightarrow \psi$ is TFE to $(\phi \rightarrow \psi) \wedge (\psi \rightarrow \phi)$. So we can define \leftrightarrow with \sim and \wedge .

We leave it to the reader to show that \sim and \vee are adequate to define the remaining connectives in SL. ■

We don't actually need all five connectives, but for the sake of convenience we keep them all. We also asked the following: Are the five operations we have enough? That is, are there other logical operations we can't express and should add notation for?

²⁰See (Post 1921, Hodges 2001b: 17).

It turns out that our connectives are adequate. There are no other logical operations we can't express. Although it does not strictly depend on the DNF theorem, the idea behind that theorem lets us prove this.

Theorem 2.78. The Truth-functional Expressive Completeness Theorem: *Any truth-functional connective of any fixed number of arguments (ternary, quaternary, etc.) is already expressible in SL.*

Proof: Any truth functional connective of a fixed number of arguments assigns T or F depending only on the values of the components, so it can be exactly described by a truth table. For example, consider the 4-place operation % given by truth table 2.2 below. We could attempt to find a complicated sentence in terms of various con-

ϕ_1	ϕ_2	ϕ_3	ϕ_4	$\%(\phi_1, \phi_2, \phi_3, \phi_4)$
T	T	T	T	T
T	T	T	F	F
T	T	F	T	T
T	T	F	F	F
T	F	T	T	F
T	F	T	F	T
T	F	F	T	F
T	F	F	F	F
F	T	T	T	T
F	T	T	F	F
F	T	F	T	F
F	T	F	F	F
F	F	T	T	F
F	F	T	F	F
F	F	F	T	T
F	F	F	F	F

Table 2.2: Truth Table for %

nectives that would express this, but it will be better for our purposes to construct a DNF equivalent systematically. We know from the first line that the expression $\%(\phi_1, \phi_2, \phi_3, \phi_4)$ is true when all components are, that is, if $(\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4)$ is true; we know from the third line it is true when the first two, ϕ_1 and ϕ_2 , and fourth, ϕ_4 , are true and the third, ϕ_3 , false, i.e., $(\phi_1 \wedge \phi_2 \wedge \sim \phi_3 \wedge \phi_4)$. We also know it is true when only the first, ϕ_1 , and third, ϕ_3 , are true, i.e., $(\phi_1 \wedge \sim \phi_2 \wedge \phi_3 \wedge \sim \phi_4)$, when the first, ϕ_1 , is false and the other three, ϕ_2 , ϕ_3 , and ϕ_4 , are true i.e., $(\sim \phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4)$ and when all but the fourth, ϕ_4 , are false, i.e., $(\sim \phi_1 \wedge \sim \phi_2 \wedge \sim \phi_3 \wedge \phi_4)$. Because each of these conjunctions is true exactly when the corresponding line is true the whole sentence will be true when any one of them is true, i.e., it is equivalent to the formula:

$$33. (\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4) \vee (\phi_1 \wedge \phi_2 \wedge \sim\phi_3 \wedge \phi_4) \vee (\phi_1 \wedge \sim\phi_2 \wedge \phi_3 \wedge \sim\phi_4) \vee (\sim\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4) \vee (\sim\phi_1 \wedge \sim\phi_2 \wedge \sim\phi_3 \wedge \phi_4)$$

We could simplify this sentence further, but that is not important for our purposes. We now can see how to read off from any truth table for any operation on any fixed number of sentences a DNF representation that is equivalent. Our five connectives are enough. ■

Since either of the pairs ‘ \sim ’ and ‘ \wedge ’, or ‘ \sim ’ and ‘ \vee ’ are adequate to define the remaining connectives in SL (see ex. 2.74, on page 51) we can see that either of those pairs is adequate to define all truth-functional connectives. We can even improve on that though, because there are two connectives either of which would be adequate all by itself. One is the Sheffer stroke, named after the logician who first demonstrated its properties and the symbol he used, ‘|’, but it is sometimes called NAND. Its definition is that $(\phi_1|\phi_2|\dots|\phi_n)$ is true iff at least one component is false. So, $(\phi_1|\phi_1)$ is equivalent to $\sim\phi$. And $(\sim\phi_1|\sim\phi_2|\dots|\sim\phi_n)$ is true just in case at least one component is false. That means at least one ϕ_i is true, which is to say that the sentence is equivalent to a disjunction. The other connective that is adequate by itself is NOR, which is defined to be true iff all the components are false. It is left to the reader as an optional exercise to show how to define the other connectives using NOR.

Thus, to get a language just as expressive as SL, we only need one logical connective (either NAND or NOR), not five. If we had a taste for cutting down basic symbols, we could go further and generate an infinite set of sentence letters by using just one symbol, ‘A’, and generating new sentence letters by concatenating prime marks ‘ $'$ ’ to it. Then all we need are parentheses (though there are ways to do without these too). Such a language is sparse, but it is just as expressive as SL. Most people would find such a language difficult to work with, but computers love them.

2.8 Exercises

2.8.1 Recursive Definition Problems

1. Although it's not framed as one, definition 2.9 (on page 19) of order is a recursive definition. Rewrite it so that the base, generating, and closure clauses are explicit.
2. Although we don't give a recursive definition, a recursive definition can be given for the unofficial SL sentences. (Definition 2.4, on page 17 is the definition we give.) Write down a recursive definition for unofficial SL sentences.

2.8.2 Construction Trees

Write the construction tree for each of the following SL sentences.

1. $\sim\sim\sim B$
2. $\sim(B \vee (A \rightarrow A))$
3. $(\sim B \vee (A \rightarrow A))$
4. $((A \wedge B) \leftrightarrow A) \rightarrow \sim(B \rightarrow C)$
5. $((A \leftrightarrow B) \wedge A) \rightarrow \sim(B \rightarrow C)$
6. $(P \wedge (Q \wedge R))$
7. $((P \wedge Q) \wedge R)$
8. $((P \wedge \sim R) \rightarrow \sim Q)$
9. $(P \wedge (\sim R \rightarrow \sim Q))$
10. $\sim((P \rightarrow Q) \vee (P \rightarrow Q))$

2.8.3 Official and Unofficial Sentences

Which of these are official sentences? Which are unofficial? Which are neither official nor unofficial sentences (i.e., not a sentence in any sense)? If neither, how could you make it either an official or unofficial sentence? Note: there might be multiple different ways to make it an official or unofficial sentence. Finally, if it's a sentence (official or unofficial), then give its order and the number of subsentences in it.

1. $(A \rightarrow B \wedge C)$
2. $(A \rightarrow (B \rightarrow C))$
3. $A \rightarrow (B \wedge C \wedge B)$
4. $(A \rightarrow (\theta \wedge C))$
5. $(A \rightarrow (B \wedge C \vee D))$
6. $(A \rightarrow (Z \wedge C))$
7. $(\sim A \rightarrow (B_{374} \wedge C))$
8. $(A \rightarrow (B \wedge C))$
9. $(\sim(B \wedge C))$
10. $(B \wedge \sim\sim M \wedge D)$

2.8.4 Truth in a Model

Consider the model \mathbf{m}_1 such that $\mathbf{m}_1(A) = \text{T}$, $\mathbf{m}_1(B) = \text{F}$, $\mathbf{m}_1(C) = \text{T}$, $\mathbf{m}_1(D) = \text{F}$, and $\mathbf{m}_1(E) = \text{T}$; and the model \mathbf{m}_2 such that $\mathbf{m}_2(A) = \text{T}$, $\mathbf{m}_2(B) = \text{T}$, $\mathbf{m}_2(C) = \text{F}$, $\mathbf{m}_2(D) = \text{F}$, and $\mathbf{m}_2(E) = \text{F}$. Give the truth values of each of the following SL sentences on each of these two models.

1. $(A \vee B) \rightarrow (C \wedge D)$
2. $(A \vee B) \rightarrow (C \wedge \sim D)$
3. $(C \wedge D) \rightarrow (A \vee B)$
4. $(A \rightarrow C) \wedge E$
5. $\sim(B \rightarrow E) \wedge D$
6. $\sim(A \vee B) \vee (A \wedge B \wedge (E \rightarrow E))$
7. $A \vee (B \rightarrow (E \rightarrow E))$
8. $(A \wedge E) \vee (\sim D \wedge C) \vee (B \wedge \sim A)$

2.8.5 TFT, TFF, and TFC

For each of the following say whether the sentence is TFC, TFF or TFT. If it is TFC, give a model which makes the sentence true and another model which makes it false. If it is TFF, justify your answer without truth tables (i.e., explain why there is no

model which makes the sentence true). If it is TFT, again justify your answer without truth tables (i.e., explain why every model makes the sentence true).

1. $(A \rightarrow B) \rightarrow (\sim B \vee \sim A)$
2. $(A \wedge B) \rightarrow (A \leftrightarrow B)$
3. $(\sim A \vee \sim B) \rightarrow \sim(A \wedge B)$
4. $A \vee (A \rightarrow B)$
5. $\sim(A \vee B) \rightarrow (\sim A \wedge \sim B)$
6. $\sim(A \leftrightarrow B) \rightarrow (\sim A \leftrightarrow B)$
7. $(A \wedge (B \vee C)) \rightarrow ((A \wedge B) \vee C)$
8. $\sim A \rightarrow (A \rightarrow B)$
9. $(\sim A \wedge \sim B) \rightarrow \sim(A \vee B)$
10. $A \rightarrow (A \rightarrow B)$
11. $(A \leftrightarrow B) \rightarrow (A \wedge B)$
12. $\sim(A \rightarrow B) \rightarrow A$
13. $\sim(A \wedge B) \rightarrow (\sim A \vee \sim B)$
14. $A \rightarrow (B \rightarrow A)$
15. $(A \rightarrow B) \rightarrow (A \wedge \sim B)$
16. $\sim(A \vee (A \rightarrow B))$

2.8.6 Entailment Problems for SL

For each of the following, without using truth tables show whether or not the entailment holds. There are a number of different methods for thinking through these problems. Remember that an entailment means that on all \mathbf{m} if the LHS is true then the RHS is also true. One approach is to show that making the LHS true forces the RHS to be true. Another is to show that making the RHS false forces the LHS to be false. Both are examples of arguing that it is not possible for the LHS to be true and the RHS false. Another method is showing that all \mathbf{m} either make the LHS false or the RHS true. If the entailment does not hold, you can show this by providing a counterexample.

1. $\sim A \models (A \rightarrow B)$
2. $(A \rightarrow B) \models (\sim B \vee \sim A)$
3. $(\sim A \wedge \sim B) \models \sim(A \vee B)$
4. $(\sim A \vee \sim B) \models \sim(A \wedge B)$
5. $A \models (A \rightarrow B)$
6. $\sim(A \leftrightarrow B) \models (\sim A \leftrightarrow B)$
7. $(A \wedge (B \vee C)) \models ((A \wedge B) \vee C)$
8. $\sim(A \vee B) \models (\sim A \wedge \sim B)$
9. $A \models (B \rightarrow A)$
10. $(A \wedge B) \models (A \leftrightarrow B)$
11. $\sim(A \rightarrow B) \models (B \rightarrow A)$
12. $\sim(A \wedge B) \models (\sim A \vee \sim B)$
13. $\sim(A \rightarrow B) \models A$
14. $(A \leftrightarrow B) \models (A \wedge B)$

2.8.7 More Entailment Problems for SL

Show whether, for any SL sentence ϕ , θ , and ψ , each of the following statements is true.

1. $\models (\phi \rightarrow \theta) \vee (\theta \rightarrow \phi)$
2. Either $\models (\phi \rightarrow \theta)$, or $\models (\theta \rightarrow \phi)$
3. If $\models (\phi \rightarrow \theta)$ and $\models \phi$, then $\models \theta$
4. If $\models (\phi \rightarrow \theta)$ and $\models \sim \phi$, then $\models \sim \theta$
5. $\models (\phi \rightarrow \theta) \vee (\theta \rightarrow \psi)$
6. If $(\phi \wedge \theta) \models \psi$, then both $\phi \models \psi$ and $\theta \models \psi$
7. If $\psi \models (\phi \wedge \theta)$, then both $\psi \models \phi$ and $\psi \models \theta$

2.8.8 Even More Entailment Problems: Truth-preservation Lemma

Show that, for any SL sentence ϕ , θ , and ψ , each of the following entailments holds. Showing that these entailments hold will be helpful later, since they are needed in the proof of theorems 7.4 (on page 181) and 6.6 (on page 156).

1. $\phi \models \phi$
2. $\theta \rightarrow \psi, \theta \models \psi$
3. $\theta \wedge \psi \models \psi$
4. $\theta \wedge \psi \models \theta$
5. $\theta, \psi \models \theta \wedge \psi$
6. $\theta \vee \psi, \theta \rightarrow \phi, \psi \rightarrow \phi \models \phi$
7. $\theta \models \theta \vee \psi$
8. $\theta \rightarrow (\psi \wedge \sim \psi) \models \sim \theta$
9. $\sim \theta \rightarrow (\psi \wedge \sim \psi) \models \theta$
10. $\theta \rightarrow \psi, \psi \rightarrow \theta \models \theta \leftrightarrow \psi$
11. $\theta \leftrightarrow \psi, \psi \models \theta$
12. $\theta \leftrightarrow \psi, \theta \models \psi$

13. $\psi \rightarrow \theta, \sim\theta \models \sim\psi$

16. $\theta, \sim\theta \models \psi$

14. $\psi \vee \theta, \sim\theta \models \psi$

15. $\theta \vee \psi, \sim\psi \models \theta$

17. $\psi \leftrightarrow \theta \models \sim\psi \leftrightarrow \sim\theta$

2.8.9 Truth Functional Equivalence

Without using truth tables, show that each of the following pairs of sentences is TFE, for any sentences got by substituting into the schemas. Showing that these pairs are TFE will be helpful later, since it's both needed for the proof of theorem 6.6 (on page 156) and it amounts to proving theorem 6.9 (on page 158) (including for \leftrightarrow -Exchange), which is needed to prove theorem 6.8 (on page 157).

1. $\sim(\phi_1 \wedge \dots \wedge \phi_n), \sim\phi_1 \vee \dots \vee \sim\phi_n$

2. $\sim(\phi_1 \vee \dots \vee \phi_n), \sim\phi_1 \wedge \dots \wedge \sim\phi_n$

3. $\sim\sim\phi, \phi$

4. $\phi \rightarrow \theta, \sim\phi \vee \theta$

5. $\phi \rightarrow \theta, \sim\theta \rightarrow \sim\phi$

6. $\sim(\phi \rightarrow \theta), \phi \wedge \sim\theta$

7. $\theta \wedge (\phi_1 \vee \dots \vee \phi_n), (\theta \wedge \phi_1) \vee \dots \vee (\theta \wedge \phi_n)$

8. $(\phi_1 \vee \dots \vee \phi_n) \wedge \theta, (\phi_1 \wedge \theta) \vee \dots \vee (\phi_n \wedge \theta)$

9. $\theta \vee (\phi_1 \wedge \dots \wedge \phi_n), (\theta \vee \phi_1) \wedge \dots \wedge (\theta \vee \phi_n)$

10. $(\phi_1 \wedge \dots \wedge \phi_n) \vee \theta, (\phi_1 \vee \theta) \wedge \dots \wedge (\phi_n \vee \theta)$

11. $\theta \leftrightarrow \psi, (\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)$

2.8.10 Relations Between SL Sentences

What relations hold among these sentences? Specifically, say whether they are contradictory, contrary, subcontrary, independent, or truth functionally equivalent. (So, you need to supply 30 answers for each problem: for each sentence ϕ , you must determine for each of the 5 relations whether it holds between ϕ and each of the 6 other sentences.)

1. A

4. $A \rightarrow C$

2. $A \wedge B$

5. $A \rightarrow \sim C$

3. $\sim A \wedge B$

6. $(A \wedge B) \vee C$

7. $D \wedge \sim D$

2.8.11 Recursive Definitions and Proofs

1. Give a recursive definition of the even positive integers.
2. Give a recursive definition of the odd positive integers that are greater than 100.
3. Prove that all positive multiples of 10 are also multiples of 5.
4. For each even positive integer n , prove that dividing n by 2 results in another positive integer.

2.8.12 SL Recursive Proofs

Prove each of the following claims using a recursive proof.

1. In every SL sentence which is TFF, there is a subsentence which is TFC. *Hint:* The proof and the basic idea behind it is very easy; don't over-think it.
2. If two models \mathbf{m} and \mathbf{m}' agree on all of the sentence letters of ϕ , then ϕ is true in \mathbf{m} iff ϕ is true in \mathbf{m}' . (This is theorem 2.68, on page 42.) To be explicit, we say that \mathbf{m} and \mathbf{m}' agree on the sentence letters of ϕ iff they assign the same truth value to each of those sentence letters, i.e. both assign true or both assign false. We say that \mathbf{m} and \mathbf{m}' agree on a sentence iff they assign the same truth value to the sentence.
3. Show that every TFF sentence of SL contains at least one ' \sim '. *Hint:* To prove this, it is useful to prove a more specific statement, namely that if ϕ is an SL sentence that does not contain any negations then ϕ is true on the model that assigns true to all sentence letters.
4. For every sentence ϕ of SL, the number of left parentheses occurring in ϕ is less than the number of subsentences. In other words, if $\text{LP}\phi$ is the number of left parentheses in ϕ and $\text{SS}\phi$ is the number of subsentences, then $\text{LP}\phi < \text{SS}\phi$.
5. The number of subsentences in any official SL sentence ϕ is equal to: the number of tokens of sentence letters in ϕ plus the number of tokens of negation in ϕ plus the number of tokens of left parentheses in ϕ .
6. For every sentence ϕ of SL, there is a TFE sentence ϕ' without conditionals or biconditionals.

2.8.13 DNF

Put the following into disjunctive normal form.

1. $\sim(Q \rightarrow R) \vee (Q \rightarrow \sim R)$

2. $O \wedge (O \rightarrow Q)$

3. $((Q \rightarrow R) \rightarrow Q) \rightarrow Q$

4. $\sim(Q \rightarrow R) \wedge (O \vee P)$

Chapter 3

Quantifier Language I

3.1 The Language QL1

3.1.1 Sentences of QL1

SL allows us to investigate certain aspects of logical consequence and English-language inference well, but leaves out a great deal of interest. The ‘atoms’ of SL are sentence letters, which are each assigned either T or F by a model. Sentence letters can only represent declarative sentences of the English language, which means that SL is too coarse-grained to mimic the *parts* of a sentence. While SL can capture some forms of logical consequence in English, it neglects others. For example:

1. All women are mortal.
2. Ophelia is a woman.
- Therefore,
3. Ophelia is mortal.

The third sentence is a logical consequence of the first two, but SL isn’t capable of representing this as an entailment. Let sentence letter A stand for ‘All women are mortal,’ let B stand for ‘Ophelia is a woman,’ and let C stand for ‘Ophelia is mortal.’ The entailment claim $A, B \models C$ does not hold, because there is a model \mathbf{m} that assigns T to A and B , but assigns F to C . We need a formal language more expressive than SL.

Our new language needs symbols that represent named objects, including people like Ophelia. It also needs symbols that mimic the predicates of English. Predicates correspond roughly to what you get if you take an English sentence and remove a name, leaving a blank, e.g.:

4. ‘John is tall’ \Rightarrow ‘_____ is tall’
5. ‘Ophelia is a woman’ \Rightarrow ‘_____ is a woman’
6. ‘Ophelia is mortal’ \Rightarrow ‘_____ is mortal’

If we want to apply a predicate without invoking a name, we’ll need to use a variable, which functions somewhat like a pronoun in English. And to account for the word

‘all’ in our new language, we will need to use a *quantifier*. (We’ll discuss quantifiers a bit later.) In this chapter we outline a formal language with these features, and thus which can capture the kind of logical consequence exhibited above.

Many logic texts call the following language PL, for *predicate language*. We use ‘QL’ for *quantifier language*, because the quantifiers are more important than the predicates. However, before turning to the full language of QL (in the next chapter), we first consider a simpler sublanguage which we’ll call ‘QL1’. We call it ‘QL1’ because the predicates will all be 1-place.¹

QL1 has all the basic symbols of SL, plus a few more.

Definition 3.1. The *basic symbols* of QL1 are:

- (1) Logical Connectives: those of SL, plus \forall and \exists
- (2) Punctuation Symbols: those of SL
- (3) Sentence Letters: those of SL
- (4) Individual Constants: $a, b, c, d, \dots, p, a_1, b_1, c_1, \dots, p_1, a_2, \dots$
- (5) Individual Variables: $u, v, w, x, y, z, u_1, v_1, \dots, z_1, u_2, \dots$
- (6) 1-Place Predicates: $A', B', \dots, T', A'_1, B'_1, \dots, T'_1, A'_2, B'_2, \dots$

The most prominent additions are the new logical connectives, the two quantifiers. The first, ‘ \forall ’, is called the *universal quantifier*. It corresponds roughly to the words ‘all’ or ‘every’ in English. The second, ‘ \exists ’, is called the *existential quantifier*. It corresponds roughly to ‘there exists’, ‘there is’, or ‘some’, as in ‘Some elephants live a long time.’²

The individual constants of QL correspond roughly to names in English. They are lowercase Roman letters that start at ‘a’ and stop at ‘p’, and then they start over with subscripted integers. So we have, for example, a_1, a_2, a_3 , and so on.

Next, the individual variables correspond roughly to pronouns in English. They are Roman lowercase letters that go from ‘u’ to ‘z’ and then start over with subscripted positive integers.

We also have 1-place predicates in QL1. The one-place predicates are capital Roman letters going from ‘A’ to ‘T’ and then starting over with subscripted integers. There are an infinite number of each of the individual constants, variables, and 1-place predicates. This is so that however complex a sentence or argument we want to analyze, we never run out of QL1 symbols.

¹In QL there will be many-place predicates. The basics of a less formal version of QL1 were developed by Aristotle over 2,000 years before Gottlob Frege and others developed the full language we’re calling QL. QL was a big step for mankind.

²Although Frege, Peirce, and Mitchell first introduced quantifiers, the notation used here comes from Russell, who Church (1956: 288) says modified Peano’s notation.

3.1.2 Formulas of QL1

Our ultimate interest is in *sentences* of QL1, but to get to the sentences we have to work with a larger set of strings called ‘formulas.’ Formulas are defined recursively, starting with the atomic formulas (that is, those given by the base clause of the recursive definition). But before giving the definition we need to expand MathEnglish and add another kind of metavariable for QL1 individual variables.³ We use lowercase Greek letters to stand for variables of QL1, usually but not necessarily always from the beginning of the Greek alphabet (e.g., ‘ α ’ and ‘ β ’). Although we also used lowercase Greek letters as variables for SL sentences and will continue to do so for QL1 sentences and formulas, confusion shouldn’t arise as we typically use ‘ ϕ ’, ‘ ψ ’, and ‘ θ ’ for SL and QL1 sentences and use ‘ α ’ and ‘ β ’ for QL1 variables.

Definition 3.2. The *formulas of QL1* are given by the following recursive definition:

Base Clauses:

- (1) A sentence letter (atomic sentence of SL) is a formula.
- (2) A 1-place predicate followed by one individual constant or variable is a formula.

Generating Clauses:

- (1) If ϕ is a formula, then so is $\sim\phi$.
- (2) If ϕ and θ are formulas, then so are $(\phi \rightarrow \theta)$ and $(\phi \leftrightarrow \theta)$.
- (3) If all of $\phi_1, \phi_2, \phi_3, \phi_4, \dots, \phi_n$ are formulas (the list must include at least two formulas and be finite), then so are $(\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4 \wedge \dots \wedge \phi_n)$ and $(\phi_1 \vee \phi_2 \vee \phi_3 \vee \phi_4 \vee \dots \vee \phi_n)$.⁴
- (4) If α is a QL1 variable and ϕ is a formula that does not contain an expression of the form $\forall\alpha$ or $\exists\alpha$, then $\forall\alpha\phi$ and $\exists\alpha\phi$ are formulas.

Closure Clause: A string of symbols is a formula only if it can be generated by the clauses above.

For example, $A'b$ is a formula, and so is $D'x_4$. Each of these formulas is atomic. Formulas of the form $\forall\alpha\phi$ are called *universal* formulas and formulas of the form $\exists\alpha\phi$ are called *existential*. Here are some additional examples of QL1 formulas.

³MathEnglish variables are symbols of the metalanguage while individual variables of QL1 are symbols of the object language.

⁴Remember that this generating clause is non-standard; most logic books treat conjunction and disjunction as binary (two-place) operations. Our definition is closer to English and avoids unnecessary parentheses.

- | | |
|------------------------|---|
| 1. $\forall xJ'x$ | 4. $\exists y\sim\forall xG'y$ |
| 2. $\sim\exists yK'x$ | 5. $\forall x\exists yH'z$ |
| 3. $\exists zL'_{12}b$ | 6. $(\forall x\forall zP'z_{176}\rightarrow\forall yG'y)$ |

Contrarily, $\forall x\forall xG'x$ is *not* a formula. That's because it's of the form $\forall x\phi$ where ϕ is a formula that contains the expression $\forall x$.⁵ Neither is $\forall aG'a$, because \forall *must* be paired with a variable, and 'a' is a constant.

Finally, we have unofficial formulas, just as we had unofficial sentences in SL (compare with def. 2.4, on page 17).

Definition 3.3. A string of symbols is an *unofficial* formula iff we can obtain it from an official formula by

- (1) deleting outer parentheses,
- (2) replacing official parentheses () with square brackets [] or curly brackets { },
or
- (3) omitting primes ' on a predicate letter.

As in SL, from an unofficial sentence we can unambiguously reconstruct the corresponding official sentence.

3.1.3 Other Properties of Formulas

As in section 2.1.4 for sentences of SL, we can define the concepts of subformula, order, main connective, and construction tree for formulas of QL1.

Definition 3.4. The *subformulas* of a formula are defined parallel to that of an SL sentence (see definition 2.5, on page 18) with the extra clauses that adding a quantifier adds one new subformula, the whole formula.

Importantly, the quantifier phrase is *not* a subformula. Thus, $\forall x\forall zG'x$ has three subformulas: $\forall x\forall zG'x$, $\forall zG'x$, and $G'x$. $\forall x$ is not a subformula, and neither is $\forall xG'x$.

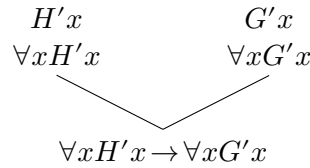
Definition 3.5. The *order* of a formula is defined parallel to that of an SL sentence (see def. 2.9, on page 19) with the extra clauses that adding a quantifier adds one to the order of the new formula.

⁵Continuing the practice started in section 2.1.3, we will not always put symbols, expressions, and sentences of QL1 that are mentioned (instead of used) in single quotes. For example, the tokens of the universal and existential quantifiers in definition 3.14 (on page 67) should, strictly speaking, be in quotes because they *mention* the symbols. Being stringent in the use of single quotes—and overly sensitive to the use/mention distinction in general—tends to cloud what are relatively clear and straightforward concepts. Accordingly, we will tend to favor conceptual clarity over notational rigor. Wherever it may be helpful, we will provide footnotes with more rigorous and detailed explanation.

Definition 3.6. The *main connective* for a formula is defined as before (def. 2.11, on page 19). It is the connective token (or tokens) that occur(s) in the formula but in no proper subformula.

Definition 3.7. The *construction tree* for a formula is defined as before (def. 2.13, on page 19) with the obvious extensions. As before, the order of a formula is the height of the construction tree's longest branch, measured by counting nodes. Each node in the tree (including the bottom node) is a subsentence, and the main connective is the connective added at the very bottom of the tree.

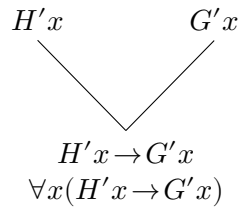
Example 3.8. Consider the (unofficial) formula $\forall x H'x \rightarrow \forall x G'x$. It's a conditional; i.e., its main connective is the arrow. The construction tree of the formula is:



We can read off from this construction tree that the order of the formula is 3. It has five subformulas:

- | | |
|---|-----------|
| (1) $\forall x H'x \rightarrow \forall x G'x$ | |
| (2) $\forall x H'x$ | (4) $H'x$ |
| (3) $\forall x G'x$ | (5) $G'x$ |

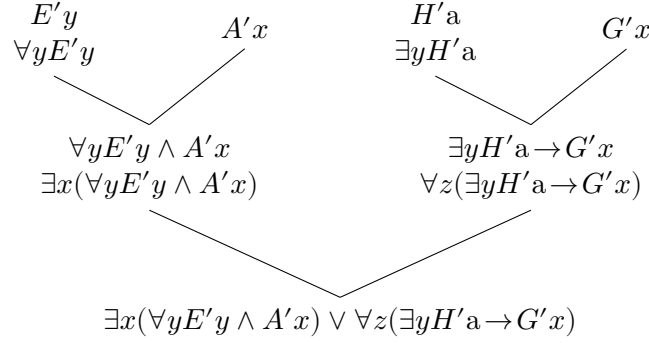
Example 3.9. Next consider the formula $\forall x (H'x \rightarrow G'x)$, which many people confuse with the conditional from example 3.8. Consider carefully the differences between the two. This is a universal formula; i.e., its main connective is the universal quantifier. The construction tree of the formula is:



And we can read off from this construction tree that the order of the formula is 3. It has four subformulas:

- | | |
|--------------------------------------|-----------|
| (1) $\forall x(H'x \rightarrow G'x)$ | (3) $H'x$ |
| (2) $(H'x \rightarrow G'x)$ | (4) $G'x$ |

Example 3.10. For a more complicated example consider $\exists x(\forall yE'y \wedge A'x) \vee \forall z(\exists yH'a \rightarrow G'x)$. This is a disjunction; i.e., its main connective is vee, \vee . The construction tree of the formula is:



We can read off from this construction tree that the order of the formula is 5. It has eleven subformulas:

- | | |
|---|---|
| (1) $\exists x(\forall yE'y \wedge A'x) \vee \forall z(\exists yH'a \rightarrow G'x)$ | |
| (2) $\exists x(\forall yE'y \wedge A'x)$ | (7) $\forall z(\exists yH'a \rightarrow G'x)$ |
| (3) $\forall yE'y \wedge A'x$ | (8) $\exists yH'a \rightarrow G'x$ |
| (4) $\forall yE'y$ | (9) $\exists yH'a$ |
| (5) $A'x$ | (10) $G'x$ |
| (6) $E'y$ | (11) $H'a$ |

3.1.4 Sentences of QL1

Now that we have defined which strings of basic symbols are formulas of QL1 we can define which are sentences. The definition of a sentence depends on a few supporting definitions. We'll give the supporting definitions afterwards.

Definition 3.11. A string of QL symbols is a *sentence* iff it is a formula that contains no free variables.

Definition 3.12. A variable (token) in a formula ϕ is *free* iff it is not bound.

Definition 3.13. In a formula $\exists\alpha\phi$ or $\forall\alpha\phi$, we say that ϕ is the *scope* of the quantifier $\exists\alpha$ or $\forall\alpha$.

This definition applies whether $\exists\alpha\phi$ or $\forall\alpha\phi$ is a stand-alone formula or is instead a proper subformula of some other formula.

Definition 3.14. A variable token α in a formula ϕ is *bound* iff either (i) it appears as part of a quantifier expression with that variable, $\exists\alpha$ or $\forall\alpha$; or (ii) it occurs within the scope of a quantifier expression with that variable, $\exists\alpha$ or $\forall\alpha$.

E.g., in the sentence $\exists x(G'x \wedge H'x)$, the first token of x is bound because it's part of the quantifier expression ' $\exists x$ '. The second and third tokens of x are bound because they are within the scope of a quantifier expression with that variable.

Alternatively, we can understand when a variable is bound by thinking of a tree of the sentence. A variable token is bound iff a quantifier with the same variable appears below or at the same level as (but still on the same branch as) that token. The quantifier that binds a variable is the *first* quantifier that appears below or at the same level as (but still on the same branch as) the variable.

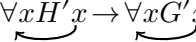
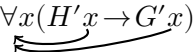
Definition 3.15. An *atomic sentence* of QL1 is an atomic formula of QL1 which is also a sentence; that is, it is an atomic formula of QL1 that has no free variable.

We have unofficial sentences too, just as we have unofficial formulas.

Definition 3.16. A string of symbols is an *unofficial* sentence iff it's an unofficial formula that contains no free variables. In other words, we can get an unofficial sentence from an official one by

- (1) deleting outer parentheses,
- (2) replacing official parentheses () with square brackets [] or curly brackets { },
or
- (3) omitting primes ' on the predicate letters.

Example 3.17. Both formulas $\forall xH'x \rightarrow \forall xG'x$ and $\forall x(H'x \rightarrow G'x)$ from examples 3.8 and 3.9 are sentences, because in both formulas all variables are bound. The following arrows point from each variable to the quantifier that binds it. (We'll ignore the variables in quantifier expressions; it's obvious which quantifiers they belong to.)

- (1) $\forall xH'x \rightarrow \forall xG'x$

- (2) $\forall x(H'x \rightarrow G'x)$


To see that these variables are bound by those quantifiers, look at the construction trees of the formulas from examples 3.8 and 3.9. You will see that in each case the indicated quantifier is the first that appears below, but still on the same branch as, the token variable.

Example 3.18. The formula $\exists x(\forall yE'x \wedge A'x) \vee \forall z(\exists yH'a \rightarrow G'x)$ is not a sentence. That's because it has a free variable. The free variable in the formula is underlined, while arrows point from each bound variable to the quantifier that binds it (again, ignoring variables in quantifier expressions).

$$(1) \quad \exists x(\forall y E' x \wedge A' x) \vee \forall z(\exists y H' a \rightarrow G' \underline{x})$$

The two quantifiers on the RHS of the disjunction are not binding any token variables, besides the ones that appear in the quantifier expressions themselves.

Example 3.19. In each of the following three formulas the free variable tokens are underlined, and arrows go from variable tokens to the quantifiers that bind them.

$$(1) \quad \forall x(B \wedge \exists z K' x) \rightarrow \exists x N \underline{x}$$

$$(2) \quad D \wedge (\exists u \forall w P' u \vee C' \underline{y})$$

$$(3) \quad \exists z((H' \underline{x} \wedge G) \leftrightarrow D z)$$

Because (1) has no free variables while (2) and (3) do, (1) is a sentence but (2) and (3) are not.

3.2 Models

As with SL, sentences of QL1 have no inherent semantics. But, also as with SL, we can give models for sentences of QL1, and these models allow us to investigate entailment for QL1. There are many ways to carry out the details of QL1 semantics (i.e., give models), but each leads to essentially the same results. They all end up with the same set of logical truths and the same entailment relation.⁶

Sentences of QL1 correspond roughly to the sentences of English, and we need a definition for ‘model’ such that each model gives each QL1 sentence a determinate truth value. Formulas that aren’t sentences correspond to grammatical sentences of English that don’t have a determinate truth value, because they contain pronouns. For example, consider the English sentence ‘He is the author of Waverley.’ The sentence may be either true or false, depending on who ‘he’ is. Often we implicitly invoke context in order to determine the referent of a pronoun. Someone reading a biography of the English author Sir Walter Scott may read the above sentence and evaluate it as true. However, in a conversation in which ‘he’ refers to Aristotle, we’ll evaluate the sentence as false. By itself, however, it can’t be evaluated without further specification. As with pronouns of English, the unbound variables of QL1 formulas aren’t going to have any context-independent ‘referent’ (i.e., assignment). We don’t want formulas of QL1 that aren’t sentences to get determinate truth values from the models of QL1.

⁶See Hodges 2001b: 49ff, 1997.

3.2.1 Models in QL1

All that is required for an SL model for ϕ is the assignment of a truth value to each sentence letter in ϕ . QL1 has predicate letters and constants, which will require a different kinds of assignments. Furthermore, QL1 has quantifiers. As we noted before, the quantifiers roughly correspond to English words such as *all* or *some*, so each model must specify a set of objects that the quantifiers are about. In more technical language, a QL1 model must fix a domain of objects over which we can quantify.

Definition 3.20. A *model* for ϕ , \mathbf{m} , consists of:

- (1) an assignment of a truth value T or F to each sentence letter in ϕ ;
- (2) a single, non-empty set U , called the *universe* or *domain*;
- (3) an assignment of a subset of U to each 1-place predicate in ϕ ;
- (4) an assignment of an object from U to each individual constant in ϕ .⁷

As with SL models, we use the following notational conventions: Given some sentence letter, like P , $\mathbf{m}(P)$ is the truth value \mathbf{m} assigns to P . Given an individual constant, like a , $\mathbf{m}(a)$ is the object from U assigned by \mathbf{m} to a . Given a 1-place predicate, like G' , $\mathbf{m}(G')$ is the set of objects from U that are assigned to G' by \mathbf{m} .⁸

Earlier we said that the individual constants are roughly similar to proper names in English. One difference is that, in QL1, each individual constant in ϕ corresponds to exactly one object in the domain. In English, on the other hand, some proper names—e.g., ‘John Smith’—correspond to more than one person, and some—e.g., ‘Mordecai Alonzo Frazzle III’—do not correspond to any person. While we require that each constant in ϕ is assigned an object from the domain, we do not require that different constants be assigned different objects.

We distinguish different models by affixing integers as subscripts to the symbol ‘ \mathbf{m} ’. So, for example, \mathbf{m}_1 , \mathbf{m}_2 , \mathbf{m}_3 , ..., \mathbf{m}_{316} , etc., are each different models.

As with SL, we have QL1 models for sets of sentences:

Definition 3.21. Given that Δ is a set of QL1 sentences, \mathbf{m} is a *model for* Δ iff \mathbf{m} is a model for each sentence in Δ .

There are also models that make assignments to all the sentence letters, constants, and 1-place predicates of QL1. Any such model is a model for every QL1 sentence. Let’s call these *models for QL1*:

⁷Recall the discussions of ‘set’, ‘subset’, and ‘element’ in section 1.4.2 in chapter 1.

⁸We pause here to make two points for those keeping careful score: (1) For those comfortable with the abstract notion of a function, we can think of models as functions from the set of basic symbols of QL1 (less the logical operators, variables, and parentheses) to the kinds of objects mentioned in definition 3.2 (on the previous page) (objects or subsets of U). Thought of in this way, this notation is just the normal notation for functions. (2) Those trying to keep careful track of the use/mention distinction should note that here we’ve been especially loose. Because it’s the symbols ‘ P ’, ‘ a ’, ‘ G' ’ (etc) themselves that the model maps to some object (a truth value, subset of U , etc), we really should put the argument of the model (construed as a function) in single quotes. E.g., we should write ‘ $\mathbf{m}('P')$ ’, not ‘ $\mathbf{m}(P)$ ’, when denoting the object from U assigned to the *symbol* ‘ P ’ by \mathbf{m} .

Definition 3.22. \mathbf{m} is a *model for QL1* iff \mathbf{m} is a model for every sentence of QL1.

As we define truth in QL1, we need to make sure every model for ϕ fixes a unique truth value for ϕ .⁹ But there is a price we must pay for QL1's superior models. QL1 is more complicated than SL, and will require some additional metalinguistic tools. Before defining truth for all QL1 sentences, we must define 'terms' and 'model variants'.

Definition 3.23. The *individual terms* of QL1 are the constants and variables of QL1. We will use ' q ', ' r ', ' s ', and ' t ' (along with subscripts) as MathEnglish variables for them.

This means that italic roman ' q ', ' r ', ' s ', ' t ', and the Greek ' α ', ' β ', etc., can be MathEnglish variables for QL variables. However, while ' α ', ' β ', etc. stand *only* for variables, the italic roman letters can also range over QL1 constants. The use of metavariables for individual terms in the object language simplifies our notation considerably.

At this point we could define truth for sentences without quantifiers, though for now we offer only a short sketch. The sentence Ga is true on \mathbf{m} iff the object \mathbf{m} assigns to ' a ' is an element of the set \mathbf{m} assigns to ' G '; i.e., iff $\mathbf{m}(a) \in \mathbf{m}(G)$. If the object \mathbf{m} assigns to ' a ' isn't a member of the set assigned to ' G ', Ga is false on \mathbf{m} .

Quantifiers require more complexity. A sentence like $\forall xEx$ is true iff every object in the domain, U , is an element of $\mathbf{m}(E)$. So if \mathbf{m} assigns U the set of even integers and $\mathbf{m}(E) = U$, $\forall xEx$ is true. But if instead $\mathbf{m}(E) = \{2, 4, 6\}$, then there are objects in the domain (e.g. 8) not in $\mathbf{m}(E)$, and so $\forall xEx$ is false. For simple quantified sentences this quick definition is good enough; but it won't work for more complex sentences, such as $\forall x\exists y(Hy \rightarrow Gx)$.

To define truth for quantified sentences more precisely, we first need a convenient way to refer to models that make identical assignments everywhere except at one constant.

Definition 3.24. A *t-variant* of \mathbf{m} is any model \mathbf{m}^* that makes any assignment to t but otherwise makes exactly the same assignments as \mathbf{m} .¹⁰

Put another way, a *t-variant* of \mathbf{m} makes all the same assignments as \mathbf{m} except possibly at constant t . But what if \mathbf{m} doesn't assign anything to t ? In that case, the *t*-variants are all the models that are identical with \mathbf{m} but that also assign something to t .

We will generally denote *t*-variants of a model \mathbf{m} by affixing t as a superscript to ' \mathbf{m} '. For example, \mathbf{m}^c is a *c*-variant of \mathbf{m} . On this example \mathbf{m}^c and \mathbf{m} make all the same assignments, except possibly to the constant c . We extend this notation when

⁹We define truth so that it applies only to *sentences*, not *formulas*. Almost all other textbooks define truth (or a related concept) for formulas, and then in turn use that to define truth of a sentence. We find this an unnecessarily circuitous way of defining truth in QL1. Following [Mates 1972](#), we instead define truth of a sentence directly, bypassing truth for formulas altogether.

¹⁰One result of this definition is that, for every term t , any model \mathbf{m} that assigns something to t is a *t*-variant of itself.

the symbol denoting the original model is itself complex. So, given an c-variant of model \mathbf{m} , i.e., \mathbf{m}^c , \mathbf{m}^{cd} is a d-variant of \mathbf{m}^c . For another example, if \mathbf{m}_4^e is an e-variant of \mathbf{m}_4 , then \mathbf{m}_4^e and \mathbf{m}_4 make identical assignments to everything except maybe e.

We need one more piece of notation before we can get to the definition of truth.

Definition 3.25. If ϕ is a QL1 formula and t and s are terms, then $\phi s/t$ is the formula you get by replacing each unbound token of t in ϕ with a token of s .

Example 3.26.

- (1) If ϕ is A , then $\phi y/x$ is A .
- (2) If ϕ is Bx , then $\phi y/x$ is By .
- (3) If ϕ is By , then $\phi y/x$ is By .
- (4) If ϕ is Bx , then $\phi y/w$ is Bx .
- (5) If ϕ is $\forall x Bx$, then $\phi y/x$ is $\forall x Bx$.
- (6) If ϕ is $Cx \wedge \forall x Bx$, then $\phi y/x$ is $Cy \wedge \forall x Bx$.
- (7) If ϕ is $\exists y(Cx \wedge \forall x Bx)$, then $\phi y/x$ is $\exists y(Cy \wedge \forall x Bx)$.
- (8) If ϕ is $\exists y(Cx \wedge \forall x Bx)$, then $\phi a/x$ is $\exists y(Ca \wedge \forall x Bx)$.
- (9) If ϕ is $\exists y(Cx \wedge Bx)$, then $\phi a/x$ is $\exists y(Ca \wedge Ba)$.

3.2.2 Truth in a Model

We are finally ready to define truth in a model for QL1.

Definition 3.27. The following clauses fix when a QL1 sentence θ is *true* (or *false*) on a model for θ , \mathbf{m} :

- (1) A sentence letter ϕ is true on \mathbf{m} iff \mathbf{m} assigns true to it, i.e. iff $\mathbf{m}(\phi) = \top$.
- (2) An atomic sentence Pt with a 1-place predicate P and an individual term t is true on \mathbf{m} iff what \mathbf{m} assigns to the individual term t is in the set \mathbf{m} assigns to the predicate, i.e. iff $\mathbf{m}(t) \in \mathbf{m}(P)$.
- (3) A negation $\sim\phi$ is true on \mathbf{m} iff the unnegated formula ϕ is false on \mathbf{m} .
- (4) A conjunction $(\phi_1 \wedge \dots \wedge \phi_n)$ is true on \mathbf{m} iff all conjuncts ϕ_1, \dots, ϕ_n are true on \mathbf{m} .
- (5) A disjunction $(\phi_1 \vee \dots \vee \phi_n)$ is true on \mathbf{m} iff at least one disjunct ϕ_1, \dots, ϕ_n is true on \mathbf{m} .
- (6) A conditional $(\psi \rightarrow \phi)$ is true on \mathbf{m} iff the LHS ψ is false or the RHS ϕ is true on \mathbf{m} , or both.
- (7) A biconditional $(\psi \leftrightarrow \phi)$ is true on \mathbf{m} iff both sides, ψ and ϕ , have the same truth value on \mathbf{m} .

- (8) A universal quantification $\forall\alpha\phi$ is true on \mathbf{m} iff $\phi t/\alpha$ is true on *all* t -variants of \mathbf{m} (where t is the first constant not in ϕ).
- (9) An existential quantification $\exists\alpha\phi$ is true on \mathbf{m} iff $\phi t/\alpha$ is true on *some* t -variant of \mathbf{m} (where t is the first constant not in ϕ).
- (10) A sentence ϕ is false on \mathbf{m} iff ϕ is not true on \mathbf{m} .

The following table has two example models, *Pos Int* and *States*:

Symbol		Model	
		Pos Int	States
Universe:		The set of positive integers	The set of current US states
Sent. Let.:	A	true	false
	B	true	false
	C	false	true
	D	true	false
	E	true	false
	G	false	true
Constants:	a	1	Louisiana
	b	9	Maine
	c	72	Georgia
	d	3	Nebraska
	e	1	New Mexico
	f	2	Texas
1-place:	A'	all pos int	Midwestern
	B'	empty set	name with > 5 letters
	C'	even	Coastal
	D'	odd	on the Pacific coast
	E'	prime	{Ohio}
	G'	multiple of 7	{Ohio,Alabama}

Table 3.1: Example Models

Example 3.28. The sentence $\sim Gb$ is true on the model *Pos Int* given in table 3.1 (on the current page).

Proof: The model *Pos Int* has as its domain (or universe) the set of positive integers. *Pos Int* assigns 9 to b , and the set of multiples of 7 (i.e., $\{7, 14, 21, 28, \dots\}$) to G .

The number 9 is not a multiple of 7, so $\text{Pos Int}(b) \notin \text{Pos Int}(G)$.¹¹ To see this, we

¹¹As a reminder, $\text{Pos Int}(b) \notin \text{Pos Int}(G)$ asserts that what *Pos Int* assigns to b is not an element

need only to look at the set assigned to G and see if 7 is a member; it's not. So, Gb is false on $Pos Int$. It follows that $Pos Int$ makes $\sim Gb$ true. ■

Example 3.29. The sentence $(Gd \rightarrow De)$ is true on the model $Pos Int$ given in table 3.1 (on the previous page).

Proof: $(Gd \rightarrow De)$ is true on $Pos Int$ iff the LHS is false or the RHS is true. The LHS, Gd , is true on $Pos Int$ iff what the model assigns to d is a member of the set assigned to G . $Pos Int$ assigns the number 3 to d , but 3 isn't a multiple of 7. So, $Pos Int(d) \notin Pos Int(G)$, and Gd is false on $Pos Int$. It follows that $(Gd \rightarrow De)$ is true on $Pos Int$. ■

Example 3.30. The sentence $Aa \vee Gc$ is false on the model $States$ given in table 3.1 (on the previous page).

Proof: The model $States$ assigns Louisiana to a , Georgia to c , the set of Midwestern states to A , and the set $\{\text{Ohio, Alabama}\}$ to G .

Aa is true on $States$ iff what the model assigns to a is a member of the set assigned to A , i.e., $States(a) \in States(A)$. But Louisiana isn't in the set of Midwestern states; while it might be unclear exactly which states are genuinely Midwestern, Louisiana definitely doesn't belong. So Aa is false on $States$.

Gc is true on $States$ iff what the model assigns to c is in the set assigned to G ; i.e., iff $States(c) \in States(G)$. But $States$ assigns $\{\text{Ohio, Alabama}\}$ to G . $States$ assigns Georgia to c , and Georgia isn't in that set. Thus, $States$ makes Gc false.

Both disjuncts are false on $States$, so $States$ makes $Aa \vee Gc$ false. ■

Often, whether a sentence is true depends on the model selected. Because the above models involve things such as numbers and states, whether the given sentence is true depends on facts about numbers and states. When we're trying to compute the truth value of a sentence in a given model, the truth value depends not only on our definitions, but also on facts about the objects in the universe of the model.

Example 3.31. (i) The sentence $\forall x Dx \rightarrow \forall x Gx$ is true on the model $Pos Int$, given in table 3.1 (on the previous page), while (ii) the sentence $\forall x (Dx \rightarrow Gx)$ is false on that same model.

Proof: These two sentences look deceptively similar, but they have different truth values on the model $Pos Int$.

(i) $\forall x Dx \rightarrow \forall x Gx$ is true on $Pos Int$ iff $Pos Int$ either makes the LHS false or the RHS true.

$\forall x Dx$ is true on $Pos Int$ iff for the first constant not in the sentence, t , every t -variant of $Pos Int$ makes Dt true.¹² We *must* use 'a' as our constant; there are no

of the set that $Pos Int$ assigns to G .

¹²Remember that t isn't itself a constant—it's a metavariable that stands for a constant in the object language, QL1.

constants in Dx , so we pick the first one available. Thus, $\forall xDx$ is true on $Pos Int$ iff every object in the domain of $Pos Int$ that could be assigned to a makes Da come out true.

$Pos Int$ assigns the set of odd numbers to D . But what happens when we consider the a -variant of $Pos Int$ such that $Pos Int^a(a) = 2$? The number 2 isn't in the set of odd numbers; i.e., $Pos Int^a(a) \notin Pos Int^a(D)$. So, there is a $Pos Int^a$ such that Da is false. That means, in turn, that $Pos Int$ makes $\forall xDx$ false. Thus, $\forall xDx \rightarrow \forall xGx$ is true on $Pos Int$.

(ii) $\forall x(Dx \rightarrow Gx)$ is true on $Pos Int$ iff every a -variant of $Pos Int$ makes $Da \rightarrow Ga$ true.

Is there any a -variant of $Pos Int$ that makes Da true and Ga false? Yes. $Pos Int$ assigns the set of odd numbers to D and the set of multiples of 7 to G . Let $Pos Int^a(a) = 3$. The number 3 is odd, but it isn't a multiple of 7. So, $Pos Int^a(a) \in Pos Int^a(D)$ and $Pos Int^a(a) \notin Pos Int^a(G)$. Thus, $Pos Int^a$ makes Da true and Ga false, which in turn makes $Da \rightarrow Ga$ false. Because there is a a -variant of $Pos Int$ that makes $Da \rightarrow Ga$ false, $\forall x(Dx \rightarrow Gx)$ is false on $Pos Int$. ■

Example 3.32. On the model $Pos Int$, given in table 3.1 (on page 72), (i) the sentence $\exists x(Cx \wedge Dx)$ is false but (ii) $(\exists xCx \wedge \exists xDx)$ is true.

Proof: As with the last example, these two sentences are deceptively similar. They have different truth on $Pos Int$, however.

(i) First let's consider the sentence $\exists x(Cx \wedge Dx)$. $\exists x(Cx \wedge Dx)$ is true on $Pos Int$ iff there is an a -variant of $Pos Int$ that makes $Ca \wedge Da$ true.

Is there an a -variant of $Pos Int$ such that $Ca \wedge Da$ is true? No. The only way an a -variant can make $Ca \wedge Da$ true is if it makes both conjuncts true. But $Pos Int$ assigns the even numbers to C and the odd numbers to D . There is no number that is both even and odd! It follows that every variant must make either Ca or Da false.

$\exists x(Cx \wedge Dx)$ is false on $Pos Int$.

(ii) $(\exists xCx \wedge \exists xDx)$ is true on $Pos Int$ iff both conjuncts are true on $Pos Int$.

$\exists xCx$ is true on $Pos Int$ iff there is an a -variant of $Pos Int$ that makes Ca true. Is there any such a -variant? Yes; consider the variant $Pos Int^a$ that assigns 4 to a . $Pos Int^a$ assigns the even numbers to C , just like $Pos Int$. The number 4 is even, so $Pos Int^a(a) \in Pos Int^a(C)$. Thus, $Pos Int^a$ makes Ca true, and so $Pos Int$ makes $\exists xCx$ true.

Similar reasoning holds for the other conjunct. $\exists xDx$ is true on $Pos Int$ iff there is an a -variant of $Pos Int$ that makes Da true. $Pos Int^a$ assigns the odd numbers to D . The number 3 is odd, so when $Pos Int^a(a) = 3$, $Pos Int^a(a) \in Pos Int^a(D)$. Thus, $Pos Int^a$ makes Da true. It follows that $Pos Int$ makes $\exists xDx$ true.

$Pos Int$ makes $(\exists xCx \wedge \exists xDx)$ true. ■

3.2.3 Minimal Models in QL1

We may calculate the truth value of a QL1 sentence ϕ on some \mathbf{m} iff \mathbf{m} is a model for ϕ . If \mathbf{m} isn't a model for ϕ , then ϕ has no truth value on it. Be careful! Even if \mathbf{m}_1 is a model for some sentence ϕ and \mathbf{m}_2 is a model for some sentence ψ , it doesn't follow that either \mathbf{m}_1 or \mathbf{m}_2 is a model for, say, $\phi \rightarrow \psi$.

We were able to calculate the truth values of the previous example problems using the two models in table 3.1 (on page 72) because none of the sentences contained sentence letters, constants, or predicates that weren't given an assignment in the table. Those assignments not mentioned, in effect, don't matter for truth evaluation. The following theorem proves that the unmentioned assignments won't make a difference.

As with SL, we have *minimal models* in QL1:

Definition 3.33. Model \mathbf{m} is a *minimal model* for ϕ iff \mathbf{m} makes the minimum assignments necessary for \mathbf{m} to be a model for ϕ ; i.e., makes assignments to the universe U , to each sentence letter, constant, and 1-place predicate in ϕ , but to nothing else.

For an example minimal model, consider the sentence $(\exists x Cx \wedge \exists x Dx)$. This sentence has two 1-place predicates, C and D , and no sentence letters or constants. A minimal model \mathbf{m} will not make assignments to any sentence letter or constant. \mathbf{m} will assign subsets of U to C and D , but won't make an assignment to any other 1-place predicate.

When calculating the value of a sentence ϕ on a model, we only need to worry about the assignments to the symbols in ϕ and the universe. We can ignore the other assignments. The following theorem demonstrates this. (This is the QL1 version of theorem 2.68 in chapter 2.)

Theorem 3.34. Let ϕ be any QL1 sentence. If there are two models for ϕ , \mathbf{m}_1 and \mathbf{m}_2 , that have the same domain, U , and make the same assignments for all the sentence letters, individual constants, and 1-place predicates contained in ϕ , then ϕ is true on \mathbf{m}_1 iff ϕ is true on \mathbf{m}_2 .

Proof:

Base Step: Let θ be a sentence of order 1. θ must be either (i) a sentence letter, or (ii), a 1-place predicate followed by a constant.

(i) If θ is a sentence letter, and, as we assumed, \mathbf{m}_1 and \mathbf{m}_2 make the same assignments for all the sentence letters, then θ is true on \mathbf{m}_1 iff θ is true on \mathbf{m}_2 .

(ii) If θ is a 1-place predicate, P , followed by a constant, t , and, as we assumed, \mathbf{m}_1 and \mathbf{m}_2 make the same assignments for all the constants and 1-place predicates, then $\mathbf{m}_1(P) = \mathbf{m}_2(P)$ and $\mathbf{m}_1(t) = \mathbf{m}_2(t)$. It follows that $\mathbf{m}_1(t) \in \mathbf{m}_1(P)$ iff $\mathbf{m}_2(t) \in \mathbf{m}_2(P)$. Thus, θ is true on \mathbf{m}_1 iff θ is true on \mathbf{m}_2 .

Either way, θ is true on \mathbf{m}_1 iff θ is true on \mathbf{m}_2 .

Inheritance Step: Recursive Assumption: Assume that for each QL1 sentence θ of order n , θ is true on \mathbf{m}_1 iff θ is true on \mathbf{m}_2 . Let ψ be of order $n + 1$.

Negation: The reasoning here is exactly the same reasoning in the corresponding clause of theorem 2.68 in chapter 2.

Conditional, Biconditional, Disjunction, Conjunction: Same as above.

Universal Quantification: Say that the sentence ψ is of the form $\forall\alpha\theta$. Because $\forall\alpha\theta$ is a sentence, it has no unbound variables. So, dropping the quantifier, θ has a lone free variable: α .

$\forall\alpha\theta$ is true on \mathbf{m}_1 iff $\theta t/\alpha$ is true on every t -variant of \mathbf{m}_1 , where t is the first constant not in θ (definition of truth, \forall). The same holds for \mathbf{m}_2 .

Recall that \mathbf{m}_1 and \mathbf{m}_2 have the same domain, U . It follows that for each t -variant of \mathbf{m}_1 , there is a corresponding t -variant of \mathbf{m}_2 that assigns the same object from U to t ; i.e., for each \mathbf{m}_1^t there is a \mathbf{m}_2^t such that $\mathbf{m}_1^t(t) = \mathbf{m}_2^t(t)$. By Recursive Assumption (RA), $\forall\alpha\theta$ is of order $n + 1$, so $\theta t/\alpha$ is of order n . Also by RA, for each t -variant of \mathbf{m}_1 and its corresponding t -variant of \mathbf{m}_2 , $\theta t/\alpha$ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Either (i) $\theta t/\alpha$ is true on every t -variant of \mathbf{m}_1 or (ii) it isn't.

(i) Assume $\theta t/\alpha$ is true on every t -variant of \mathbf{m}_1 . It follows that for each t -variant of \mathbf{m}_1 that makes $\theta t/\alpha$ true, the corresponding t -variant of \mathbf{m}_2 also makes $\theta t/\alpha$ true. Thus, $\theta t/\alpha$ is true on every t -variant of \mathbf{m}_2 . Therefore $\forall\alpha\theta$ is true on both \mathbf{m}_1 and \mathbf{m}_2 .

(ii) Assume $\theta t/\alpha$ isn't true on every t -variant of \mathbf{m}_1 . There must be a t -variant of \mathbf{m}_1 that makes $\theta t/\alpha$ false. Call it \mathbf{m}_1' . So, there is some element in U such that when it is assigned to t by \mathbf{m}_1' , $\theta t/\alpha$ comes out false. It follows that there is a corresponding t -variant of \mathbf{m}_2 that makes the same assignment to t , call it \mathbf{m}_2' . Therefore, $\theta t/\alpha$ is false on \mathbf{m}_2' , and so $\forall\alpha\theta$ is false on both \mathbf{m}_1 and \mathbf{m}_2 .

From (i) and (ii), we may conclude that $\forall\alpha\theta$ is true on \mathbf{m}_1 iff $\forall\alpha\theta$ is true on \mathbf{m}_2 .

Existential Quantification: Say that ψ is of the form $\exists\alpha\theta$. As in the last clause, $\exists\alpha\theta$ is a sentence, so it has no unbound variables. θ has one free variable, α .

$\exists\alpha\theta$ is true on \mathbf{m}_1 iff $\theta t/\alpha$ is true on at least one t -variant of \mathbf{m}_1 , where t is the first constant not in θ (definition of truth, \exists). The same holds for \mathbf{m}_2 .

As before, \mathbf{m}_1 and \mathbf{m}_2 have the same domain. For each t -variant of \mathbf{m}_1 , there is a corresponding t -variant of \mathbf{m}_2 that assigns the same object from U to t . That is, for each \mathbf{m}_1^t there is a \mathbf{m}_2^t such that $\mathbf{m}_1^t(t) = \mathbf{m}_2^t(t)$. By Recursive Assumption (RA), $\exists\alpha\theta$ is of order $n + 1$, so $\theta t/\alpha$ is of order n . Also by RA, for each t -variant of \mathbf{m}_1 and its corresponding t -variant of \mathbf{m}_2 , $\theta t/\alpha$ is true on \mathbf{m}_1 iff it's true on \mathbf{m}_2 .

Either (i) $\theta t/\alpha$ is true on some t -variant of \mathbf{m}_1 or (ii) it isn't.

(i) Assume $\theta t/\alpha$ is true on some t -variant of \mathbf{m}_1 . Call this t -variant \mathbf{m}_1' . There is some element in U such that when it is assigned to t by \mathbf{m}_1' , $\theta t/\alpha$ comes out true. There is a corresponding t -variant of \mathbf{m}_2 that makes the

same assignment to t ; let's call it \mathbf{m}'_2 . So, $\theta t/\alpha$ is true on \mathbf{m}'_2 . Therefore $\exists\alpha\theta$ is true on both \mathbf{m}_1 and \mathbf{m}_2 .

(ii) Assume $\theta t/\alpha$ is false on *every* t -variant of \mathbf{m}_1 . For each t -variant of \mathbf{m}_1 , there is a corresponding t -variant of \mathbf{m}_2 that makes the same assignment from the domain to t . Thus, for each t -variant of \mathbf{m}_1 that makes $\theta t/\alpha$ false, the corresponding t -variant of \mathbf{m}_2 also makes $\theta t/\alpha$ false. Therefore, $\theta t/\alpha$ is false on every t -variant of \mathbf{m}_2 , and $\exists\alpha\theta$ is false on both \mathbf{m}_1 and \mathbf{m}_2 .

By (i) and (ii), $\exists\alpha\theta$ is true on \mathbf{m}_1 iff $\exists\alpha\theta$ is true on \mathbf{m}_2 .

Closure Step: There is no other way to form a QL1 sentence ϕ , so the above clauses are sufficient to show that if \mathbf{m}_1 and \mathbf{m}_2 have the same domain and make the same assignments, then ϕ is true on \mathbf{m}_1 iff ϕ is true on \mathbf{m}_2 . ■

3.2.4 Logical Truth: QT, QF, & QC

Just as we have a rigorous notion of *logical truth*, *falsity*, and *contingency* for SL (see section 2.2.3), we have analogous notions for QL1.

Definition 3.35. A sentence ϕ of QL is *quantificationally true* (QT) iff it is true on every model for ϕ .

Definition 3.36. A sentence ϕ of QL is *quantificationally false* (QF) iff it is false on every model for ϕ .

Definition 3.37. A sentence ϕ of QL is *quantificationally contingent* (QC) iff there's at least one model \mathbf{m}_1 on which it's true and at least one model \mathbf{m}_2 on which it's false.

Example 3.38. (i) The sentence $\forall x(Bx \vee \sim Bx)$ is QT, but (ii) $\forall x(Bx \wedge \sim Bx)$ is QF.

Proof: (i) $\forall x(Bx \vee \sim Bx)$ is true on \mathbf{m} iff every a -variant of \mathbf{m} , \mathbf{m}^a , makes $Ba \vee \sim Ba$ true.

One of two things must be true for each \mathbf{m}^a . Either what \mathbf{m}^a assigns to a is in the set it assigns to B or it isn't. I.e., either $\mathbf{m}^a(a) \in \mathbf{m}^a(B)$ or $\mathbf{m}^a(a) \notin \mathbf{m}^a(B)$. If it is, then Ba is true on \mathbf{m}^a , and so is $Ba \vee \sim Ba$. If it isn't, then $\sim Ba$ is true on \mathbf{m}^a , and so is $Ba \vee \sim Ba$. So, $Ba \vee \sim Ba$ is true on every \mathbf{m}^a .

Therefore, $\forall x(Bx \vee \sim Bx)$ is true on \mathbf{m} . Because we assumed nothing particular about \mathbf{m} , it makes no difference for our argument what assignments it makes. Thus what we concluded of \mathbf{m} —that $\forall x(Bx \vee \sim Bx)$ is true—holds for all models. Thus, $\forall x(Bx \vee \sim Bx)$ is QT.

(ii) $\forall x(Bx \wedge \sim Bx)$ is true on \mathbf{m} iff every a -variant of \mathbf{m} , \mathbf{m}^a , makes $Ba \wedge \sim Ba$ true.

As before, either $\mathbf{m}^a(a) \in \mathbf{m}^a(B)$ or $\mathbf{m}^a(a) \notin \mathbf{m}^a(B)$. If the former, then $\sim Ba$ is false on \mathbf{m}^a , and so is $Ba \wedge \sim Ba$. If the latter, then Ba is false on \mathbf{m}^a , and so is $Ba \wedge \sim Ba$. So, $Ba \wedge \sim Ba$ is false on every \mathbf{m}^a .

Thus, $\forall x(Bx \wedge \sim Bx)$ is false on \mathbf{m} . We assumed nothing particular about \mathbf{m} , so its assignments don't matter for our argument. What we concluded of \mathbf{m} —that $\forall x(Bx \wedge \sim Bx)$ is false—holds for all models. Therefore, $\forall x(Bx \wedge \sim Bx)$ is QF. ■

In the last example, we gave sustained arguments to prove that sentences were either QT or QF. To show that a QL1 sentence is QC, we use a different strategy. A sentence is QC iff it's true on one model and false on another (definition of QC). That means all we have to do is provide two appropriate models to demonstrate that a sentence is QC.

For example, we determined earlier that $\forall xDx \rightarrow \forall xGx$ is true on the model *Pos Int*. If we can find a model in which it is false, we will have shown it is QC. We can modify *Pos Int* to make a new falsifying model, *Pos Int**. Let *Pos Int** assign the entire domain to *D* so that it makes $\forall xDx$ true. We can keep the assignment *Pos Int* makes to *G*, the multiples of 7. So, on *Pos Int**, $\forall xGx$ is false. Thus, $\forall xDx \rightarrow \forall xGx$ is false on *Pos Int**.

The definition of model only requires that predicates be assigned a subset of the domain; the assigned subset can be the entire domain, as in this example, or the empty set. You will find that using the empty set or the entire domain as assignments is often helpful in constructing models for a desired outcome.

Example 3.39. The sentence $\forall xBx \vee \forall x\sim Bx$ is QC.

Proof: Let \mathbf{m}_1 be a model with a single element universe $U = \{1\}$ such that $\mathbf{m}_1(B) = \{1\}$. This makes $\forall xBx$ true, which in turn makes $\forall xBx \vee \forall x\sim Bx$ true in \mathbf{m}_1 . Everything in the domain—keeping in mind that ‘everything’ is just the object ‘1’—is also in the set assigned to *B*. So, when we look at the part of the sentence $\forall xBx$ governed by the quantifier and replace the variable with a constant (i.e., *Ba*), then no matter what the model assigns to the constant, the result is true.

Let \mathbf{m}_2 be a model with a two element universe $U = \{1, 2\}$ such that $\mathbf{m}_2(B) = \{1\}$. Then $\forall xBx \vee \forall x\sim Bx$ is false in \mathbf{m}_2 , because both disjuncts are false. $\forall xBx$ is false on \mathbf{m}_2 because not everything in the domain is in the set assigned to *B*; and $\forall x\sim Bx$ is false on \mathbf{m}_2 because the model doesn't make *B* the empty set.

Thus, the sentence $\forall xBx \vee \forall x\sim Bx$ is QC. ■

Example 3.40. The sentence $\forall xDx \rightarrow \sim \exists x\sim Dx$ is QT.

Proof: Assume that some model \mathbf{m} makes $\forall xDx \rightarrow \sim \exists x\sim Dx$ false. $\forall xDx \rightarrow \sim \exists x\sim Dx$ is false on \mathbf{m} iff the LHS is true and the RHS is false. Thus, $\forall xDx$ is true on \mathbf{m} and $\sim \exists x\sim Dx$ is false on \mathbf{m} .

From the truth of $\forall xDx$, it follows that every a-variant of \mathbf{m} makes *Da* true.

Because \mathbf{m} makes $\sim \exists x\sim Dx$ false, it follows that $\exists x\sim Dx$ is true. Given that \mathbf{m} makes $\exists x\sim Dx$ true, there must be some a-variant, \mathbf{m}^a , that makes $\sim Da$ true. On \mathbf{m}^a , *Da* must be false.

But we already concluded that every a-variant of \mathbf{m} makes *Da* true! It's contradictory for \mathbf{m}^a to make *Da* both true and false. Our original assumption, that some

model makes $\forall xDx \rightarrow \sim\exists x\sim Dx$ false, must be wrong. Therefore, $\forall xDx \rightarrow \sim\exists x\sim Dx$ is true on all models, and is thus QT. ■

3.3 Entailment and other Relations

We defined the following notions in SL: entailment, equivalence, contradictory, contrary, subcontrary, and logical independence. Now we extend these notions to QL1 sentences. While the definitions for SL refer to models of SL sentences, the corresponding definitions for QL1 refer to models of QL1 sentences. To remind ourselves of this difference, we will talk of *truth functional* entailment, *truth functional* equivalence, etc., for SL sentences, but talk of *quantificational* entailment, *quantificational* equivalence, etc., for QL1. Nevertheless, we don't want to overstate the differences: the concepts underlying the definitions are exactly the same whether we're working in SL or QL1.

Definition 3.41. A set Δ of QL1 sentences, possibly empty or infinite, quantificationally entails another QL1 sentence θ iff every model for Δ and θ either makes at least one sentence in Δ false or makes θ true; i.e. iff every model for Δ and θ which makes all sentences in Δ true also makes θ true.

As we did with SL (section 2.3.1), we'll give two narrower consequences of this definition.

- (1) A finite set of QL1 sentences ϕ_1, \dots, ϕ_n quantificationally entails another QL1 sentence θ iff every model for ϕ_1, \dots, ϕ_n , and θ either makes at least one of ϕ_1, \dots, ϕ_n false or makes θ true; i.e. iff every model for ϕ_1, \dots, ϕ_n , and θ that makes all of ϕ_1, \dots, ϕ_n true also makes θ true.
- (2) A sentence ϕ of QL *quantificationally entails* another sentence θ of QL1 iff every model for ϕ and θ either makes ϕ false or makes θ true; i.e. iff every model for ϕ and θ that makes ϕ true also makes θ true.

Also as before we'll use the double turnstile to represent the entailment relation. Thus if ϕ quantificationally entails θ , we'll write ' $\phi \models \theta$ '. If the finite set of sentences ϕ_1, \dots, ϕ_n quantificationally entails θ , we'll write ' $\phi_1, \dots, \phi_n \models \theta$ '. And, if a set Δ of QL sentences quantificationally entails θ we'll write ' $\Delta \models \theta$ '.

Example 3.42. Show whether the following holds: $\forall xGx \models Ga$.

Proof: Assume a model for $\forall xGx$ and Ga , \mathbf{m} , such that $\forall xGx$ is true. By the definition of truth for \forall , it follows that Ga is true on all a-variants of \mathbf{m} . The model \mathbf{m} is an a-variant of itself,¹³ so Ga is true on \mathbf{m} .

Our only assumption is that the model \mathbf{m} makes the LHS of the double turnstile true. It follows that Ga is true, so the entailment holds. ■

¹³For any term t and for any model \mathbf{m} , \mathbf{m} is a t -variant of itself!

Example 3.43. $\forall xGx \models Gb$

Proof: This entailment is slightly harder to prove than the last. It's a quirk of our definition of truth that makes the last so easy to establish. By changing the constant in the sentence on the RHS from 'a' to 'b', we add a few steps to our proof.

Assume a model \mathbf{m} such that $\forall xGx$ is true. So Ga is true on all a-variants of \mathbf{m} .

There is some a-variant, \mathbf{m}^a , that assigns to a the exact same object from U that \mathbf{m} assigns to b . (All a-variants of \mathbf{m} have the same domain as \mathbf{m} itself.) So, we know that $\mathbf{m}^a(a) \in \mathbf{m}^a(G)$. \mathbf{m}^a and \mathbf{m} assign the same set to G , so $\mathbf{m}^a(a) \in \mathbf{m}(G)$. And because $\mathbf{m}^a(a)$ is the same object as $\mathbf{m}(b)$, it follows that $\mathbf{m}(b) \in \mathbf{m}(G)$. And so Gb is true on \mathbf{m} . The entailment holds. ■

Example 3.44. $\forall x(Cx \rightarrow Dx), Co \models Do$

Proof: The entailment holds. Assume some model \mathbf{m} such that $\forall x(Cx \rightarrow Dx)$ and Co are true. By the definition of truth for \forall , it follows that $(Ca \rightarrow Da)$ is true on all a-variants of \mathbf{m} . Let's take the object that \mathbf{m} assigns to 'o' and name it 'Ophelia'. Now take the a-variant that assigns Ophelia to 'a' and call it \mathbf{m}^a . (We know there is such an assignment because \mathbf{m} and \mathbf{m}^a have the same universe.) Thus, $\mathbf{m}(o) = \mathbf{m}^a(a)$. And \mathbf{m}^a is an a-variant of \mathbf{m} , so they make all the same assignments to the predicate letters.

The model \mathbf{m} makes Co true, therefore $\mathbf{m}(o) \in \mathbf{m}(C)$. Because $\mathbf{m}^a(a) = \mathbf{m}(o)$ and $\mathbf{m}^a(C) = \mathbf{m}(C)$, it follows by substitution that $\mathbf{m}^a(a) \in \mathbf{m}^a(C)$. Hence, \mathbf{m}^a makes Ca true. And because $(Ca \rightarrow Da)$ is also true on \mathbf{m}^a , it follows that \mathbf{m}^a makes Da true.

From this it follows that $\mathbf{m}^a(a) \in \mathbf{m}^a(D)$. We know that $\mathbf{m}^a(a) = \mathbf{m}(o)$ and $\mathbf{m}^a(D) = \mathbf{m}(D)$, so by substitution we get: $\mathbf{m}(o) \in \mathbf{m}(D)$. Thus, \mathbf{m} makes Do true.

Any model that makes the LHS true also makes the RHS true. Therefore, the entailment holds.¹⁴ ■

Assume that we have an entailment that holds when the set Δ is empty; i.e., $\models \phi$. When an entailment holds, every model \mathbf{m} must either make a sentence on the left false or the sentence on the right true (definition of \models). Because, in this case, there are no sentences on the LHS, every model must make the RHS, ϕ , true. Therefore, as was the case with SL in chapter 2, $\models \phi$ iff ϕ is QT.

Definition 3.45. Two QL1 sentences θ and ϕ are *quantificationally equivalent* iff all models for θ and ϕ assign them the same truth value, which is the same as saying they entail each other, i.e. $\theta \models \phi$ and $\phi \models \theta$.

Definition 3.46. Two QL1 sentences θ and ϕ are *quantificationally contradictory* iff all models for θ and ϕ assign them opposite truth values, which is the same as saying that each sentence is equivalent to the negation of the other.

¹⁴This entailment resembles the argument discussed at the beginning of the chapter: (1) All women are mortal, (2) Ophelia is a woman, therefore (3) Ophelia is mortal. To see this, interpret C as the set of women and D as the set of mortals.

Definition 3.47. Two QL1 sentences θ and ϕ are *quantificationally contrary* iff they cannot both be true in the same model \mathbf{m} .

Definition 3.48. Two QL1 sentences θ and ϕ are *quantificationally subcontrary* iff they cannot both be false in the same model \mathbf{m} .

Definition 3.49. Two QL1 sentences θ and ϕ are *quantificationally independent* iff none of the above hold (including entailment), i.e. iff there are four models:

- (1) A model in which both θ and ϕ are true;
- (2) A model in which both θ and ϕ are false;
- (3) A model in which θ is true and ϕ is false; and
- (4) A model in which θ is false and ϕ is true.

First, a minor note. In QL1 we have both formulas and sentences. (Remember that all sentences are also formulas, but not all formulas are sentences.) Because we do not assess formulas that *aren't* sentences for truth value, none of the definitions above apply to them. These definitions only make sense for QL1 sentences.

Except for the fact that we're considering sentences of QL1 instead of SL, and models for QL1 sentences instead of models for SL sentences, these definitions are exactly the same as the corresponding ones for SL. We might say that these definitions have the same “structure”. The *ideas* of equivalence, being contradictory, etc., haven't changed, even though the details of the definitions are a little different.

The following four facts from SL also hold for QL1 (compare with the examples in section 2.3.4). (1) Contradictory sentences are also contrary, but sentences can be contrary without being contradictory: e.g. $C \wedge D$ and $C \wedge \sim D$. (2) Contradictory sentences are also subcontrary, but sentences can be subcontrary without being contradictory: e.g. D and $C \vee \sim D$. (3) If two sentences are both contrary and subcontrary, they are contradictory. (4) Any two atomic sentences are independent of each other. Because every sentence of SL is also a sentence of QL1, the examples given here still work.

Finally, recall from section 2.3.3 that in SL we have the important simple theorem (Thm. 2.48, on page 33) that for all SL sentences ϕ and θ , $\phi \models \theta$ iff $\models (\phi \rightarrow \theta)$. The same theorem holds for QL1 sentences, and the proof is more or less the same.

Theorem 3.50. QL1 Exportation Theorem: *For all QL1 sentences ϕ and θ , $\phi \models \theta$ iff $\models (\phi \rightarrow \theta)$.*

In addition, all the generalizations of this theorem given in the next (Thm. 2.49) also hold for QL1 sentences, and again the proofs are more or less the same. We will not explicitly restate this theorem for QL1.

3.4 Exercises

3.4.1 Formulas, Order, and Subformulas

Which of the following are QL1 *formulas*? For those that are formulas, what is their order? How many subformulas does each have?

1. $\forall x(H'x \rightarrow G'x)$
2. $\forall x(H'x \rightarrow G''x)$
3. $\forall x(H'x \rightarrow G'_7x)$
4. $\forall x\forall z(H'x \rightarrow G''xy)$
5. $\exists y\forall x(H'x \rightarrow G'x)$
6. $\forall t(H'x \rightarrow G'x)$
7. $H'x \vee G'x$
8. $\forall x(H'y \wedge G'z)$

	Symbol	Model	
		Pos Int	States
Universe:		The set of positive integers	The set of states
Sent. Let.:	A	true	false
	B	true	false
	C	false	true
	D	true	false
	E	true	false
	G	false	true
Constants:	a	1	Louisiana
	b	9	Maine
	c	72	Georgia
	d	3	Nebraska
	e	1	New Mexico
	f	2	Texas
1-place:	A'	all pos int	Midwestern
	B'	empty set	name with > 5 letters
	C'	even	Coastal
	D'	odd	on the Pacific coast
	E'	prime	{Ohio}
	G'	multiple of 7	{Ohio,Alabama}

Table 3.2: Example Models

3.4.2 Truth in a Model

Give the truth value of each of the following sentences on both of the models found in table 3.2 (on the previous page).

- | | |
|---------------------------------------|---|
| 1. $\exists x Gx$ | 8. $\exists x(Cx \wedge Dx)$ |
| 2. $\sim \exists x Gx$ | 9. $\sim \exists x(Cx \wedge Dx)$ |
| 3. $\exists x \sim Gx$ | 10. $\forall x(Cx \wedge Dx)$ |
| 4. $\forall x Gx$ | 11. $\forall x(Cx \rightarrow Dx)$ |
| 5. $\sim \forall x Gx$ | 12. $\forall x Cx \rightarrow \forall x Dx$ |
| 6. $\forall x \sim Gx$ | 13. $\sim \forall x(Cx \rightarrow Dx)$ |
| 7. $\exists x Cx \wedge \exists x Dx$ | 14. $\exists x(Cx \rightarrow Dx)$ |

3.4.3 Quantificational Truth Problems

For each sentence below, say whether or not it's a quantificational truth. If so, prove it. If not, give a model \mathbf{m} that makes it false.

- | | |
|---|---|
| 1. $\forall y(Ay \rightarrow By) \vee \forall y(By \rightarrow Ay)$ | 5. $\exists y(Ay \rightarrow By) \rightarrow (\forall y Ay \rightarrow \forall y By)$ |
| 2. $\exists y(Ay \rightarrow By) \vee \exists y(By \rightarrow Ay)$ | 6. $\forall y \sim Ay \rightarrow \sim \exists y Ay$ |
| 3. $\forall y(Ay \rightarrow By) \rightarrow (\exists y Ay \rightarrow \exists y By)$ | 7. $\sim \exists y Ay \rightarrow \forall y \sim Ay$ |
| 4. $\exists y(Ay \rightarrow By) \rightarrow (\exists y Ay \rightarrow \exists y By)$ | 8. $\sim \forall y Ay \rightarrow \exists y \sim Ay$ |
| 9. $\forall y(Ay \rightarrow By) \rightarrow (\forall y Ay \rightarrow \forall y By)$ | |
| 10. $\forall z(Az \rightarrow (Bz \vee Cz)) \rightarrow (\forall z(Az \rightarrow Bz) \vee \forall z(Az \rightarrow Cz))$ | |
| 11. $\forall y(Ay \rightarrow By) \rightarrow (\forall y(By \rightarrow Cy) \rightarrow \forall y(Ay \rightarrow Cy))$ | |
| 12. $\forall y(Ay \rightarrow By) \rightarrow (\forall y(Cy \rightarrow By) \rightarrow \forall y(Ay \rightarrow Cy))$ | |
| 13. $\forall y(Ay \rightarrow By) \rightarrow (\exists y(By \rightarrow Cy) \rightarrow \forall y(Ay \rightarrow Cy))$ | |
| 14. $\forall y(Ay \rightarrow By) \rightarrow (\exists y(By \rightarrow Cy) \rightarrow \exists y(Ay \rightarrow Cy))$ | |

3.4.4 Entailment Problems for QL1

For each entailment below, either prove that it holds or show that it doesn't hold by giving a model that make the sentences on the LHS of the turnstile true and the sentence on the RHS false.

1. $\forall y(Ay \rightarrow By), \forall yAy \models \forall yBy$
2. $\forall y(Ay \rightarrow By), \exists yAy \models \exists yBy$
3. $\exists y(Ay \rightarrow By), \exists yAy \models \exists yBy$
4. $\forall yAy \rightarrow \forall yBy \models \forall y(Ay \rightarrow By)$
5. $\exists y(Ay \vee By) \models \exists yAy \vee \exists yBy$
6. $\exists y(Ay \rightarrow By), \forall yAy \models \forall yBy$
7. $\forall z(Az \rightarrow (Bz \vee Cz)) \models (\forall z(Az \rightarrow Bz) \vee \forall z(Az \rightarrow Cz))$
8. $\forall y(Ay \rightarrow By), \exists y(By \rightarrow Cy) \models \exists y(Ay \rightarrow Cy)$

3.4.5 Relations Between QL1 Sentences

For each sentence below, say whether it entails, it's entailed by, is equivalent to, contradicts, is contrary to, is subcontrary to, or is independent from each of the other sentences.

1. $\forall z(Gz \rightarrow Dz)$
2. $\forall zGz \rightarrow \forall zDz$
3. $\exists z(Gz \wedge \sim Dz)$
4. $\exists z(Gz \wedge Dz)$
5. $\forall z(Gz \wedge Dz)$
6. $\exists z(Gz \rightarrow Dz)$
7. $\forall z(Gz \rightarrow \sim Dz)$

Chapter 4

Quantifier Language II

4.1 The Language QL

4.1.1 Symbols

In this chapter we extend our language to include many-place predicates. The resulting language is QL, and its development was a significant event in the history of logic.¹ The 2-place predicates correspond roughly to what you get if you take an English sentence and remove two names, leaving blanks, e.g.:

1. ‘Goliath is taller than David’ \Rightarrow ‘_____ is taller than _____’

We may think of 2-place predicates as expressing a 2-place *relation*. In the above example, we have the ‘taller than’ relation. This relation holds between two objects when one is taller than the other. Another 2-place relation is ‘loves’:

2. ‘Juliet Capulet loves Romeo Montague’ \Rightarrow ‘_____ loves _____’

The ‘loves’ relation holds when one person—or object of whatever kind—loves another. We may understand 3-place predicates in a similar way. For example:

3. ‘Three is between two and four’ \Rightarrow ‘_____ is between _____ and _____’

For any $n \geq 2$, an n -place predicate can mirror a corresponding n -place relation. The introduction of many-place predicates significantly increases the power of our formal language. Consider the following argument:

4. Socrates is older than Plato.

¹The development of QL goes back to Gottlob Frege (1879; 1966 [1891]; 1893/1903), O. H. Mitchell (1883) and Charles S. Peirce (1883), with Frege’s work being independent of and unknown to the latter two. See Church 1956: 288 and Hodges 2001b: 34. Although it’s probably safe to say that Frege and Peirce/Mitchell developed quantificational logic independently, the extent to which Peirce and his students (like Mitchell) knew of Frege’s work is a matter of debate. It’s clear they at least knew of Frege. E.g., Ladd-Franklin (1883) cites Frege’s (1879) through a review of it by Ernst Schröder. See (Dipert 1984) for a brief discussion on the situation.

5. Plato is older than Aristotle.

Therefore,

6. Socrates is older than Aristotle.

This argument is a good one, but we cannot represent it as an entailment in QL1. The full language of QL can handle such arguments because it can represent the ‘older than’ relation with a 2-place predicate. One caveat is necessary—the last sentence isn’t a logical consequence of the first two. The first two sentences only entail the third if we also assume that the ‘older than’ relation is transitive. Roughly, the way to express this transitivity would be as follows:

For any x , y , and z such that x is older than y and y is older than z ,
 x is older than z .

Thankfully, we can express this in QL, using quantifiers and 2-place predicates.

QL has all the basic symbols of QL1, plus predicate letters for n -placed predicates, for every integer n such that $n \geq 2$.

Definition 4.1. The *basic symbols* of QL are:

- (1) Logical Connectives, Punctuation Symbols, Sentence Letters, Individual Constants, Individual Variables: same as QL1
 - (2) 1-Place Predicates: $A', B', \dots, T', A'_1, B'_1, \dots, T'_1, A'_2, B'_2, \dots$
 - (3) 2-Place Predicates: $A'', B'', \dots, T'', A''_1, B''_1, \dots, T''_1, A''_2, B''_2, \dots$
 - (4) 3-Place Predicates: A''', B''', \dots
- . . . and so on for all positive integers.

The only difference in appearance between 1-place and 2-place predicates is that the former have only one prime mark and the latter have two. Like their 1-place counterparts, 2-place predicates go from ‘ A'' ’ to ‘ T''' ’. And the same also goes for every n -place predicate.

The superscript prime marks play the significant logical role of marking the arity, or number of places, of the predicate, while the subscripts play the lesser role of making sure we have enough predicates. For every integer n , QL contains an infinite number of n -place predicates.

4.1.2 Formulas of QL

Our ultimate interest is in sentences of QL, but as with QL1 we must first define formulas. The second base clause expands the definition from the QL1 definition of formula.

Definition 4.2. The *formulas of QL* are given by the following recursive definition:

Base Clauses:

- (1) A sentence letter (atomic sentence of SL) is a formula.
- (2) An n -place predicate followed by n occurrences (tokens) of individual constants or variables is a formula.

Generating Clauses:

- (1) If ϕ is a formula, then so is $\sim\phi$.
- (2) If ϕ and θ are formulas, then so are $(\phi \rightarrow \theta)$ and $(\phi \leftrightarrow \theta)$.
- (3) If all of $\phi_1, \phi_2, \phi_3, \phi_4, \dots, \phi_n$ are formulas (the list must include at least two formulas and be finite), then so are $(\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4 \wedge \dots \wedge \phi_n)$ and $(\phi_1 \vee \phi_2 \vee \phi_3 \vee \phi_4 \vee \dots \vee \phi_n)$.
- (4) If ϕ is a formula and it does not contain an expression of the form $\forall\alpha$ or $\exists\alpha$ for some QL variable α , then $\forall\alpha\phi$ and $\exists\alpha\phi$ are formulas.

Closure Clause: A string of symbols is a formula iff it can be generated by the clauses above.

$A'b$ is a formula (an atomic one, to be specific), as is $A''xa$. But $B''x$ is not a formula, because it has a 2-place predicate followed only by one individual variable. To determine whether some string is a formula, we must count *tokens* of variables and constants. For example, $C'''xxx$ is a formula because it has a 3-place predicate followed by three tokens of an individual variable.

From base clause 2 we know that $G''xy$ is a formula, and so from clause 4 we know that the following are also formulas. (The list is not exhaustive.)

- | | |
|---------------------|-------------------------------|
| 7. $\forall xG''xy$ | 10. $\exists y\forall xG''xy$ |
| 8. $\exists xG''xy$ | 11. $\forall x\exists yG''xy$ |
| 9. $\exists zG''xy$ | 12. $\forall x\forall zG''xy$ |

But $\forall x\forall xG''xy$ is *not* a formula, because it's of the form $\forall x\phi$ where ϕ is a formula that contains the expression $\forall x$.

As in SL and QL1, we have unofficial formulas to improve readability.

Definition 4.3. A string of symbols is an *unofficial* formula iff we can obtain it from an official formula by

- (1) deleting outer parentheses,
- (2) replacing official parentheses () with square brackets [] or curly brackets { },
or
- (3) omitting primes ' on a predicate letter.

A unique official formula can always be reconstructed from an unofficial formula.

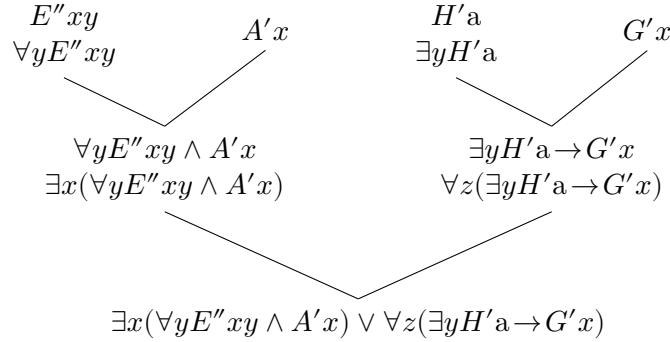
4.1.3 Other Properties of Formulas

The concepts of subformula, order, main connective, and construction tree for formulas of QL are the same as in QL1.²

Example 4.4. Consider the formula $\exists x(\forall yE''xy \wedge A'x) \vee \forall z(\exists yH'a \rightarrow G'x)$. This is a disjunction; its main connective is vee, \vee . It has eleven subformulas:

- | | |
|---|---|
| (1) $\exists x(\forall yE''xy \wedge A'x) \vee \forall z(\exists yH'a \rightarrow G'x)$ | |
| (2) $\exists x(\forall yE''xy \wedge A'x)$ | (7) $\forall z(\exists yH'a \rightarrow G'x)$ |
| (3) $\forall yE''xy \wedge A'x$ | (8) $\exists yH'a \rightarrow G'x$ |
| (4) $\forall yE''xy$ | (9) $\exists yH'a$ |
| (5) $A'x$ | (10) $G'x$ |
| (6) $E''xy$ | (11) $H'a$ |

The construction tree of the formula is:



As you can see from the construction tree, the order of the formula is 5.

4.1.4 Sentences of QL

Sentences, atomic sentences, and unofficial sentences of QL are defined exactly as in QL1.³

4.2 Models

4.2.1 Models in QL

As you can imagine, models in QL are very similar to models of QL1 except that they accommodate many-place predicates.

Definition 4.5. A *model* for ϕ , \mathbf{m} , consists of:

²See section 3.1.3 of the last chapter.

³See section 3.1.4 of the last chapter.

- (1) an assignment of a truth value \top or \bot to each sentence letter in ϕ ;
- (2) a non-empty set U , called the *universe* or *domain*;
- (3) an assignment of an object from U to each individual constant in ϕ ;
- (4) an assignment of a subset of U to each 1-place predicate in ϕ ;
- (5) an assignment of a set of ordered n -tuples to each n -place predicate in ϕ . The objects in each n -tuple are members of U .⁴

The only new part of the definition of model for ϕ in QL is clause (5). Given an n -place predicate, like B''' , we let $\mathbf{m}(B''')$ be the set of ordered n -tuples (in this case, 3-tuples) assigned to B''' by \mathbf{m} . For an illustration, imagine a model \mathbf{m} on which B''' stands for the ‘between’ relation for positive integers from 1 to 4; i.e., such that the first member of each ordered 3-tuple is a number between the second and third members. The 3-tuple $\langle 2, 1, 3 \rangle$ is one example. The model \mathbf{m} would assign all such 3-tuples to B''' as follows:

$$\mathbf{m}(B''') = \{\langle 2, 1, 3 \rangle, \langle 2, 1, 4 \rangle, \langle 3, 1, 4 \rangle, \langle 3, 2, 4 \rangle\}$$

For a second illustration, consider these three people: Jack, Jill, and Bill. Let’s say that Jack is taller than Jill, and Jill is taller than Bill. Let \mathbf{m} be a model on which $\mathbf{m}(A'')$ is the ‘taller than’ relation. The assignment to A'' would be the following set of ordered pairs:

$$\mathbf{m}(A'') = \{\langle \text{Jack}, \text{Jill} \rangle, \langle \text{Jill}, \text{Bill} \rangle, \langle \text{Jack}, \text{Bill} \rangle\}$$

4.2.2 Truth in a Model

The definition of truth in a model for QL is exactly the same as in QL1 except with an additional clause for many-place predicates.

Definition 4.6. The following clauses fix when a QL sentence θ is *true* (or *false*) on a model for θ , \mathbf{m} :

- (1) A sentence letter ϕ is true on \mathbf{m} iff \mathbf{m} assigns true to it, i.e. iff $\mathbf{m}(\phi) = \top$.
- (2) An atomic sentence Pt with a 1-place predicate P and an individual term t is true on \mathbf{m} iff what \mathbf{m} assigns to the individual term t is in the set \mathbf{m} assigns to the predicate, i.e. iff $\mathbf{m}(t) \in \mathbf{m}(P)$.
- (3) An atomic sentence $Pt_1 \dots t_n$ with an n -place predicate P is true on \mathbf{m} iff $\langle \mathbf{m}(t_1), \mathbf{m}(t_2), \dots, \mathbf{m}(t_n) \rangle \in \mathbf{m}(P)$.
- (4) A negation $\sim\phi$ is true on \mathbf{m} iff the unnegated formula ϕ is false on \mathbf{m} .

⁴For a reminder on what an n -tuple is, see section 1.4.4 of Chapter 1.

- (5) A conjunction $(\phi_1 \wedge \dots \wedge \phi_n)$ is true on \mathbf{m} iff all conjuncts ϕ_1, \dots, ϕ_n are true on \mathbf{m} .
- (6) A disjunction $(\phi_1 \vee \dots \vee \phi_n)$ is true on \mathbf{m} iff at least one disjunct ϕ_1, \dots, ϕ_n is true on \mathbf{m} .
- (7) A conditional $(\psi \rightarrow \phi)$ is true on \mathbf{m} iff the LHS ψ is false or the RHS ϕ is true on \mathbf{m} .
- (8) A biconditional $(\psi \leftrightarrow \phi)$ is true on \mathbf{m} iff both sides, ψ and ϕ , have the same truth value on \mathbf{m} .
- (9) A universal quantification $\forall \alpha \phi$ is true on \mathbf{m} iff $\phi t / \alpha$ is true on *all* t -variants of \mathbf{m} (where t is the first *constant* not contained in ϕ).
- (10) An existential quantification $\exists \alpha \phi$ is true on \mathbf{m} iff $\phi t / \alpha$ is true on *some* t -variant of \mathbf{m} (where t is the first *constant* not contained in ϕ).
- (11) A sentence ϕ is false on \mathbf{m} iff ϕ is not true on \mathbf{m} .

Let's look at a simple example to see how QL truth works for a two-place predicate. Consider a model \mathbf{m} with the following assignments:

$\mathbf{m}(j) = \text{Jack}$
 $\mathbf{m}(i) = \text{Jill}$
 $\mathbf{m}(b) = \text{Bill}$
 $\mathbf{m}(A'') = \{\langle \text{Jack}, \text{Jill} \rangle, \langle \text{Jill}, \text{Bill} \rangle, \langle \text{Jack}, \text{Bill} \rangle\}$

Is the QL sentence $A''jb$ true on \mathbf{m} ? To find out, we must check the set $\mathbf{m}(A'')$ to see if $\langle \mathbf{m}(j), \mathbf{m}(b) \rangle$ is a member. We see that $\langle \mathbf{m}(j), \mathbf{m}(b) \rangle = \langle \text{Jack}, \text{Bill} \rangle$. We then find that $\langle \text{Jack}, \text{Bill} \rangle$ is a member of $\mathbf{m}(A'')$, so $A''jb$ is true on \mathbf{m} .

Next, consider the sentence $A''bi$ on the same model. This sentence is true on \mathbf{m} iff $\langle \mathbf{m}(b), \mathbf{m}(i) \rangle$ is a member of the set $\mathbf{m}(A'')$. We know that $\langle \mathbf{m}(b), \mathbf{m}(i) \rangle = \langle \text{Bill}, \text{Jill} \rangle$. But $\langle \text{Bill}, \text{Jill} \rangle$ is not a member of $\mathbf{m}(A'')$, so $A''bi$ is false on \mathbf{m} . (Remember that $\langle \text{Bill}, \text{Jill} \rangle$ is not the same as $\langle \text{Jill}, \text{Bill} \rangle$. Order matters!)

Let's try more complicated examples, using the models provided on the table below:

Symbol	Model	
	Pos Int	States
Universe:	The set of positive integers	The set of US states (2015)
Sent. Let.: A	true	false
B	true	false
C	false	true

Table 4.1: Example Models
Continued next Page

Continued from Previous Page

	Symbol	Model	
		Pos Int	States
	D	true	false
	E	true	false
	G	false	true
Constants:	a	1	Louisiana
	b	9	Maine
	c	72	Georgia
	d	3	Nebraska
	e	1	New Mexico
	f	2	Texas
1-place:	A'	all pos int	Midwestern
	B'	empty set	name with > 5 letters
	C'	even	Coastal
	D'	odd	on the Pacific Coast
	E'	prime	{Ohio}
	G'	multiple of 7	{Ohio, Alabama}
2-place:	A''	first > second	share a border
	B''	are equal	first is north of second
	C''	first = 2 times second	first > second (area)
	D''	sum of them equals 7	first > second (population)
	E''	first < second	first is west of second
	G''	are relatively prime	both coastal, or neither
3-place:	A'''	all equal	all same population
	B'''	first < second < third	first is north of others
	C'''	all odd or all even	first > second > third (area)
	D'''	first + second = third	first + second > third (area)
	E'''	first \times second = third	first is west of the others
	G'''	are all relatively prime	at least two coastal

Table 4.1: Example Models

While looking over the many-place predicates in the table, you'll notice that we don't actually list sets of n -tuples. Instead, for each many-place predicate we provide a brief description of some relation which could be used to identify such a set. This will be our usual practice. Writing out all the specific n -tuples in our chart would be excessively tedious for the model *States*, and impossible for the model *Pos Int*.⁵

⁵Finite beings often have difficulty performing tasks with an infinite number of steps.

Example 4.7. Determine the truth values of the sentences $\forall y \exists x A''xy$ and $\exists x \forall y A''xy$ on the model *Pos Int* in table 4.1. The only difference between these two sentences is the order of the quantifiers. If we can show that they differ in truth value on the same model, then we will have established that quantifier order matters for assessing truth in a model.

Proof: First, let's consider the sentence $\forall y \exists x A''xy$.

(i) $\forall y \exists x A''xy$ is true on *Pos Int* iff $\exists x A''xa$ is true on all a-variants of *Pos Int*. The sentence $\exists x A''xa$ is true on an a-variant of *Pos Int*, $Pos Int^a$, iff $A''ba$ is true on some b-variant of $Pos Int^a$ (definition of truth, \exists). *Pos Int* assigns to A'' the set of ordered pairs such that the first is greater than the second. So, $A''ba$ is true on some b-variant of $Pos Int^a$ when it assigns a larger number to b than it assigns to a.

It doesn't really matter what $Pos Int^a$ assigns to a. There is some larger number that a b-variant of $Pos Int^a$ can assign to b. Hence, regardless of what number $Pos Int^a(a)$ is, $\exists x A''xa$ is true on $Pos Int^a$.

For every assignment to a, there is some assignment to b that is larger. So $\exists x A''xa$ is true on all a-variants of *Pos Int*. Thus, the sentence $\forall y \exists x A''xy$ is true on *Pos Int*.

(ii) $\exists x \forall y A''xy$ is true on *Pos Int* iff $\forall y A''ay$ is true on some a-variant of *Pos Int*. The sentence $\forall y A''ay$ is true on some a-variant of *Pos Int*, $Pos Int^a$, iff $A''ab$ is true on all b-variants of $Pos Int^a$.

But no matter how large of a number that a b-variant of $Pos Int^a$ assigns to a, there will always be some larger number that it could assign to b. So, $\forall y A''ay$ is false on all a-variants of *Pos Int*. Thus, $\exists x \forall y A''xy$ is false on *Pos Int*. ■

The different quantifier order affects the meanings of the sentences. The sentence $\forall y \exists x A''xy$ relative to *Pos Int* means, roughly, that 'for every positive integer there is a larger positive integer.' The sentence $\exists x \forall y A''xy$ relative to *Pos Int* means, roughly, that 'there is some positive integer that is larger than every positive integer.' Clearly the former is true and the latter isn't.⁶

Example 4.8. The sentence $\forall x \forall y \forall z ((Cxy \wedge Dxyz) \rightarrow Byxz)$ is (i) true in the model *Pos Int* given in table 4.1 (on page 90), but (ii) is false in the model *States*.

Proof: (i) The model *Pos Int* assigns to *C* the set of positive integer pairs $\langle u, v \rangle$ such that $u = 2v$, to *D* the set of positive integer triples $\langle u, v, w \rangle$ such that $u + v = w$, and to *B* the set of positive integer triples $\langle v, u, w \rangle$ such that $v < u < w$.

Consider an instantiation of the sentence $\forall x \forall y \forall z ((Cxy \wedge Dxyz) \rightarrow Byxz)$; let's say $((Cab \wedge Dabc) \rightarrow Bbac)$.⁷ Now consider an a, b, c-variant of *Pos Int* such that $Pos Int^{abc}(a) = 2Pos Int^{abc}(b)$ and $Pos Int^{abc}(a) + Pos Int^{abc}(b) = Pos Int^{abc}(c)$.

⁶We will further discuss translations of sentences in formal languages in Chapter 5.

⁷An 'instantiation' of a quantified sentence is a second sentence in which the quantifier is first dropped, and then the newly unbound variables are replaced with constants. In this case we drop all three quantifiers and substitute three different constants for the three variables in the original sentence.

Both Cab and $Dabc$ are true on any such variant. The RHS of the conditional, $Bbac$, is also true on that variant. Because $Pos\ Int^{abc}(a) = 2Pos\ Int^{abc}(b)$, it's clear that $Pos\ Int^{abc}(b) < Pos\ Int^{abc}(a)$ (0 is not a positive integer). And because $Pos\ Int^{abc}(b) + Pos\ Int^{abc}(a) = Pos\ Int^{abc}(c)$, it's clear that $Pos\ Int^{abc}(a) < Pos\ Int^{abc}(c)$. So, $Bbac$ is true on any such variant of $Pos\ Int$. Thus, every variant of $Pos\ Int$ such that $Pos\ Int^{abc}(a) = 2Pos\ Int^{abc}(b)$ and $Pos\ Int^{abc}(a) + Pos\ Int^{abc}(b) = Pos\ Int^{abc}(c)$ makes the conditional $((Cab \wedge Dabc) \rightarrow Bbac)$ true.

Any variant of $Pos\ Int$ such that either $Pos\ Int^{abc}(a) \neq 2Pos\ Int^{abc}(b)$ or $Pos\ Int^{abc}(a) + Pos\ Int^{abc}(b) \neq Pos\ Int^{abc}(c)$ makes either Cab or $Dabc$ false (def. of truth, \wedge), and hence makes the conditional $((Cab \wedge Dabc) \rightarrow Bbac)$ true (def. of truth, \rightarrow).

Therefore, every a, b, c-variant of $Pos\ Int$ makes the conditional $((Cab \wedge Dabc) \rightarrow Bbac)$ true. Put in a slightly different way, every c-variant of $Pos\ Int^{ab}$ makes $((Cab \wedge Dabc) \rightarrow Bbac)$ true. Hence $\forall z((Cab \wedge Dabz) \rightarrow Bbaz)$ is true on every a, b-variant of $Pos\ Int$.

We may repeat this kind of reasoning to add quantifiers and replace the two remaining constants with variables. Every b-variant of $Pos\ Int^a$ makes $\forall z((Cab \wedge Dabz) \rightarrow Bbaz)$ true. So, by the definition of truth for \forall , $\forall y\forall z((Cay \wedge Dayz) \rightarrow Byz)$ is true on all a-variants of $Pos\ Int$. It follows that $\forall x\forall y\forall z((Cxy \wedge Dxyz) \rightarrow Byxz)$ is true on $Pos\ Int$.

(ii) The model *States* assigns to C pairs of states $\langle u, v \rangle$ where $u > v$ (area), to D triples of states $\langle u, v, w \rangle$ where $u + v > w$ (area), and to B triples of states $\langle v, u, w \rangle$ where v is north of u and w . As in the last section, let's reason using an instance of $\forall x\forall y\forall z((Cxy \wedge Dxyz) \rightarrow Byxz)$; again, let's use $((Cab \wedge Dabc) \rightarrow Bbac)$.

Now consider an a, b, c-variant of *States* that assigns a to Alaska, b to Delaware, and c to Rhode Island. Alaska has an area of approximately 1.7×10^6 km², Delaware an area of approximately 2.5×10^3 km², and Rhode Island an area of approximately 1.5×10^3 km². Hence $States^{abc}(a) > States^{abc}(b)$ (area), $States^{abc}(a) + States^{abc}(b) > States^{abc}(c)$ (area), but $States^{abc}(b)$ is not north of both $States^{abc}(a)$ and $States^{abc}(c)$; that is, Delaware is not north of Alaska. So, by clause (3) of the definition of truth, 4.6, Cab and $Dabc$ are true on $States^{abc}$, while $Bbac$ is false on $States^{abc}$. So by the definition of truth (\wedge and \rightarrow), $((Cab \wedge Dabc) \rightarrow Bbac)$ is false on $States^{abc}$.

We may word this slightly differently: there is a c-variant of $States^{ab}$ on which $((Cab \wedge Dabc) \rightarrow Bbac)$ is false. It follows (by the definition of truth for \forall) that $\forall z((Cab \wedge Dabz) \rightarrow Bbaz)$ is false on $States^{ab}$. So, there is, in turn, a b-variant of $States^a$ on which $\forall z((Cab \wedge Dabz) \rightarrow Bbaz)$ is false. So, again, by the definition of truth for \forall , that $\forall y\forall z((Cay \wedge Dayz) \rightarrow Byz)$ is false on $States^a$.

Finally, because there is an a-variant of *States* on which $\forall y\forall z((Cay \wedge Dayz) \rightarrow Byz)$ is false, $\forall x\forall y\forall z((Cxy \wedge Dxyz) \rightarrow Byxz)$ is false on *States* (definition of truth, \forall). ■

4.2.3 Logical Truth: QT, QF, & QC

The concepts of quantificational truth (QT), quantificational falsehood (QF), and quantificational contingency (QC) are defined for QL exactly as in QL1.⁸

4.2.4 Entailment and other Relations

The concepts for entailment and the other logical relations are also defined for QL exactly as in QL1.⁹

4.3 The Dragnet Theorem

In example 3.43 we were able to establish that the entailment ' $\forall xGx \models Gb$ ' holds by reasoning as follows:

- (i) Any model \mathbf{m} that makes $\forall xGx$ true makes Ga true on all a -variants of \mathbf{m} .
- (ii) All a -variants of \mathbf{m} share the same domain and make the same set assignment to G .
- (iii) There is some a -variant of \mathbf{m} , \mathbf{m}^a , such that $\mathbf{m}^a(a) = \mathbf{m}(b)$.
- (iv) Because $\mathbf{m}^a(a) \in \mathbf{m}^a(G)$ and $\mathbf{m}^a(G) = \mathbf{m}(G)$, $\mathbf{m}^a(a) \in \mathbf{m}(G)$.
- (v) Because $\mathbf{m}^a(a) \in \mathbf{m}(G)$ and $\mathbf{m}^a(a) = \mathbf{m}(b)$, $\mathbf{m}(b) \in \mathbf{m}(G)$. Thus, Gb is true on \mathbf{m} .

By analogous reasoning, we can show that $\forall xGx$ entails Gc , Gd , Ge , and so on. But what if we want to prove that the entailment holds, regardless of what constant we pick? Proving that the entailment always holds, regardless of the constant, requires the use of metatheory. For such proofs we'll need to use a metavariable, in this case t : $\forall xGx \models Gxt/x$, where t is any constant.

And we will also want to prove rather stronger entailments, such as the following: $\forall x\phi \models \phi b/x$, where ϕ is some QL sentence with only x free.

Such proofs are much easier to complete with the following theorem, which we call the 'Dragnet Theorem'. Let's say that ϕ and ϕ^* are sentences of QL such that $\phi^* = \phi s/t$, where s and t are constants.¹⁰ Let's also say that we have two models, \mathbf{m}_1 and \mathbf{m}_2 , that make all the same assignments except that what \mathbf{m}_1 assigns to t , \mathbf{m}_2 assigns to s . I.e., $\mathbf{m}_1(t) = \mathbf{m}_2(s)$. It would seem that ϕ is true on \mathbf{m}_1 iff ϕ^* is true on \mathbf{m}_2 . This intuitively plausible claim is true, and the Dragnet theorem proves it. There are

⁸See section 3.2.4 on page 77 of the last chapter.

⁹See section 3.3 on page 79 of the last chapter.

¹⁰Recall from section 3.26, on page 71 in Chapter 3 that $\phi s/t$ is the sentence you get by replacing each unbound token of t in ϕ with a token of s . With Dragnet, we are concerned solely with cases in which s and t are constants.

many cases in which Dragnet is crucial for proving properties or relations of sentences (e.g., in theorem 7.11, on page 185, and 7.27, on page 202).¹¹

We need not restrict the claim to pairs of sentences that vary on only one constant. A pair of otherwise identical sentences ϕ and ϕ^* may differ on as many constants as you like, and as long as two models, \mathbf{m}_1 and \mathbf{m}_2 , make the same assignments for the replacement constants in ϕ^* , ϕ is true on \mathbf{m}_1 iff ϕ^* is true on \mathbf{m}_2 . That is, if there are two sentences ϕ and ϕ^* such that $\phi^* = \phi s_1/t_1, s_2/t_2, \dots, s_i/t_i$, and two models \mathbf{m}_1 and \mathbf{m}_2 such that $\mathbf{m}_1(t_1) = \mathbf{m}_2(s_1)$, $\mathbf{m}_1(t_2) = \mathbf{m}_2(s_2)$, ..., and $\mathbf{m}_1(t_i) = \mathbf{m}_2(s_i)$, then ϕ is true on \mathbf{m}_1 iff ϕ^* is true on \mathbf{m}_2 . We prove this theorem below.

Theorem 4.9. The Dragnet Theorem: *If*

- (1) *a QL sentence ϕ contains one or more of each of the constant(s) t_1, t_2, \dots, t_i , and another QL sentence $\phi^* = \phi s_1/t_1, s_2/t_2, \dots, s_i/t_i$; and*
- (2) *The models \mathbf{m}_1 and \mathbf{m}_2 differ only in that what \mathbf{m}_1 assigns to t_1 , \mathbf{m}_2 assigns to s_1 , $\mathbf{m}_1(t_2) = \mathbf{m}_2(s_2)$, ..., $\mathbf{m}_1(t_i) = \mathbf{m}_2(s_i)$,¹²*

then: ϕ is true on \mathbf{m}_1 iff ϕ^ is true on \mathbf{m}_2 .*

We proceed by recursive proof.

Proof: Throughout the proof we treat $*$ as a function that takes a QL sentence and returns the sentence you get by replacing all occurrences of t_1 with s_1 , t_2 with s_2 , and so on. Without further stipulation, $\phi^* = \phi s_1/t_1, s_2/t_2, \dots, s_i/t_i$, $\psi^* = \psi s_1/t_1, s_2/t_2, \dots, s_i/t_i$, $\theta^* = \theta s_1/t_1, s_2/t_2, \dots, s_i/t_i$, and so on for all metavariables. Accordingly, ϕ and ϕ^* satisfy Dragnet condition (1) above; and so do ψ and ψ^* , as well as θ and θ^* , etc.

Additionally, throughout the proof we assume that \mathbf{m}_1 and \mathbf{m}_2 are two arbitrary models that satisfy Dragnet condition (2) above.

Base Step: ϕ is atomic.

- (1) If ϕ is a sentence letter, then there are no constants and ϕ and ϕ^* must be identical. So, clearly, ϕ is true on \mathbf{m}_1 iff ϕ^* is true on \mathbf{m}_2 .
- (2) Say that ϕ is a predicate letter P followed by one constant: Pt . Then ϕ^* is the same, but with a different constant: Ps . In accordance with Dragnet condition (2), \mathbf{m}_1 and \mathbf{m}_2 make all the same assignments except that what \mathbf{m}_1 assigns to t , \mathbf{m}_2 assigns to s . So, $\mathbf{m}_1(P) = \mathbf{m}_2(P)$ and $\mathbf{m}_1(t) = \mathbf{m}_2(s)$. By the definition of truth,

$$Pt \text{ is true on } \mathbf{m}_1 \text{ iff } \mathbf{m}_1(t) \in \mathbf{m}_1(P).$$

¹¹Although the basic claim behind Dragnet seems obviously true, it turns out that stating the theorem precisely and proving is difficult. There are different ways to state the theorem, not all exactly equivalent. See Mates 1972: 66 and Bergmann et al. 2003: 577 for two alternative examples.

¹²In other words, \mathbf{m}_1 and \mathbf{m}_2 make the same assignments to U , the predicates, the sentence letters, and the constants, except that what \mathbf{m}_1 assigns to t_1 , \mathbf{m}_2 assigns to s_1 ; what \mathbf{m}_1 assigns to t_2 , \mathbf{m}_2 assigns to s_2 ; and so on.

But because $\mathbf{m}_1(P) = \mathbf{m}_2(P)$, we can substitute on the RHS to get:

$$Pt \text{ is true on } \mathbf{m}_1 \text{ iff } \mathbf{m}_1(t) \in \mathbf{m}_2(P).$$

We also know that $\mathbf{m}_1(t) = \mathbf{m}_2(s)$, so we can make another substitution to get:

$$Pt \text{ is true on } \mathbf{m}_1 \text{ iff } \mathbf{m}_2(s) \in \mathbf{m}_2(P).$$

Finally, by the definition of truth, we can replace the RHS to get:

$$Pt \text{ is true on } \mathbf{m}_1 \text{ iff } Ps \text{ is true on } \mathbf{m}_2.$$

And that's what we wanted to prove for this base clause.

- (3) Say that ϕ is a predicate letter P followed by n constants, q_1, q_2, \dots, q_n . As we stipulated earlier, ϕ^* is exactly the same, except that some or all of the constants of ϕ have been replaced with other constants. Let's assume, without loss of generality, that $\phi = Pq_1 \dots t_1 \dots t_2 \dots t_i \dots q_n$. That is, let t_1, t_2, \dots , and t_i be the constants of ϕ that will be replaced in ϕ^* . So, ϕ^* is $Pq_1 \dots s_1 \dots s_2 \dots s_i \dots q_n$. By the definition of truth,

$$\phi \text{ is true on } \mathbf{m}_1 \text{ iff } \langle \mathbf{m}_1(q_1), \dots, \mathbf{m}_1(t_1), \dots, \mathbf{m}_1(t_i), \dots, \mathbf{m}_1(q_n) \rangle \in \mathbf{m}_1(P).$$

We know that $\mathbf{m}_1(P) = \mathbf{m}_2(P)$, so we substitute on the RHS to get:

$$\phi \text{ is true on } \mathbf{m}_1 \text{ iff } \langle \mathbf{m}_1(q_1), \dots, \mathbf{m}_1(t_1), \dots, \mathbf{m}_1(t_i), \dots, \mathbf{m}_1(q_n) \rangle \in \mathbf{m}_2(P).$$

Because models \mathbf{m}_1 and \mathbf{m}_2 meet Dragnet condition (2), $\mathbf{m}_1(t_1) = \mathbf{m}_2(s_1)$, $\mathbf{m}_1(t_2) = \mathbf{m}_2(s_2)$, ..., $\mathbf{m}_1(t_i) = \mathbf{m}_2(s_i)$. Hence, by more substitutions on the RHS we get:

$$\phi \text{ is true on } \mathbf{m}_1 \text{ iff } \langle \mathbf{m}_1(q_1), \dots, \mathbf{m}_2(s_1), \dots, \mathbf{m}_2(s_i), \dots, \mathbf{m}_1(q_n) \rangle \in \mathbf{m}_2(P).$$

The models \mathbf{m}_1 and \mathbf{m}_2 otherwise make all the same assignments; so for the constants that aren't changed from ϕ to ϕ^* , they are each assigned the same object on both models. I.e., for the unchanged constants of q_1 through q_n we know that $\mathbf{m}_1(q_1) = \mathbf{m}_2(q_1)$, $\mathbf{m}_1(q_2) = \mathbf{m}_2(q_2)$, ..., $\mathbf{m}_1(q_n) = \mathbf{m}_2(q_n)$. Thus, we can carry out even more substitutions to get:

$$\phi \text{ is true on } \mathbf{m}_1 \text{ iff } \langle \mathbf{m}_2(q_1), \dots, \mathbf{m}_2(s_1), \dots, \mathbf{m}_2(s_i), \dots, \mathbf{m}_2(q_n) \rangle \in \mathbf{m}_2(P).$$

And we also know, by the definition of truth, that:

$$\langle \mathbf{m}_2(q_1), \dots, \mathbf{m}_2(s_1), \dots, \mathbf{m}_2(s_i), \dots, \mathbf{m}_2(q_n) \rangle \in \mathbf{m}_2(P) \text{ iff } \phi^* \text{ is true on } \mathbf{m}_2.$$

So it follows that ϕ is true on \mathbf{m}_1 iff ϕ^* is true on \mathbf{m}_2 .

Inheritance Step:

Recursive Assumption Assume that the Dragnet theorem holds for all QL sentences of order k or less, and that ϕ is an QL sentence of order $k + 1$. Consider the following ways in which ϕ might be of order $k + 1$.

Negation: ϕ is a negation; i.e., is of the form $\sim\psi$. ϕ^* is the result of substituting s_1 for t_1 , s_2 for t_2 , etc., in $\sim\psi$, which is the same as making the substitutions in ψ and putting a ' \sim ' in front. That is, $\sim(\psi)^*$ is the same formula as $(\sim\psi)^*$. Because ψ is of order k , by the recursive assumption (RA):

$$\psi \text{ is true on } \mathbf{m}_1 \text{ iff } \psi^* \text{ is true on } \mathbf{m}_2,$$

and it follows from this that:

$$\psi \text{ is false on } \mathbf{m}_1 \text{ iff } \psi^* \text{ is false on } \mathbf{m}_2.$$

The sentence ψ is false on \mathbf{m}_1 iff $\sim\psi$ is true on \mathbf{m}_1 ; and the same holds for ψ^* . So,

$$\sim\psi \text{ is true on } \mathbf{m}_1 \text{ iff } \sim\psi^* \text{ is true on } \mathbf{m}_2,$$

which is what we want to show.

Conjunction: ϕ is a conjunction; i.e., is of the form $(\psi_1 \wedge \psi_2 \wedge \dots \wedge \psi_n)$. The sentence ϕ^* is the result of substituting s_1 for t_1 , s_2 for t_2 , etc., in ϕ , which is the same as if we make the substitutions in each conjunct and then put ' \wedge '(s) between them, i.e. $(\psi_1^* \wedge \psi_2^* \wedge \dots \wedge \psi_n^*)$.

Each conjunct ψ_j is of order k or lower. (Let ψ_j be the j^{th} conjunct of ϕ , where j is some arbitrary integer from 1 to n .) So, by RA,

$$\psi_j \text{ is true on } \mathbf{m}_1 \text{ iff } \psi_j^* \text{ is true on } \mathbf{m}_2.$$

There is one such biconditional for each conjunct of ϕ . So, conjoining all the left-hand sides and all the right-hand sides of these n biconditionals we get:

All of $\psi_1, \psi_2, \dots, \psi_n$ are true on \mathbf{m}_1 iff all of $\psi_1^*, \psi_2^*, \dots, \psi_n^*$ are true on \mathbf{m}_2 .

So, by the definition of truth for \wedge ,

$$(\psi_1 \wedge \psi_2 \wedge \dots \wedge \psi_n) \text{ is true on } \mathbf{m}_1 \text{ iff } (\psi_1^* \wedge \psi_2^* \wedge \dots \wedge \psi_n^*) \text{ is true on } \mathbf{m}_2.$$

And that's what we want to prove.

Disjunction: We leave this case for the reader to do as an exercise.

Conditional: ϕ is of the form $(\psi \rightarrow \theta)$. By the definition of truth for \rightarrow we know that

$$(\psi \rightarrow \theta) \text{ is true on } \mathbf{m}_1 \text{ iff either (i) } \psi \text{ is false on } \mathbf{m}_1, \text{ or (ii) } \theta \text{ is true on } \mathbf{m}_1.$$

ψ is of order k or lower; so by the RA, we know that:

$$\psi \text{ is true on } \mathbf{m}_1 \text{ iff } \psi^* \text{ is true on } \mathbf{m}_2.$$

From which it follows that:

$$\psi \text{ is false on } \mathbf{m}_1 \text{ iff } \psi^* \text{ is false on } \mathbf{m}_2.$$

Using this and the earlier result in this clause, we substitute to get that

$(\psi \rightarrow \theta)$ is true on \mathbf{m}_1 iff either (i) ψ^* is false on \mathbf{m}_2 , or (ii) θ is true on \mathbf{m}_1 .
 θ is of order k or lower; so by the RA we also know that:

$$\theta \text{ is true on } \mathbf{m}_1 \text{ iff } \theta^* \text{ is true on } \mathbf{m}_2.$$

We substitute again to get:

$(\psi \rightarrow \theta)$ is true on \mathbf{m}_1 iff either (i) ψ^* is false on \mathbf{m}_2 , or (ii) θ^* is true on \mathbf{m}_2 .

Finally, by the definition of truth (\rightarrow), we can substitute to get:

$$(\psi \rightarrow \theta) \text{ is true on } \mathbf{m}_1 \text{ iff } (\psi^* \rightarrow \theta^*) \text{ is true on } \mathbf{m}_2.$$

From which we get what we wanted to show:

$$(\psi \rightarrow \theta) \text{ is true on } \mathbf{m}_1 \text{ iff } (\psi \rightarrow \theta)^* \text{ is true on } \mathbf{m}_2.$$

Biconditional: We leave this case for the reader to do as an exercise.

Universal Quantification: ϕ is a universal quantification; i.e., is of the form $\forall \beta \psi$, where ψ is a formula that has exactly one free variable, β . Because β is a variable, and thus is different from all constants, $(\forall \beta \psi)^* = \forall \beta \psi^*$.

According to the definition of truth for \forall ,

$$\begin{aligned} \forall \beta \psi \text{ is true on } \mathbf{m}_1 &\text{ iff } \psi q / \beta \text{ is true on every } q\text{-variant of } \mathbf{m}_1, \\ &\text{where } q \text{ is the first constant not in } \psi, \end{aligned}$$

and...

$$\begin{aligned} \forall \beta \psi^* \text{ is true on } \mathbf{m}_2 &\text{ iff } \psi^* r / \beta \text{ is true on every } r\text{-variant of } \mathbf{m}_2, \\ &\text{where } r \text{ is the first constant not in } \psi^*.^{13} \end{aligned}$$

There are two differences between the sentences $\psi q / \beta$ and $\psi^* r / \beta$. First, where $\psi q / \beta$ contains the constant q , the sentence $\psi^* r / \beta$ contains the constant r . Second, whereas constants t_1, \dots, t_n are in $\psi q / \beta$, they are replaced by the constants s_1, \dots, s_n in $\psi^* r / \beta$. Apart from these differences $\psi q / \beta$ and $\psi^* r / \beta$ are identical, and so they satisfy Dragnet condition (1).

We must prove that $\forall \beta \psi$ is true on \mathbf{m}_1 iff $\forall \beta \psi^*$ is true on \mathbf{m}_2 . To show this, we can take the RHS from each of the two biconditionals above, and prove the following biconditional:

$$\begin{aligned} (\psi q / \beta \text{ is true on every } q\text{-variant of } \mathbf{m}_1) &\text{ iff} \\ (\psi^* r / \beta \text{ is true on every } r\text{-variant of } \mathbf{m}_2). \end{aligned}$$

¹³The definition of truth for \forall uses ‘ t ’ to stand for the first constant not in the formula in question. Here we instead use ‘ q ’ and ‘ r ’ so that we don’t confuse these with the list of constants t_1, t_2, \dots, t_i . We also take care to distinguish ‘ q ’ and ‘ r ’. We cannot assume that $q = r$, because there are certain cases in which they’re different. For example, consider the case such that $\forall \beta \psi = \forall x(Dx \rightarrow Gxb)$ and $\forall \beta \psi^* = \forall \beta \psi a/b = \forall x(Dx \rightarrow Gxa)$.

Because this will take some work, we separate it off as a subproof.

Subproof: (\Rightarrow) Assume that the LHS of the biconditional is true: $\psi q/\beta$ is true on every q -variant of \mathbf{m}_1 . We want to show that $\psi^* r/\beta$ is true on every r -variant of \mathbf{m}_2 .

Assume some arbitrary q -variant of \mathbf{m}_1 ; call it \mathbf{m}_1^q . Let's give an arbitrary name to the object that \mathbf{m}_1^q assigns to q : 'Kate'. The models \mathbf{m}_1 and \mathbf{m}_2 share the same domain, and thus so do all variants of those models. Therefore, we know that there is an r -variant of \mathbf{m}_2 that assigns Kate to r . Let's call this r -variant \mathbf{m}_2^r . To put it another way, let \mathbf{m}_2^r be such that

$$\mathbf{m}_1^q(q) = \mathbf{m}_2^r(r).$$

We know that $\mathbf{m}_1(t_1) = \mathbf{m}_2(s_1)$, $\mathbf{m}_1(t_2) = \mathbf{m}_2(s_2)$, ..., $\mathbf{m}_1(t_i) = \mathbf{m}_2(s_i)$. We need to show that the same equivalences hold for the model variants, \mathbf{m}_1^q and \mathbf{m}_2^r . The model \mathbf{m}_1 differs from \mathbf{m}_1^q on the assignment to q , but q is the first constant not in ψ . Hence, q is not in the list of constants t_1, \dots, t_i . \mathbf{m}_1 and \mathbf{m}_1^q differ only on q , so it follows that they make the same assignments to each of t_1, \dots, t_i . Analogous reasoning shows that \mathbf{m}_2 and \mathbf{m}_2^r make the same assignments to each of s_1, \dots, s_i . Therefore, $\mathbf{m}_1^q(t_1) = \mathbf{m}_2^r(s_1)$, $\mathbf{m}_1^q(t_2) = \mathbf{m}_2^r(s_2)$, ..., $\mathbf{m}_1^q(t_i) = \mathbf{m}_2^r(s_i)$.

And because $\mathbf{m}_1^q(q) = \mathbf{m}_2^r(r)$, it follows that what \mathbf{m}_1^q assigns to each constant of $\psi q/\beta$ is the same as what \mathbf{m}_2^r assigns to the analogous constant in the corresponding location of $\psi^* r/\beta$. Thus, \mathbf{m}_1^q and \mathbf{m}_2^r satisfy Dragnet condition (2).

The sentences $\psi q/\beta$ and $\psi^* r/\beta$ are each of order k , so, by RA,

$$\psi q/\beta \text{ is true on } \mathbf{m}_1^q \text{ iff } \psi^* r/\beta \text{ is true on } \mathbf{m}_2^r.$$

We initially assumed that $\psi q/\beta$ is true on every q -variant of \mathbf{m}_1 . It follows that $\psi q/\beta$ is true on \mathbf{m}_1^q . Therefore, given the above, $\psi^* r/\beta$ is true on \mathbf{m}_2^r . We also assumed a single q -variant of \mathbf{m}_1 , \mathbf{m}_1^q ; but nothing in our argument depends on anything specific about the object \mathbf{m}_1^q assigns to q . We did nothing more than give it an arbitrary name. So the argument above is perfectly general, and it applies equally for any other q -variant as well. That is, for each q -variant of \mathbf{m}_1 , there is a corresponding r -variant of \mathbf{m}_2 , such that,

$$\mathbf{m}_1^q(q) = \mathbf{m}_2^r(r).$$

Therefore, because $\psi q/\beta$ is true on every q -variant of \mathbf{m}_1 , it follows that $\psi^* r/\beta$ is true on every r -variant of \mathbf{m}_2 .

(\Leftarrow) The argument in this section of the subproof perfectly mirrors that of the last.

First assume the RHS is true. That is, assume that ψ^*r/β is true on every r -variant of \mathbf{m}_2 . We want to show that the LHS follows, i.e., that $\psi q/\beta$ is true on every q -variant of \mathbf{m}_1 .

Assume some arbitrary r -variant of \mathbf{m}_2 , called \mathbf{m}_2^r . Let's name the object that \mathbf{m}_2^r assigns to r 'Irene'. The variants of \mathbf{m}_1 and \mathbf{m}_2 share the same domain, so there's a q -variant of \mathbf{m}_1 that assigns Irene to q . Name that q -variant \mathbf{m}_1^q . So,

$$\mathbf{m}_2^r(r) = \mathbf{m}_1^q(q).$$

As before, $\mathbf{m}_1(t_1) = \mathbf{m}_2(s_1)$, $\mathbf{m}_1(t_2) = \mathbf{m}_2(s_2)$, ..., $\mathbf{m}_1(t_i) = \mathbf{m}_2(s_i)$. The model \mathbf{m}_2 differs from \mathbf{m}_2^r on r , but r is the first constant not in ψ^* . Hence, r is not any of the following constants: s_1, \dots, s_i . \mathbf{m}_2 and \mathbf{m}_2^r differ only on r , so they make the same assignments to each of s_1, \dots, s_i . Likewise, \mathbf{m}_1 and \mathbf{m}_1^q make the same assignments to each of t_1, \dots, t_i . Therefore, $\mathbf{m}_1^q(t_1) = \mathbf{m}_2^r(s_1)$, $\mathbf{m}_1^q(t_2) = \mathbf{m}_2^r(s_2)$, ..., $\mathbf{m}_1^q(t_i) = \mathbf{m}_2^r(s_i)$.

Because $\mathbf{m}_2^r(r) = \mathbf{m}_1^q(q)$, it follows that what \mathbf{m}_2^r assigns to each constant of ψ^*r/β is the same as what \mathbf{m}_1^q assigns to the analogous constant in the corresponding location of $\psi q/\beta$. Thus, \mathbf{m}_2^r and \mathbf{m}_1^q satisfy Dragnet condition (2).

The sentences ψ^*r/β and $\psi q/\beta$ are each of order k , so, by RA,

$$\psi^*r/\beta \text{ is true on } \mathbf{m}_2^r \text{ iff } \psi q/\beta \text{ is true on } \mathbf{m}_1^q.$$

We assumed that ψ^*r/β is true on every r -variant of \mathbf{m}_2 . So ψ^*r/β is true on \mathbf{m}_2^r . Therefore, given the above, $\psi q/\beta$ is true on \mathbf{m}_1^q .

As before, we used a single r -variant of \mathbf{m}_2 , \mathbf{m}_2^r , without our argument depends on anything specific about the object it assigns to r . So the argument above is generalizable, and it applies equally to all other r -variants. That is, for each r -variant of \mathbf{m}_2 , there is a corresponding q -variant of \mathbf{m}_1 , such that,

$$\mathbf{m}_2^r(r) = \mathbf{m}_1^q(q).$$

So, because ψ^*r/β is true on every r -variant of \mathbf{m}_2 , it follows that $\psi q/\beta$ is true on every q -variant of \mathbf{m}_1 .

■

We have established that

$$\begin{aligned} &(\psi q/\beta \text{ is true on every } q\text{-variant of } \mathbf{m}_1) \text{ iff} \\ &(\psi^*r/\beta \text{ is true on every } r\text{-variant of } \mathbf{m}_2), \end{aligned}$$

...so, by the definition of truth for \forall , it follows that

$$\forall \beta \psi \text{ is true on } \mathbf{m}_1 \text{ iff } \forall \beta \psi^* \text{ is true on } \mathbf{m}_2.$$

Existential Quantification: We leave this case for the reader to do as an exercise.

Closure Step: We have now exhausted the ways to construct a sentence of order $k + 1$. There is no other way to construct a sentence of QL, and so we have shown that the Dragnet theorem holds for all sentences ϕ . ■

Dragnet is particularly useful when making arguments about sentences with unknown structures. To illustrate, let's look at the following example.

Example 4.10. Let ϕ be some formula whose only free variable is x . Demonstrate the following for all such substitutions for ϕ : $\forall x\phi \models \phi b/x$.

Proof: Suppose $\forall x\phi$ is true on \mathbf{m} . Thus, every t -variant of \mathbf{m} makes $\phi t/x$ true, where t is the first constant not in ϕ . Consider the t -variant, \mathbf{m}^t , that makes the same assignment to t that \mathbf{m} assigns to b . Because all t -variants make $\phi t/x$ true, it follows that \mathbf{m}^t does too.

The sentences $\phi b/x$ and $\phi t/x$ are the same, except that the latter substitutes the constant t for b . So Dragnet condition (1) is met. The models \mathbf{m} and \mathbf{m}^t make all the same assignments, except that \mathbf{m}^t assigns to t what \mathbf{m} assigns to b . So Dragnet condition (2) is met.

Thus, by Dragnet:

$$\phi b/x \text{ is true on } \mathbf{m} \text{ iff } \phi t/x \text{ is true on } \mathbf{m}^t.$$

And because $\phi t/x$ is true on \mathbf{m}^t , it follows that \mathbf{m} makes $\phi b/x$ true. Therefore, the entailment holds, regardless of the internal structure of ϕ . ■

How do you know when the Dragnet Theorem might be useful? The general answer is that when you are trying to prove something about the truth of a sentence given some information about a closely related sentence, that is, one that differs by some constant substitution(s), then you should think about whether the Dragnet Theorem will help. In some cases you will need to select an appropriate variant of a model. To apply the theorem you need two sentences and two models that meet the two Dragnet restrictions. Practice will help.

While the Dragnet theorem is helpful, it can be a bit unwieldy. So, we prove another theorem, the 'Free Choice' theorem, which builds on Dragnet and saves us a good deal of trouble.

Theorem 4.11. The Free Choice Theorem: (i) A QL sentence of the form $\forall\alpha\phi$ is true on some model \mathbf{m} iff all s -variants of \mathbf{m} make $\phi s/\alpha$ true, where s is any constant not in ϕ ; and (ii) a QL sentence of the form $\exists\alpha\phi$ is true on some model \mathbf{m} iff some s -variant of \mathbf{m} makes $\phi s/\alpha$ true, where s is any constant not in ϕ .

Proof: (i) (\Rightarrow): Assume that a QL sentence of the form $\forall\alpha\phi$ is true on some model \mathbf{m} . By the definition of truth for \forall , $\phi t/\alpha$ is true on all t -variants of \mathbf{m} . Let s be some arbitrary constant not in ϕ . We want to show that $\phi s/\alpha$ is true on all s -variants of \mathbf{m} .

The sentences $\phi t/\alpha$ and $\phi s/\alpha$ are exactly the same, except the latter substitutes the constant s where the former has t . So this pair of sentences satisfies condition (1) of Dragnet.

For any t -variant of \mathbf{m} , there is a corresponding s -variant of \mathbf{m} that assigns to s what the t assigns to t . Let's take one such pair of variants, \mathbf{m}^t and \mathbf{m}^s , such that $\mathbf{m}^t(t) = \mathbf{m}^s(s)$. Does this pair, \mathbf{m}^t and \mathbf{m}^s , satisfy condition (2) of Dragnet? Sadly, no; they vary on assignments to two constants, s and t . We'll get around this problem by using an intermediate model.

Let \mathbf{m}^* be a 'buffer' model, which make all the same assignments as \mathbf{m}^t , except that what \mathbf{m}^t assigns to t , \mathbf{m}^* assigns to s . Thus, \mathbf{m}^* and \mathbf{m}^t satisfy condition (2) of Dragnet. By Dragnet,

$$\phi t/\alpha \text{ is true on } \mathbf{m}^t \text{ iff } \phi t/\alpha \text{ is true on } \mathbf{m}^*.$$

So \mathbf{m}^* makes $\phi t/\alpha$ true. \mathbf{m}^* differs from \mathbf{m}^s on only one constant assignment: what \mathbf{m}^s assigns to s , \mathbf{m}^* assigns to t . Thus, \mathbf{m}^* and \mathbf{m}^s satisfy condition (2) of Dragnet. By Dragnet,

$$\phi t/\alpha \text{ is true on } \mathbf{m}^* \text{ iff } \phi s/\alpha \text{ is true on } \mathbf{m}^s.$$

So \mathbf{m}^s makes $\phi s/\alpha$ true.

This argument holds not just for this pair of variants, \mathbf{m}^t and \mathbf{m}^s . All t -variants of \mathbf{m} make $\phi t/\alpha$ true, and using analogous reasoning, we see that all s -variants of \mathbf{m} make $\phi s/\alpha$ true.

We assumed nothing special about s , except that it's a constant not in ϕ . Therefore, all s -variants of \mathbf{m} make $\phi s/\alpha$ true, where s is any constant not in ϕ .

(\Leftarrow): Assume that all s -variants of some model \mathbf{m} make $\phi s/\alpha$ true, where s is any constant not in ϕ . Let constant t be the first constant not in ϕ . It follows that all t -variants of \mathbf{m} make $\phi t/\alpha$ true. Therefore, by the definition of truth for \forall , $\forall \alpha \phi$ is true on \mathbf{m} .

(ii) We leave the case with the existential quantifier as an exercise for the reader.

■

Let's do an example proof that uses the Free Choice theorem.

Example 4.12. Let ϕ and ψ be formulas whose only free variable is x . Prove that $\forall x(\phi \rightarrow \psi) \models \forall x\phi \rightarrow \forall x\psi$.

Proof: Let's assume, for reductio, that \mathbf{m} is a counterexample to the entailment. So, \mathbf{m} makes $\forall x(\phi \rightarrow \psi)$ true and $\forall x\phi \rightarrow \forall x\psi$ false.

Because \mathbf{m} makes $\forall x\phi \rightarrow \forall x\psi$ false, it makes $\forall x\phi$ true and $\forall x\psi$ false (definition of truth, \rightarrow). It follows that all t -variants of \mathbf{m} make $\phi t/x$ true, where t is the *some* constant not in ϕ (Free Choice theorem). In fact, let's stipulate that t occurs nowhere

in the entire entailment claim, in neither ϕ nor ψ .¹⁴ And because \mathbf{m} makes $\forall x\psi$ false, there is some t -variant of \mathbf{m} makes $\psi t/x$ false (Free Choice).

The model \mathbf{m} makes the LHS, $\forall x(\phi \rightarrow \psi)$, true. Hence, all t -variants of \mathbf{m} make $(\phi t/x \rightarrow \psi t/x)$ true (Free Choice). Because $\phi t/x$ is true on all t -variants of \mathbf{m} , it follows that $\psi t/x$ is true on all t -variants of \mathbf{m} (definition of truth, \rightarrow). But we'd concluded that some t -variant of \mathbf{m} makes $\psi t/x$ false. This is contradictory.

Therefore, our initial assumption—that some model is a counterexample to the entailment—is false. No such model exists. The entailment holds. ■

4.4 Exercises

4.4.1 Formulas, Order, and Subformulas

Which of the following are *formulas*? For those which are formulas, what is their order? How many subformulas does each have?

- | | |
|--|--|
| 1. $\forall x(H'x \rightarrow G'x)$ | 9. $\forall x\forall z(H'x \rightarrow G''xy)$ |
| 2. $\forall x(H'x \rightarrow G''x)$ | 10. $\forall x_9\forall z(H'x_9 \rightarrow G''xy)$ |
| 3. $\forall x(H'x \rightarrow G'_7x)$ | 11. $\sim\forall x\forall y(H'x \rightarrow G''xy)$ |
| 4. $\forall x\forall z(H'x \rightarrow G''xy)$ | 12. $\forall b\forall y(H'b \rightarrow G''xy)$ |
| 5. $\forall x\forall y(H'x \rightarrow G''xy)$ | 13. $\forall x(\forall xH'x \rightarrow G''xy)$ |
| 6. $\forall x\forall z(H'a \rightarrow G''xy)$ | 14. $\forall x(\forall zH'x \rightarrow G''xy)$ |
| 7. $\forall x\forall z(Hx \rightarrow G''xy)$ | 15. $\forall y(\forall xH'x \rightarrow \forall xG''xy)$ |
| 8. $\forall x\forall z(W'x \rightarrow G''xy)$ | |

4.4.2 Sentences and Order

For each of the following, say whether it is an official sentence, an unofficial sentence, an official formula but not a sentence, an unofficial formula but not a sentence or none of the above. If it is a formula or sentence (official or unofficial) say what its order is and how many subformulas it has.

- | | |
|--|--|
| 1. $\forall x\forall y(H'x \rightarrow G''xy)$ | 4. $\forall x\forall y(Hx \rightarrow Gxy \rightarrow Ky)$ |
| 2. $\forall x\exists z(H'x \rightarrow G''xy)$ | 5. $\forall x\forall y(\sim Hx \wedge Gzy \wedge Ky)$ |
| 3. $\forall x\forall y(Hx \wedge Gxy \wedge Ky)$ | 6. $\forall x\forall y(Hx \wedge (Gzy \wedge Ky))$ |

¹⁴The Free Choice theorem allows us to make this stipulation, and it makes the proof considerably easier.

7. $\forall x \forall y (H'x \wedge (G''zy \wedge K''y))$

9. $\forall x \exists z \forall y (Hx \wedge Gxy \wedge Ky)$

8. $\forall x \forall y Hx \wedge (Gzy \wedge Ky)$

10. $\forall x \exists y \forall z (Hx \wedge \forall y Gxy \wedge Ky)$

4.4.3 Truth in a Model

Give the truth value of each of the following sentences on both of the models found in table 4.2 (on the current page).

1. $\forall x \forall y ((Ax \wedge By) \rightarrow Axy)$

9. $\exists y \exists x (Cxy \wedge Dxy)$

2. $\forall x \forall y ((Cx \wedge Dy) \rightarrow Dxy)$

10. $\exists y \exists x (Exy \wedge Gxy)$

3. $\forall x ((Cx \wedge Ex) \rightarrow Cxa)$

11. $\forall x (Gx \rightarrow \forall w (Axw \rightarrow (Aw \vee Cw)))$

4. $\forall x \forall y (Cxy \rightarrow Axy)$

12. $\exists x (Cx \rightarrow \forall y Cy)$

5. $\forall x Cx \rightarrow \forall y Dy$

13. $\forall z \forall w \forall x (Dxzw \rightarrow Axw)$

6. $\forall z \forall w ((Gz \wedge Gw) \rightarrow \sim Gzw)$

14. $\forall y (Ay \rightarrow \exists x (Ex \wedge Axy))$

7. $\forall z \forall w \forall x (Cxyz \rightarrow (Cx \leftrightarrow Cw))$

15. $\forall x \forall y (Gxy \rightarrow Gyx)$

Symbol		Model	
		Pos Int	States
Universe:		The set of positive integers	The set of US states (2015)
Sent. Let.:	A	true	false
	B	true	false
	C	false	true
	D	true	false
	E	true	false
	G	false	true
Constants:	a	1	Louisiana
	b	9	Maine
	c	72	Georgia
	d	3	Nebraska
	e	1	New Mexico
	f	2	Texas
1-place:	A'	all pos int	Midwestern
	B'	empty set	name with > 5 letters

Table 4.2: Example Models
Continued next Page

Continued from Previous Page

	Symbol	Model	
		Pos Int	States
2-place:	C'	even	Coastal
	D'	odd	one of original 13
	E'	prime	{Ohio}
	G'	multiple of 7	{Ohio, Alabama}
	A''	first > second	share a border
	B''	are equal	first is north of second
	C''	first = 2 times second	first > second (area)
3-place:	D''	sum of them equals 7	first > second (population)
	E''	first < second	first is west of second
	G''	are relatively prime	both coastal, or neither
	A'''	all equal	all same population
	B'''	first < second < third	first is north of others
	C'''	all odd or all even	first > second > third (area)
	D'''	first + second = third	first + second > third (area)
	E'''	first \times second = third	first is west of the others
	G'''	are all relatively prime	at least two coastal

Table 4.2: Example Models

4.4.4 Quantificational Truth Problems

For each sentence below, say whether or not it's a quantificational truth. If so, prove it. If not, show it by giving a model \mathbf{m} that makes it false.

1. $\forall x \forall y Hxy \rightarrow \forall y \forall x Hxy$
2. $\exists x \exists y Hxy \rightarrow \exists y \exists x Hxy$
3. $\forall x \exists y Hxy \rightarrow \exists y \forall x Hxy$
4. $\exists y \forall x Hxy \rightarrow \forall x \exists y Hxy$
5. $\forall x (Ax \rightarrow \exists y (Hxy \wedge By))$
6. $\exists y (Ay \wedge \forall z (Bz \rightarrow Hyz))$
7. $\forall x (\exists y Hxy \wedge \sim Hxx \wedge \forall y \forall z ((Hxy \wedge Hyz) \rightarrow Hxz))$

4.4.5 Preliminary Dragnet Practice Problems

1. If ϕ is $\forall z_3 (Gz_1 z_3 \wedge \exists x Bx z_3 y_2) \rightarrow (A \rightarrow \forall y_2 Dz_3 y_2)$, what is
 - (a) $\phi a / z_3$
 - (b) $\phi a / y_2$
 - (c) $\phi z_3 / x$
 - (d) $\phi a / x$

2. If $\phi x/y$ is Dxx , can you determine what ϕ is? If so, what is it? If not, why not?

4.4.6 Dragnet Practice

(i) For each of the following statements, work out whether it is true or false. (ii) For each statement, determine whether Dragnet would be required to prove it if it is true. (iii) For which ones is Dragnet required to disprove it if it is false? Note that each of θ , ψ , and ϕ should be understood as standing for complex formulas of QL whose only free variables are x , and which do not contain the variables y , z or w .

1. $\forall w(\phi w/x \wedge \psi w/x) \models \forall x\phi \rightarrow \forall x\psi$
2. $\forall x\phi \rightarrow \forall x\psi \models \forall y\phi y/x \rightarrow \forall z\psi z/x$
3. $\forall x\phi \rightarrow \forall x\psi \models \exists x(\phi \wedge \psi)$
4. $\forall x(\phi \rightarrow \psi) \models \forall y\phi y/x \rightarrow \forall z\psi z/x$
5. $\forall y\phi y/x \rightarrow \forall z\psi z/x \models \forall x(\phi \rightarrow \psi)$
6. $\forall y\phi y/x \rightarrow \forall z\psi z/x, \exists x(\phi \wedge \psi) \models \forall x(\phi \rightarrow \psi)$
7. If $\models \forall x(\phi \rightarrow \psi)$, then $\models \forall x((\phi \wedge \theta) \rightarrow (\psi \wedge \theta))$
8. $\exists y(\phi y/x \wedge \sim \psi y/x) \models \forall y(\phi y/x \rightarrow \sim \psi y/x)$
9. $\sim \forall x\phi \models \exists x\sim \phi$
10. $\models \forall x(\phi \rightarrow \psi) \vee \forall y(\phi y/x \rightarrow \sim \psi y/x)$

Chapter 5

Translations

5.1 SL Applications

Sentences in English have meanings. Declarative sentences say something about the world and in virtue of that they have truth conditions: ways the world must be for them to be true. By contrast, the sentences of SL (and QL) are, considered in isolation, meaningless strings of symbols. To the extent that we can ascribe meaning to SL (and QL) sentences at all, that meaning comes from a model. The way that we specify models in Chapter 2 does not distinguish between different sentences with the same truth value, because for the purposes of assessing logical truth in SL, truth value is all that matters. But if we want to translate English into SL we must specify the models less directly: by associating English sentences with the sentence letters. If we are careful in our selection of English sentences we may assign each of them unambiguous truth values. We then take these values as the assignments that some model \mathbf{m} makes to the corresponding sentence letters. We assign truth values to the sentences according to what we know about reality, in which case \mathbf{m} would be a ‘model’ of reality, or at least that part of reality described by those English sentences. Alternatively, we may assign truth values to English sentences so that they do not match what we know about reality, in which case \mathbf{m} would be a model of a non-actual state of affairs. Finally, we could consider sentences whose values we don’t know and assign them values, in which case \mathbf{m} would be a model of a hypothetical state of affairs.

Ideally, the English sentences in question should be unambiguous, not vague, and will not have a truth value that changes over time. More practically, if we use sentences that are unambiguous in the appropriate context, specific enough to have determinate truth values, and whose truth values will not change during the discussion, we can analyze them with the tools of logic without going astray. One source of sentences with these characteristics is mathematics. For other sentences, we can usually add enough detail about context to meet our criteria. For example, we might casually say that ‘Texas is a US state’ is a true sentence of English. The context would normally indicate that we mean Texas is a US state at the time the sentence is uttered. If we consider the sentence apart from some context of utterance matters aren’t so clear. The sentence was false in 1844 and is true now, so its truth value has varied over time. Officially we prefer a less ambiguous sentence, such as ‘Texas was a US state

at noon CDT on May 1, 1983.’ Unofficially, we often make background assumptions about what is relevant, who is being discussed, the time of assessment, and so on. For similar reasons, ‘Washington was elected President’ has an obvious and natural interpretation, one that’s distinct from ‘Joan Washington was elected President of the Montrose Dog Walkers Association.’

There is a controversy in philosophy of logic and philosophy of language over which objects are the most fundamental bearers of truth value: sentence types, sentence tokens, propositions, judgments, or statements. We regard sentences, whether types or tokens, as better objects to work with than the alternatives. We have a better understanding of what a sentence is than what propositions, judgments, or statements are. That is why in Chapter 2 we use the name ‘Sentential Logic’ rather than ‘Propositional Logic’, which is used by some logic texts. For our purposes we can be agnostic as to whether it’s sentence types or tokens because, given our restrictions, all tokens of a type will have the same truth value in our specific translation context.

We mentioned associating English sentences with SL sentence letters. To do this, we give a *translation key* that maps sentence letters from SL to sentences of English. A translation key is different from a model, but a good translation key will help fix a specific model for the relevant sentences of SL. We just need to supply the truth values of the English sentences of the key. Once we’ve mapped sentences of English to sentence letters we can use the key to translate more complex English sentences into SL. We may think of complex sentences of English as being composed of (i) simpler English sentences and (ii) logical connectives of English.

But how do we know when a sentence of English is sufficiently simple to be assigned to a sentence letter? The answer is that in putting together a key we should capture as much of the relevant logical structure of the English as is practical and useful. All else being equal, translation keys that hide important logical structure in a sentence letter are deficient. For example, the following sentence is not ordinarily a good candidate for sentence letter assignment:

1. Mares eat oats and does eat oats and little lambs eat ivy.

The word ‘and’ plays a truth-functional role in this sentence, analogous to the role that ‘ \wedge ’ plays in SL. A better translation key would assign each of ‘Mares eat oats’, ‘Does eat oats’, and ‘Little lambs eat ivy’ its own sentence letter, so that the original sentence of English could be expressed as a conjunction in SL.

More generally, we want to translate the truth-functional connectives of English sentences with appropriate logical connectives from SL. But before considering other examples, we need to say a little more about connectives. In Chapter 2 (in definition 2.1, on page 15) we simply listed what we called the “logical connectives” of SL. We’ve seen the role they play in building the sentences of SL and in fixing the truth value of those sentences in a model. But we haven’t said anything explicit about what connectives are. The basic idea of a sentential *connective*—whether a “logical” one in some formal language or an expression in a natural language like English—is a bit

of a language that can be used to combine, or connect, one or more sentences of the language into a new sentence.¹ We say that one connective is within the *scope* of a second connective iff the first connective is within a sentence directly connected with the second connective. The *main connective* is the one connective of the sentence that's not within the scope of any other connectives.

A connective is *truth-functional* iff the truth value of every new sentence formed by that connective depends solely on the truth value of the constituent sentences. (If we're talking about a connective in a formal language like SL, we mean the truth value of a sentence *in a model*.) Not all connectives of English are truth-functional. In particular, modal connectives—such as ‘necessarily’ and ‘possibly’, among others—defy truth-functional characterization. For example, the sentence

2. Necessarily, $2 + 2 = 4$.

seems plausibly true. If we replace ‘ $2 + 2 = 4$ ’ with another true claim, however, the result may not be so plausible:

3. Necessarily, there is life on earth.

Both ‘ $2 + 2 = 4$ ’ and ‘There is life on earth’ are true. But it is easier to imagine all life on earth perishing than for $2 + 2$ not to be 4. That there is life on earth is clearly contingent. These examples make clear that the truth of a sentence with a modal main connective depends on more than the truth value of the part of the sentence governed by the modal connective.

On the other hand, our definition of truth in a model (def. 2.21, on page 23) makes it apparent that the five logical connectives of SL are all truth-functional. This disparity means that SL is not capable of capturing all of the logical structure of English; the best it can do is to approximate certain features of English. Some formal languages have the resources to capture the logical structure of modal claims. We consider one such language in Chapter 8. We hasten to add, however, that SL has sufficient resources to render many English sentences without significantly distorting their basic meaning.

5.1.1 Connectives and Translations

A logical connective of SL is a suitable translation of a (truth-functional) connective of English iff they express the same truth function (see section 2.2.2, on page 25)—that is, iff the new sentences formed by those connectives share the same truth value whenever the constituent sentences that made them up share the same truth values. We've already mentioned one such pair of corresponding connectives: ‘and’ and ‘ \wedge ’. We know that the English sentence

4. The sun is shining *and* the birds are chirping.

¹Other connectives play a subtler role, as with the quantifiers of QL. In QL quantifiers can be used to combine predicates to make a more complex predicate.

is true iff the following two sentences are also true:

5. The sun is shining.
6. The birds are chirping.

The same holds of other conjunctions in English. The SL connective ‘ \wedge ’ is a good translation of ‘and’ most of the time. Sometimes the structure of English conjunctions is more hidden, as in the following example:

7. Nathanael Green *and* ‘Light Horse Harry’ Lee were officers in the Continental Army.

We see here the word ‘and’, but a translation key should not break the sentence at that word and then give the LHS—‘Nathanael Green’—its own sentence letter. The LHS is not a declarative sentence. It’s instead clear that (7) expresses the conjunction of the following two sentences:

8. Nathanael Green was an officer in the Continental Army.
9. ‘Light Horse Harry’ Lee was an officer in the Continental Army.

It is thus appropriate to map each of these latter sentences to a sentence letter—say, N and L , respectively—and express ‘Nathanael Green and ‘Light Horse Harry’ Lee were officers in the Continental Army’ as a conjunction of these sentence letters—e.g., $(N \wedge L)$. Other words of English also play the role of conjunction. Consider the sentence:

10. Mary wants to drive her mother’s car, *but* she is *not* old enough.

Here the word ‘but’ connects two independent claims. The sentence as a whole is true iff the LHS and the RHS are each true. So ‘but’ also plays the same truth-functional role as ‘ \wedge ’ in SL. The word ‘not’ is another connective. As you can probably guess, it corresponds to the ‘ \sim ’ of SL.

Negations are not always this obvious. The words “not” and “no” are easy to translate with \sim . But the connective also appears in many other forms in English. “Innumerable” is not numerable, “unmistakable” is not mistakable, “never” is not ever, “immoral” is not moral, “clueless” is to have no clue, and so on. One of the things that makes translating complex is that while the prefix “in-” is often negation, as in “inevitable”, it is also often a direction, as in “influx”. The latter case is not one of negation.

Notice that for the RHS of 10, context indicates that ‘she’ is Mary, and that mention of her being not ‘old enough’ is about the fact that she is not old enough *to drive her mother’s car*. English is, in this respect, more efficient than SL. So, to translate this sentence into SL we’ll need to use something like the following:

Translation Key:

A: Mary wants to drive her mother's car.

O: Mary is old enough to drive her mother's car.

We took out the 'not' because we can represent that with the ' \sim '. So now we may render the original sentence in SL as $(A \wedge \sim O)$. This is indistinguishable from the result that we would give if we were instead translating

11. Mary wants to drive her mother's car *and* she is not old enough to drive her mother's car.

We typically say ' Φ but $\sim\Theta$ ' when we think ' Φ and $\sim\Theta$ ' is true and we think that our listeners will expect ' Θ ' when they hear ' Φ '. Often the latter conjunct in a 'but' sentence has an implicit or explicit negation, as in sentence (10); though occasionally the negation may be on the left, as in 'John didn't study, but he passed the exam'. In other cases, the latter conjunct has no negation but draws a contrast or expresses something that the speaker thinks will defy the expectations of the listeners, given the earlier conjunct(s). This difference from the word 'and' cannot be captured in SL—it will be lost in translation. Remember that these SL translations are approximations of English, and as such they can't always preserve all of the original meaning.

The word 'and' can do more than serve as a conjunction. Sometimes the order of the conjuncts makes a difference in how we interpret meaning. Compare the following two sentences:

12. John took off his clothes and went to bed.
13. John went to bed and took off his clothes.

The order of the conjuncts suggests that John performed these actions in a different order. Yet if we were to translate (12) and (13) into SL, the resulting sentences will be truth-functionally equivalent. We think this equivalence is defensible, because (12) and (13) don't literally describe the order of events; we tend to read order into them because it's usual in conversation to recount events in chronological order. Notice that there is nothing contradictory about the following sentence:

14. John went to bed and took off his clothes, but I don't know in which order.

Another connective of English is the word 'or':

15. *Either* the tigers will get us *or* the lions will.

Here the word 'or' connects two claims. The sentence as a whole is true iff a claim on either side of the 'or' is true. This is analogous to the truth-function of the ' \vee ' connective in SL, and so that's how we'll translate it.

English	SL
Φ and Θ	$\phi \wedge \theta$
both Φ and Θ	$\phi \wedge \theta$
Φ , but Θ	$\phi \wedge \theta$
Φ , but not Θ	$\phi \wedge \sim\theta$
not Φ , but Θ	$\sim\phi \wedge \theta$

Table 5.1: Translations for Common English Conjunctions

Translation Key:

I : The tigers will get us.

L : The lions will get us.

The resulting translation: $(I \vee L)$. We again use context to fill out the sentence ‘the lions will [get us]’.

It might not seem quite right to translate ‘or’ as ‘ \vee ’. The potential problem is that in SL $(\phi \vee \theta)$ is true (on a model) when both ϕ and θ are true (on that model). But sometimes in English when we say ‘ Φ or Θ ’ we mean that one or the other of Φ and Θ is true—but not both. We say that a disjunction which allows for both disjuncts to be true is *inclusive*, while one that doesn’t allow for both disjuncts to be true is *exclusive*. Does this mean that ‘ Φ or Θ ’ is exclusive, and hence shouldn’t be translated by the inclusive $(\phi \vee \theta)$?

This is a complicated question that we cannot fully answer here. We believe, however, that ‘or’ is inclusive in English, and that it is best translated as ‘ \vee ’. Only in certain conversational contexts is the exclusive disjunction clearly intended. We will motivate our stance with a few brief considerations.²

One case in which ‘or’ sounds exclusive involves disjuncts that cannot possibly both be true. For example, I might say “Mia took the highway or a plane”. If I’m referring to two possible routes for the same journey, clearly it’s just not physically possible for Mia to have taken both the highway and a plane. But it doesn’t follow from the fact that it’s impossible for both disjuncts to be true that the utterance of the sentence itself says, or has the meaning that, both disjuncts cannot be true. For cases in which it’s not possible for both disjuncts to be true it’s tempting to assimilate that fact into the meaning itself. But that leap is unjustified. Someone could, after all, understand “Mia took the highway or a plane” and agree that it’s true, but also think that both disjuncts are true. Perhaps Mia’s journey was more disjointed and

²This discussion borrows from Smith 2012: 117–22.

English	SL
Φ or Θ	$\phi \vee \theta$
either Φ or Θ	$\phi \vee \theta$
Φ or Θ , but not both	$(\phi \vee \theta) \wedge \sim(\phi \wedge \theta)$

Table 5.2: Translations for Common English Disjunctions

involved both. To infer that this disjunction is exclusive requires more information than the sentence itself conveys.

There are two kinds of cases in which it's more plausible that 'or' is exclusive. The first case involves commands or rules. If you are at a fancy restaurant and a waiter says, "You may have the soup or a salad", it's typically understood that you may have one or the other, but not both. The second case involves elliptical clauses. A disjunctive phrase ' Φ or Θ ' can be elliptical for the longer ' Φ or Θ , but not both'. If so, then the intended message is exclusive. But this doesn't count as a case in which 'or' by itself expresses exclusive disjunction. It's a case in which 'or' plus the modifying phrase 'but not both' together express exclusive disjunction—it's just that the latter phrase 'but not both' is tacit. Special cases like these, which are rare but salient, might lead you to think that disjunctions in English are sometimes exclusive. However, in these cases contextual clues give such disjunctions their exclusive character, independently of the literal meaning of the relevant sentences. We know as a matter of cultural familiarity that restaurants don't offer *both* soup *and* salad unless we pay extra. When intergalactic visitors discover Earth and visit our restaurants, we recommend that waiters make the exclusivity of the disjunction explicit. (Otherwise there could be an interstellar incident!) They can even use SL to do so:

16. The visiting intergalactic visitor may have the soup *or* a salad, *but not* both.

This sentence has several truth-functional connectives, so we must be careful to understand which connectives govern which. We recognize 'or', 'but', and 'not'. In this case the 'but' governs everything else, so the whole sentence is a conjunction. The visitor may have the soup or a salad, *and* he may not have both the soup and a salad. (As we fill out the rest of the compressed meaning of the sentence we find another connective.) On to the translation:

Translation Key:

P: The visiting intergalactic visitor may have the soup.

D: The visiting intergalactic visitor may have a salad.

English	SL
if Φ , then Θ	$\phi \rightarrow \theta$
Φ only if Θ	$\phi \rightarrow \theta$
Φ if Θ	$\theta \rightarrow \phi$
Φ provided that Θ	$\theta \rightarrow \phi$
provided Φ , Θ	$\phi \rightarrow \theta$
Φ assuming that Θ	$\theta \rightarrow \phi$
assuming Φ , Θ	$\phi \rightarrow \theta$
for Φ , it's necessary that Θ	$\phi \rightarrow \theta$
for Φ , it's sufficient that Θ	$\theta \rightarrow \phi$

Table 5.3: Translations for Common English Conditionals

First we translate the LHS of the ‘but’: $(P \vee D)$. If we translate the RHS without the ‘not’—i.e., ‘the intergalactic visitor may have both soup and a salad’—we get: $(P \wedge D)$. We negate the result to account for the ‘not’: $\sim(P \wedge D)$. Putting it all together, the resulting translation is: $((P \vee D) \wedge \sim(P \wedge D))$.

Another connective of English is the *conditional*. Conditionals are expressible in many ways, but perhaps the most distinctive way is ‘if ... then ...’ For example,

17. If George Washington crosses the Delaware River then the Hessians will be defeated.

Is the conditional of English a truth-functional connective? This is a hotly debated topic that has yet to be resolved. SL has only truth-functional connectives, so if the conditional is not truth-functional then we cannot fully capture the meaning in SL. For our purposes, we can show that \rightarrow is the best model of the conditional and as we discuss translations we can see how well the model fits English.

How well does the ‘ \rightarrow ’ capture the meaning of the English language conditional? The only way for an SL sentence of the form $(\phi \rightarrow \theta)$ to be false on a model \mathbf{m} is for \mathbf{m} to make ϕ true and θ false. Clearly such an assignment of truth values makes corresponding English language conditionals false: “If $2 + 2 = 4$ then the US State Department is secretly controlled by super-intelligent hamsters.” So far, so good. On all other combinations of truth values for ϕ and θ , SL sentences of the form $(\phi \rightarrow \theta)$ are true. Consider cases in which both the LHS and the RHS of a conditional are false. Some such conditionals are true, for example: “If George Washington landed on

the moon then George Washington landed on the moon.” Indeed, this sentence seems to be a logical truth. Others are more dubious: “If Soviet cosmonauts landed on the moon in 1968, then a cabal of super-intelligent hamsters will seize control of the US federal government later this year.” We cannot distinguish these connectives purely by reference to the truth values of each side; in both cases, the LHS and the RHS are each false. In order to admit the former as true, we must also admit the latter. This is one price of understanding conditionals as truth-functional.

Next we address the case in which an SL model makes ϕ false and θ true, and hence also makes $(\phi \rightarrow \theta)$ true. But first consider the following, in which each side is true:

18. If George Washington crossed the Delaware River then George Washington crossed the Delaware River.

This sentence is a logical truth of English. But if we were to turn the LHS into a conjunction and insert a false conjunct, we’ll still think the result is true. E.g.,

19. If George Washington crossed the Delaware River and Thomas Jefferson invented bifocals, then George Washington crossed the Delaware River.

(The true inventor of bifocals was probably Benjamin Franklin.) Even though the conjunction on the LHS is false, the whole conditional is still a logical truth. And, indeed, the SL translation is TFT: $((G \wedge B) \rightarrow G)$. So, while the ‘ \rightarrow ’ is not a perfect analogue of the English language conditional, it seems to be the best truth-functional translation.

Biconditionals in English are closely related to conditionals, and so they’re subject to some of the same worries. Nevertheless, we can treat them as truth-functional connectives for the purpose of translating them into SL:

20. Ruth may play outside if and only if she cleans her room.

Translation Key:

H : Ruth may play outside.

C : Ruth cleans her room.

With this key we may translate the sentence as: $(H \leftrightarrow C)$. We use the ‘ \leftrightarrow ’ to translate English biconditionals.

We saw above that the restaurant context may add exclusivity to an ‘or’, which by itself is inclusive. Similarly, an ‘if’ in a particular context may be interpreted as an ‘iff’ because of background context. As in other cases, this context adds more information but is not itself part of the sentence. A parent who says, ‘You can have ice cream if you finish your homework,’ is relying on context to add ‘but not otherwise’. Otherwise, it might turn out that the child can have ice cream either way. We know that parents

English	SL
Φ if and only if Θ	$\phi \leftrightarrow \theta$
for Φ it's necessary and sufficient that Θ	$\phi \leftrightarrow \theta$
Φ just when Θ	$\phi \leftrightarrow \theta$
Φ just in case Θ	$\phi \leftrightarrow \theta$

Table 5.4: Translations for Common English Biconditionals

often use punishments and rewards to shape child behavior, so it is appropriate to interpret them accordingly. Nevertheless, our policy is to translate what the sentence says literally. If it is important for an argument to articulate what is being added by context, that should be specified explicitly.

Let's look at a few more example translations.

Example 5.1.

21. Either Mia flew or both Jackson and Harper took off early.

The first step is to identify the main connective of the sentence. In this case it's 'either ... or'. When we translate 'either ... or' as \vee , we get:

22. (Mia flew) \vee (both Jackson and Harper took off early).

Notice that the word "both" clarifies the logical structure of the sentence. Without it, the sentence is, "Either Mia flew or Jackson and Harper took off early." This is ambiguous, and could be read as saying that either Mia or Jackson flew.

Next we identify and translate the main connectives in the connecting clauses. There are no connectives in 'Mia flew', so there is nothing to do with it. There is one connective in 'both Jackson and Harper took off early', 'both ... and'. So we finish by translating this connective as a conjunction (\wedge):

23. (Mia flew) \vee ((Jackson took off early) \wedge (Harper took off early)).

Translation Key:

N : Mia flew.

J : Jackson took off early.

H : Harper took off early.

24. $N \vee (J \wedge H)$

Example 5.2.

25. If Mia got the job and Jackson didn't, then Mia will take off tomorrow and Harper will have to come in.

As before we identify the main connective, which is 'if ... then'. This is translated as a conditional (\rightarrow), yielding:

26. (Mia got the job and Jackson didn't) \rightarrow (Mia will take off tomorrow and Harper will have to come in).

Next we look at the LHS of the conditional, 'Mia got the job and Jackson didn't'. The main (and only) connective of this sentence is 'and', which we translate as a conjunction (\wedge), yielding:

27. ((Mia got the job) \wedge (Jackson didn't get the job)) \rightarrow (Mia will take off tomorrow and Harper will have to come in).

Note that the right-hand conjunct 'Jackson didn't' is an elliptical clause. The sentence is saying that Mia got the job and Jackson didn't *get the job*. The word 'did' can function sort of like a pronoun, except instead of referring to some person or object it alludes to another clause. We've made this explicit in by putting the part of the clause that was left tacit in brackets. The clause 'Jackson didn't' has a negation in it. It combines 'not' with 'Jackson got the job'.

We finish the translation by working on the RHS of the sentence, which is a conjunction:

28. ((Mia got the job) \wedge (\sim (Jackson got the job))) \rightarrow ((Mia will take off tomorrow) \wedge (Harper will have to come in)).

There are no more connectives, so from here we just need a translation key.

Translation Key:

N : Mia got the job.

J : Jackson got the job.

I : Mia will take off tomorrow.

H : Harper will have to come in late.

We can then finish the translation:

29. $(N \wedge \sim J) \rightarrow (I \wedge H)$

Example 5.3. Some connectives in English are not complex, but their translations into SL are complex because we did not introduce individual symbols for all possible truth-functions. Nevertheless, we can express any truth-function using some combination of the SL connectives that we *do* have. For example:

English	SL
not Φ	$\sim\phi$
it's not the case that Φ	$\sim\phi$
Φ unless Θ	$\sim\theta \rightarrow \phi$
Φ unless Θ	$\theta \vee \phi$
unless Φ , Θ	$\sim\phi \rightarrow \theta$
Φ if not Θ	$\sim\theta \rightarrow \phi$
neither Φ nor Θ	$\sim(\phi \vee \theta)$
	$\sim\phi \wedge \sim\theta$
not both Φ and Θ	$\sim(\phi \wedge \theta)$
	$\sim\phi \vee \sim\theta$

Table 5.5: Translations for Common English Negations and Complex Connectives

30. Neither Mia nor Harper were late.

Here the main connective is ‘neither ... nor’, and it joins the two sentences ‘Mia [was late]’ and ‘Harper [was] late’. (Neither of these two sentences is made up of any other connectives.) Although ‘neither ... nor’ is not complex, it is complex relative to SL. SL has no single connective which can correctly translate ‘neither ... nor’.³ Recall theorem 2.78 (on page 52), which says that any truth-functional connective can be expressed in SL. This means there must be some way to capture ‘neither ... nor’ using a complex of connectives in SL. There are two ways to do this. We may translate ‘neither Φ nor Θ ’ as either $\sim(\phi \vee \theta)$ or $\sim\phi \wedge \sim\theta$. So we have:

31. $\sim((\text{Mia was late}) \vee (\text{Harper was late}))$.

With the key:

Translation Scheme:

N : Mia was late.

H : Harper was late.

The final translation of sentence (30) is:

³But recall the logical connective NOR, discussed at the very end of section 2.6.2 (on page 48). This connective would correctly translate ‘neither ... nor’, but isn’t part of SL.

32. $\sim(N \vee H)$

We conclude this section by emphasizing two points. First, the tables in this chapter should be thought of as rough-and-ready guides. Although many particular uses of ‘and’ express conjunction, not all do. Sometimes ‘and’ doesn’t function as a connective at all, e.g. as in ‘it will be years and years before the trees bear fruit’ (Smith 2012: 107). Other times ‘and’ functions as a connective, but expresses a conditional instead of a conjunction, e.g. ‘study hard, and you will pass the exam’ (Smith 2012: 107).

Whenever deciding on how to translate a connective from English you must carefully determine what is actually being expressed. Many English sentences are ambiguous. We usually don’t notice because the context directs our attention to one meaning rather than the other. For example, “Maria and Paul are married” can convey the conjunction “Maria is married and Paul is married”, or it can express “Maria and Paul are married to each other” in which case the ‘and’ is not a conjunction but is serving a different logical purpose to be discussed in the next section.

The second point concerns how to decide what SL connective (or complex of connectives) translates a given English connective. The idea, again, is to make sure that the chosen SL connective expresses the same truth function as the English connective—that is, that the truth of the sentences formed by these connectives depends in the same way on the truth of the (sub)sentences that were joined.

5.2 QL Applications

5.2.1 Constants and Predicates

Let’s say we want to translate the following sentence into *SL*:

1. Mary is happy, smart, adorable, and a child.

The only connective we can translate is the ‘and’. The translation key will look something like the following:

Translation Key:

H : Mary is happy.

R : Mary is smart.

A : Mary is adorable.

C : Mary is a child.

So the SL result is: $(H \wedge R \wedge A \wedge C)$. This translation might work for certain purposes, but the various conjuncts have nothing in common, as a matter of logical structure. There is nothing to indicate explicitly that each conjunct is about the same person, unless we consult the translation key.

QL gives us more precision by providing individual constants and predicates. Instead of picking out simple sentences of English, we may instead pick out subjects and predicates of simple sentences. Let's use the following QL key to translate (1):

Translation Key:

m: Mary
 Ht : t a child.
 Rt : t is smart.
 At : t is adorable.
 Ct : t is a child.

Now the result is: $(Hm \wedge Rm \wedge Am \wedge Cm)$. This translation is more complex than the SL translation, but we can see that we are predicating several things about the same person. This gives QL more power and precision than SL. Consider the following sentence:

2. Ronnie and Demaryius are athletic, but Peyton isn't.

We *could* translate this as a conjunction with three conjuncts in SL. However, this would not make clear that the same predicate either applies, or doesn't, to each of the three. Instead, let's use the following QL key:

Translation Key:

r: Ronnie
d: Demaryius
p: Peyton
 At : t is athletic.

The result: $(Ar \wedge Ad \wedge \sim Ap)$. Now we can clearly see the common predicate in each conjunct.

The translation keys here resemble the QL models we provide; for example, see table 4.1 in Chapter 4. One difference is that we don't have a domain assigned in either of the QL keys above. However, we could add a domain to each of the above, and understand the constant and predicate lines of the key as making assignments from the domain. Hence, we can effectively treat model assignments as also playing a translation key role.

5.2.2 Quantifiers

We mentioned in Chapter 3 that the quantifier ' \forall ' corresponds to the English phrases 'all' or 'every'. Let's say we want to claim that every member of some class is also a member of some other class. For example:

3. All dogs are furry.

We can imagine that we have two sets: the set of all dogs and the set of all furry things. Sentence (3) effectively claims that the set of dogs is a subset of the set of furry things. How do we translate this into QL? We see the word ‘all’, so we’ll want to use a universal quantifier. This isn’t obvious from the sentence itself, but whenever we want to make claims such that one set is a subset of another, we’ll nearly always want to translate it into the following form: $\forall x(\phi \rightarrow \theta)$; i.e., with a ‘ \forall ’ governing an ‘ \rightarrow ’. This makes more sense if we paraphrase (3) in MathEnglish as: ‘For all x , if x is a dog then x is furry.’ For the rest of the translations in this section, let’s use the following model as our translation key:

Animals model:

Animals(U): All animals.
Animals(A'): is a mammal.
Animals(C'): is a cat.
Animals(D'): is a dog.
Animals(E'): is energetic.
Animals(H'): is a happy.
Animals(R'): is furry.
Animals(A''): is smarter than.

So we may translate (3) as: $\forall x(Dx \rightarrow Rx)$. We may translate other English words using the universal quantifier as well. The following uses the word ‘no’ to make a universal claim.

4. No dogs are furry.

Again, we can think of this as a claim about two sets. This sentence basically claims that the set of dogs and the set of furry things are disjoint, i.e., that nothing is a member of both sets. We can usually translate such claims into the following form in QL: $\forall x(\phi \rightarrow \sim \theta)$. To see this, consider the following MathEnglish paraphrase: ‘For all x , if x is a dog then x is not furry.’ The resulting translation is thus: $\forall x(Dx \rightarrow \sim Rx)$. The word ‘only’ can also be used for universal claims:

5. Only dogs are furry.

This is translated in exactly the same way as sentence (3), except that we reverse the order of the LHS and the RHS of the conditional governed by the ‘ \forall ’. It’s roughly equivalent to the claim that ‘All furry things are dogs.’ So: $\forall x(Rx \rightarrow Dx)$.

We said in Chapter 4 that ‘ \exists ’ corresponds to the English phrases ‘there exists’, ‘there is’, or ‘some’. Consider the following existential sentences.

6. Some dogs are furry.
7. Some dogs are not furry.

If we again think of the set of dogs and the set of furry things, (6) is a claim that there is at least one thing that is a member of each set. Notice that we are interpreting (6) as a claim about at least one object. In English we typically think that (6) is a claim about at least two dogs. For now we will ignore certain plural/singular distinctions in the way we interpret existential claims. For our purposes, ‘some’ will mean ‘at least one’. We will be able to handle ‘some’ in a more satisfactory way when we add to QL in Chapter 8.

For now, we will translate sentences like (6) into the form: $\exists x(\phi \wedge \theta)$. So, for (6) itself: $\exists x(Dx \wedge Rx)$. And we can account for the ‘not’ in sentence (7) as follows: $\exists x(Dx \wedge \sim Rx)$.

We can also translate sentence (4), ‘No dogs are furry’, as an existential governed by a negation. We could translate ‘There is a furry dog’ as: $\exists x(Rx \wedge Dx)$. Now we may negate the result to capture the meaning of (4): $\sim \exists x(Rx \wedge Dx)$. In fact, this latter translation is logically equivalent to the one we gave earlier: $\forall x(Dx \rightarrow \sim Rx)$. To see this, join these two sentences together with a biconditional, ‘ \leftrightarrow ’, and prove that the result is QT.

Let’s translate some more complicated sentences into QL:

8. All happy dogs are furry and energetic.

We’ll want to translate this as a universal quantifier governing a conditional, but we must be careful to translate each side of the conditional correctly. We can paraphrase (8) in MathEnglish as: For all x , if (x is happy and x is a dog) then (x is furry and x is energetic). So we can consider each side of the conditional as a conjunction: $\forall x((Hx \wedge Dx) \rightarrow (Rx \wedge Ex))$.

9. All cats and dogs are mammals.

This sentence is also going to be translated as a universal quantifier governing a conditional, but the word ‘and’ can be tricky. Here ‘and’ seems to conjoin predicates, not sentences. We may be tempted to translate (9) as: $\forall x((Cx \wedge Dx) \rightarrow Ax)$. But this QL sentence can be translated into MathEnglish as: ‘For every x , if (x is a cat and x is a dog) then x is a mammal.’ But that’s silly. Unless mad scientists are involved, nothing is both a cat and a dog. Instead, we should translate the ‘and’ in (9) as a ‘ \vee ’: $\forall x((Cx \vee Dx) \rightarrow Ax)$. Let’s translate this QL sentence into MathEnglish: ‘For all x , if (x is a cat or x is a dog), then x is a mammal.’ Take a moment to see how this better expresses the meaning of (9).

Now let’s look at a sentence with multiple quantifiers:

10. All dogs are smarter than all cats.

We see the words ‘all’ twice in this sentence, so we’ll want to include two universal quantifiers in the translation. Now consider a paraphrase into MathEnglish: ‘For every x , if x is a dog then (for all y , if y is a cat then x is smarter than y).’ So we may translate (10) as: $\forall x(Dx \rightarrow \forall y(Cy \rightarrow Axy))$.

We have not said much about the role of domains in translations. Because the main point of translations, other than the sheer joy of doing it, is to evaluate arguments, it is appropriate to choose a domain suitable for the arguments in question. Often a suitable choice of domain simplifies the translations. If we are translating arguments that involve essential mention of dogs and natural numbers, we need to include both in the domain and to have a predicate for each. If the arguments deal only with dogs, then we can take the domain to be dogs and we don’t need a predicate for ‘dog’.

As we observed earlier many English sentences are ambiguous. One systematic ambiguity is in sentences of the form “All As are not Bs”, which can either mean that it is not true that All As are Bs, or that all As are not-Bs. Context or content (inclusively) usually indicate what is meant: “All sheep are not good pets” would usually have the first meaning: $\sim \forall x(Sx \rightarrow (Gx \wedge Px))$; while “All sharks are not good pets” would have the second: $\forall x(Sx \rightarrow \sim(Gx \wedge Px))$. One of the values of formalization is that we can clearly and unambiguously express the structure of both.

It is important to distinguish cases of ambiguity—English sentences that have more than one meaning—from sentences for which there is more than one good translation, but which are equivalent. We see this with “No sharks are good pets”: $\forall x(Sx \rightarrow \sim(Gx \wedge Px))$ and $\sim \exists x(Sx \wedge (Gx \wedge Px))$ are equally good (and equivalent) translations.

5.3 Exercises

5.3.1 SL to English Translations

Given the following glossary, translate the following SL sentences into English.

Glossary:

C : Cindy the Capybara is a picky eater.

O : Oscar the Ocelot sleeps all day.

R : Ralph the Rhinoceros goes for a swim.

A : France is east of Spain.

- | | |
|----------------------------|--|
| 1. $C \rightarrow O$ | 5. $(C \wedge O) \vee \sim(C \leftrightarrow A)$ |
| 2. $O \rightarrow C$ | 6. $C \wedge (O \vee \sim(C \leftrightarrow A))$ |
| 3. $\sim(R \rightarrow A)$ | 7. $A \rightarrow \sim(C \wedge R)$ |
| 4. $\sim R \rightarrow A$ | 8. $(C \wedge R) \rightarrow \sim A$ |

5.3.2 English to SL Translations #1

Using some sensible translation key translate the following English sentences into SL.

1. If the sprockets come in on time, then we can fill the order.
2. Only if the sprockets come in on time can we fill the order.
3. Either the order gets filled, or the cogs come in late and the sprockets never show up.
4. It's not the case that the sprockets need to come in for the order to be filled.
5. While filling the order is important, getting the sprockets in is more so.
6. The sprockets and cogs are late, but it's still not the case that we can't fill the order on time.
7. Assuming the order gets out on time, the sprockets will fail to arrive only if the cogs are either late or defective.
8. The spork is the least appreciated utensil.
9. They dined on mince, and slices of quince, [which] they ate with a runcible spoon. (1871, Edward Lear, "Owl & Pussy-Cat" in *Nonsense Songs*)
10. You eat with a spork if and only if you eat with a foon.
11. Although Jan will be amused, if you eat with a spork Jill will leave or at least not laugh.
12. If you have a runcible spoon, then you don't need a fork, knife, or spoon.

5.3.3 English to SL Translations #2

Using some sensible translation key translate the following English sentences into SL.

1. If 14-year-olds had the vote, I'd be president. (Evel Knievel)
2. If Miami beats Cornell today and Penn State defeats Michigan State Miami will win the tournament.
3. Should senator Ervin run again, he would be a formidable opponent.
4. If Congress does not find a way to force the banking industry to lower interest rates, and if the Securities and Exchange Commission does not stop authorizing unjustified new financing by corporations, we will face an unbearable depression.
5. Provided, but only provided, that the French Fleet is sailed forthwith for British harbors, His Majesty's Government give their full consent to an armistice for France. (Churchill, June 1940)

6. For the tenability of the thesis that mathematics is logic it is not only sufficient but also necessary that all mathematical expressions be capable of definition on the basis solely of logical ones. (W.V.O. Quine)

5.3.4 Translations

Translate each of the following English sentences into QL sentences about the model **m** given in table 5.6 (on the next page).

- | | |
|---|---|
| 1. All Pacific states that border a mountainous state are coastal. | 13. All Atlantic states are not mountainous. |
| 2. Some Atlantic state and some mountainous state both share a border with a state that is neither. | 14. Some Gulf state is larger than all states that border it. |
| 3. All states are coastal and mountainous if and only if they are Pacific. | 15. Some Gulf state is an Atlantic state. |
| 4. All Atlantic states smaller than Montana share a border with Rhode Island. | 16. All states that border a Pacific state are mountainous. |
| 5. Only mountainous Pacific states are coastal. | 17. Any state that is mountainous is larger than Rhode Island. |
| 6. A Pacific state is mountainous. | 18. Any state that is mountainous is larger than all Atlantic states. |
| 7. No state is larger than itself. | 19. If any state is mountainous, California is. |
| 8. Every non-mountainous state borders a state that is larger. | 20. If any state is mountainous, it is larger than Rhode Island. |
| 9. Some Pacific states are mountainous. | 21. Any state that has no bordering states is mountainous. |
| 10. All Pacific states are mountainous. | 22. All states that are bigger than all mountainous states are coastal. |
| 11. All Pacific states are larger than all Atlantic states. | 23. No state is bigger than Montana unless it is coastal. |
| 12. No Gulf state is mountainous. | |

	Symbol	Model Assignment
Universe:		The set of states
Constants:	c	CA
	m	MT
	h	RI
	e	TX
1 place predicates:	P'	Pacific states
	A'	Atlantic states
	G'	Gulf states
	M'	Mountainous states
	C'	Coastal states
2 place predicates:	L''	is larger than (area)
	B''	borders

Table 5.6: Model for Translations in Section 5.3.4

5.3.5 More Translations

Translate each of the following English sentences into QL sentences.

1. All beavers avoid some kangaroo.
2. All beavers avoid all kangaroos.
3. Some beaver avoids all kangaroos.
4. Every kangaroo is avoided by some beaver.
5. All beavers avoid any kangaroo that frightens them.
6. Some beavers avoid any kangaroo that frightens them.
7. No kangaroo frightens any beaver.
8. No beaver is frightened by any kangaroo.
9. No beaver avoids a kangaroo unless the kangaroo frightens it.
10. Some kangaroo frightens itself.
11. No beaver avoids a kangaroo unless the beaver frightens the kangaroo.
12. Any kangaroo that is frightened of itself is frightened by any beaver.
13. Beavers avoid kangaroos only if they frighten them.
14. Kangaroos that frighten beavers frighten themselves.
15. All kangaroos avoid any kangaroo that avoids them.
16. When a kangaroo frightens a beaver, the beaver avoids it.
17. Beavers only avoid kangaroos.
18. Beavers are frightened of all kangaroos unless they avoid them.
19. Some beavers avoid only kangaroos that frighten them.
20. No beaver that avoids all kangaroos frightens itself.

Chapter 6

Derivations

6.1 Introduction

We were able to make sense of semantic notions like truth, logical truth, and entailment by introducing models for SL and QL. But proving that a sentence of SL or QL is a logical truth, or proving that an entailment holds, is difficult and involves informal reasoning in MathEnglish. We'd like a way to prove such things that's easier and which relies less on intuitive judgment. Recall from section 1.2.2 that SL and QL are formal in the sense that which strings of basic symbols are sentences of each is given by explicit definitions that make reference only to the shapes and arrangement of those basic symbols. Can we find a *formal* method for establishing logical truth and entailments?

For this purpose we'll introduce formal rules for manipulating SL and QL sentences which will allow us to write formal derivations. Roughly, a *derivation* is a finite sequence of sentences, with rules governing which sentences may follow which. For every sentence in the derivation there is a rule which sanctions (or permits) us to write it down next, usually depending on the previous sentences in the derivation.¹ Our derivations are *formal* because the rules they use are formal. That is, the rules are specified entirely in terms of the shape, or form, of the sentences.

This raises an important point: since the rules are formal, there is no *prima facie* connection between them (or derivations written using them) and our semantic notions of truth, logical truth, and entailment. For example, the rules will not say things like:

1. If $\phi \models \psi$ and you have ϕ on a previous line, then you can write down ψ .

Instead, they'll say things like:

2. If you have $\phi \wedge \psi$ on a previous line, then you can write down ψ .

Despite this lack of a transparent connection, we want our rules to match up with the semantics we defined in previous chapters. For example, an important constraint is that the rules are truth-preserving.

¹Often times the terms 'derivation' and 'proof' (or 'formal proof') are used interchangeably. The subfield of logic that studies derivations is even called proof theory. Here, for clarity, we'll always use 'derivation' to refer to the formal proofs we'll work with in this chapter and 'proof' to refer to the more informal mathematical proofs we write in MathEnglish.

Truth-preservation: If a rule allows us to write down ϕ when applied to sentences ψ_1, \dots, ψ_n , then $\psi_1, \dots, \psi_n \models \phi$.

More generally, because we want to use the derivations as a means of showing that a sentence is a logical truth or of showing that some set of sentences entails some other sentence, we want at least:

Soundness: If ϕ can be derived from ψ_1, \dots, ψ_n , then $\psi_1, \dots, \psi_n \models \phi$.

And ideally we'd like to be able to get:

Completeness: If $\psi_1, \dots, \psi_n \models \phi$, then ϕ can be derived from ψ_1, \dots, ψ_n .

We will discuss and prove these two results (and variations of them) for both SL and QL in the next chapter.

While our derivation systems (our sets of rules with which we write derivations) are definitionally independent of our semantics (our models and definitions for truth, logical truth, and entailment), there are in fact deep and important connections between them. We'll prove the most important of these connections—including soundness and completeness—in the next chapter. For now though it will probably be beneficial not to worry about them and to approach derivations on their own terms. It can be useful to think of writing a derivation as solving a puzzle, or trying to win a game. Like any game there are rules for the moves you can make, or like a puzzle there's a way the inferences made in the derivation can fit together. (Although, unlike a puzzle there are always many ways they can fit together.)

6.2 The Basic System SD

6.2.1 Introduction and Elimination Rules

We will work with two kinds of rules: basic rules and shortcut rules. The shortcut rules are, as the name suggests, not necessary. Their purpose will be to make derivations easier and shorter. Anything we can derive with the shortcut rules can be derived from the basic rules alone (see thm. 6.5, on page 152). The basic rules make up what we call *Sentential Derivation System*, or SD. SD consists of two special rules, *Assumption* and *Repetition*, and a pair of basic rules for each of the logical connectives in SL. When we move on to QL we'll add more basic rules for the logical connectives unique to QL, i.e. the quantifiers. One of the pair will be an *introduction* rule, the other will be an *elimination* rule.

For example, consider the conjunction, \wedge . If we earlier derived the sentences ϕ and ψ , we may introduce their conjunction $\phi \wedge \psi$ with the \wedge introduction rule. Alternatively, if we have derived $\phi \wedge \psi$, the \wedge elimination rule lets us take out a part of the conjunction and write either ϕ or ψ . Elimination rules typically give us a choice for the next step, and learning to use the system effectively partly involves learning

strategies for choosing the next step. Table 6.1 (on the current page) gives all the basic SD rules.²

Name	Given	May Add
<i>Assume</i>		ϕ
<i>Rep.</i>	ϕ	ϕ
\rightarrow - <i>Elim</i>	$\theta \rightarrow \psi, \theta$	ψ
\rightarrow - <i>Intro</i>	θ $:$ ψ	$\theta \rightarrow \psi$, Draw box ³
\wedge - <i>Elim</i>	$\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n$	Any one of the conjuncts i.e., θ_i
\wedge - <i>Intro</i>	$\theta_1, \theta_2, \dots \theta_n$	$\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n$
\vee - <i>Elim</i>	$\theta_1 \vee \theta_2 \vee \dots \vee \theta_n,$ $\theta_1 \rightarrow \psi,$ $\theta_2 \rightarrow \psi,$ $:$ $\theta_n \rightarrow \psi$	ψ
\vee - <i>Intro</i>	θ	$\psi_1 \vee \psi_2 \vee \dots \vee \psi_n,$ where θ is ψ_i for some i .
\sim - <i>Intro</i>	$\theta \rightarrow (\psi \wedge \sim \psi)$	$\sim \theta$
\sim - <i>Elim</i>	$\sim \theta \rightarrow (\psi \wedge \sim \psi)$	θ

Table 6.1: Basic Rules of SD

Continued next Page

²As mentioned in the preface, this kind of introduction and elimination rule-based derivation system is called a *natural deduction* system. Natural deduction systems were first invented by Stanislaw Jaskowski in 1926, though he did not publish until 1934. Other important developments are due to Gerhard Gentzen (1934); other influential natural deduction systems are given by J. M. Anderson and H. W. Johnstone (1962), Frederic Fitch (1952), Donald Kalish and Richard Montague (1964), Lemmon E. J. (1965), Dag Prawitz (1965), Willard Quine (1950), Patrick Suppes (1957), Neil Tennant (1978), R. H. Thomason (1970), and Dirk van Dalen (1980) (Hodges 2001b: 28). Unlike many systems, ours uses boxes to discharge assumptions; Jaskowski mentions in the 1934 article that he used boxes in 1926, but they aren't part of his official system in 1934.

³When using the rule \rightarrow -Intro, you must draw a box around all lines from the θ to the ψ . The line with θ must be sanctioned by the rule 'Assume'.

Continued from Previous Page

Name	Given	May Add
\leftrightarrow -Intro	$\theta \rightarrow \psi, \psi \rightarrow \theta$	$\theta \leftrightarrow \psi$
\leftrightarrow -Elim	$\theta \leftrightarrow \psi, \psi$	θ
\leftrightarrow -Elim	$\theta \leftrightarrow \psi, \theta$	ψ

Table 6.1: Basic Rules of SD

Before doing some examples, two notes about the rules are in order. First, to use the *Assumption* rule, you must draw a new vertical line to the left of your assumption (and to the right of any other vertical lines from previous steps). The vertical lines are assumption lines; assumption lines are an accounting device to keep track of what you have assumed as opposed to what you have derived (see example 8, on page 132). You may assume any sentence you like, but generally these assumptions (usually all of them) will need to be discharged. Vertical assumption lines are like your credit card balance. As with a credit card, managing assumptions wisely is a skill to be acquired.

How do you get rid of an assumption? And how do you keep track of them when you are adding some and eliminating others? We have a second accounting device: boxes. When an assumption is discharged, the relevant assumption line is turned into a box containing everything derived from the originating assumption. Writing boxes can sometimes seem tedious, but we have found that using them significantly reduces student error.

Assumptions are discharged using the \rightarrow -Intro rule. For \rightarrow -Intro you need a (part of a) derivation that begins with an unboxed assumption $| \theta$ and ends with $| \psi$. After you've added $\theta \rightarrow \psi$ to your derivation, you *must* box off the part of the derivation that began with assumption $| \theta$ and ended with $| \psi$. That is, if we have the following:

3.	1.	$ \theta$	<i>Assume</i>
	2.	$ $	
	3.	$ \quad :$	
	4.	$ $	
	5.	$ \psi$	

we may then use the rule \rightarrow -Intro to get:

4.	1.	θ	<i>Assume</i>
	2.		
	3.	\vdots	
	4.		
	5.	ψ	
	6.	$\theta \rightarrow \psi$	\rightarrow -Intro, 1–5

The point in boxing off that part of the proof is to indicate visually that those sentences in the box can't be used anymore.

6.2.2 Writing Derivations

We've said, in rough terms, that a derivation is a finite series of sentences each of which is either an assumption, or there is a rule which, given the previous sentences in the derivation, sanctions (or permits) us to write down that sentence. We'll add more structure to derivations for the sake of visual clarity and organization. First, we'll write the sentences vertically with earlier sentences above later ones so that each sentence sits on its own row in the derivation. Each row is to be numbered with consecutive positive integers, starting with 1. In a column to the right on each row we'll put either *Assume*, if the sentence is an assumption, or the name of the rule plus the numbers of the previous rows to which the rule was applied in order to get the sentence on the current row. Lastly, in a column between the row numbers and the sentences we'll keep track of the assumptions by running vertical lines in the column.

As an example, consider how we'd derive $C \wedge B$ from $B \wedge C$. This derivation would begin:

5.	1.	$B \wedge C$	<i>Assume</i>
----	----	--------------	---------------

Note that \wedge -Elim allows us to write down one of the conjuncts of a conjunction we already have. The conjuncts of $B \wedge C$ are B and C . Accordingly, we may write down B and C on new lines. We continue:

6.	1.	$B \wedge C$	<i>Assume</i>
	2.	B	\wedge -Elim, 1
	3.	C	\wedge -Elim, 1

Finally, using \wedge -Intro we put the sentences from lines 2 and 3 together in a conjunction:

7.	1.	$B \wedge C$	<i>Assume</i>
	2.	B	\wedge -Elim, 1
	3.	C	\wedge -Elim, 1
	4.	$C \wedge B$	\wedge -Intro, 2,3

This is a four-line derivation of $C \wedge B$ from $B \wedge C$. Notice that steps 2 and 3 could have been done in opposite order. In many cases the order of sentences in a derivation doesn't matter, but in other cases it is crucial. Learning the difference is an important skill.

Just as we used the double turnstile to represent when one sentence (or a set of sentences) entailed another, we use what's called the *single turnstile*, ' \vdash ', to represent when one sentence is derivable from another, or from another (finite) set of sentences. The four-line derivation shows that $B \wedge C \vdash C \wedge B$. It's also worth noting that each of the two partially completed pieces of the derivation just given are derivations themselves. The first (5) is a one-line derivation that shows that $B \wedge C \vdash B \wedge C$, while the second (5) is a three-line derivation that shows that $B \wedge C \vdash C$.

Next, as an example of how assumption lines stack up in a derivation, consider the following derivation of D from $A \wedge B$ and $B \rightarrow (C \wedge D)$.

8.	1.	$A \wedge B$	<i>Assume</i>
	2.	$B \rightarrow (C \wedge D)$	<i>Assume</i>
	3.	B	\wedge -Elim, 1
	4.	$C \wedge D$	\rightarrow -Elim, 2,3
	5.	D	\wedge -Elim, 4

As is shown, any time a new assumption line is added it must be placed to the right of the previous (unboxed) assumption lines. (We discuss this example a little more in the next section, while discussing the basic top-down and bottom-up strategies for \rightarrow .)

The last major point about the mechanics of writing a derivation is how the assumptions are closed, or discharged, when \rightarrow -Intro is used. This rule says that if you have a sentence ψ on a line as an assumption, and you've worked your way down to a line with sentence θ , then you can "close off" the assumption by drawing a box around that part of the proof and writing $\psi \rightarrow \theta$ on the next line. We can demonstrate this by applying the rule to close off the assumptions in the previous two proofs:

9.	1.	$B \wedge C$	<i>Assume</i>
	2.	B	\wedge -Elim, 1
	3.	C	\wedge -Elim, 1

4. $\boxed{C \wedge B}$ $\wedge\text{-Intro, 2,3}$
5. $(B \wedge C) \rightarrow (C \wedge B)$ $\rightarrow\text{-Intro, 1-4}$

10.
 1. $A \wedge B$ Assume
 2. $B \rightarrow (C \wedge D)$ Assume
 3. B $\wedge\text{-Elim, 1}$
 4. $C \wedge D$ $\rightarrow\text{-Elim, 2,3}$
 5. D $\wedge\text{-Elim, 4}$
 6. $(B \rightarrow (C \wedge D)) \rightarrow D$ $\rightarrow\text{-Intro, 2-5}$
 7. $(A \wedge B) \rightarrow ((B \rightarrow (C \wedge D)) \rightarrow D)$ $\rightarrow\text{-Intro, 1-6}$

In these two cases, since we've boxed off the assumptions they are no longer derivations of some sentence from another (or others). The first is a derivation of $(B \wedge C) \rightarrow (C \wedge B)$, while the second is a derivation of $(A \wedge B) \rightarrow ((B \rightarrow (C \wedge D)) \rightarrow D)$. When we have a derivation from no assumptions (or, rather, with all assumptions boxed), we represent that with a single turnstile with no formulas on the left. So these two derivations show, respectively, that $\vdash (B \wedge C) \rightarrow (C \wedge B)$ and $\vdash (A \wedge B) \rightarrow ((B \rightarrow (C \wedge D)) \rightarrow D)$.⁴

It is important to note that once we have closed off an assumption we can no longer use the sentences that we've boxed off. We can't apply rules to them anymore; they are no longer 'Given', to use the term in the Basic Rules chart. The reason we can't use sentences inside boxes is because if we could, then SD would not be sound. That is, we would be able to derive a sentence ϕ from others ψ_1, \dots, ψ_n but $\psi_1, \dots, \psi_n \models \phi$ would not hold. To see this, consider the following example.

11.
 1. $A \rightarrow B$ Assume
 2. A Assume
 3. B $\rightarrow\text{-Elim, 1,2}$
 4. $A \rightarrow B$ $\rightarrow\text{-Intro, 2-3}$
 5. B $\rightarrow\text{-Elim, 2,4}$

In line 5 we go back into the box to use A from line 2 to get B from line 4 using $\rightarrow\text{-Intro}$. But it should be easy to see that $A \rightarrow B \models B$ does not hold. (Consider a model \mathbf{m} such that $\mathbf{m}(A) = \text{F}$ and $\mathbf{m}(B) = \text{F}$.)

⁴If $\vdash \phi$, it's often said that ϕ is a theorem of the derivation system. "Theorem" is used ambiguously between the derivation of an SL(or QL) sentence from the empty set of assumptions and things we prove in MathEnglish. We will follow that standard practice.

6.2.3 *The Recursive Definition of a Derivation

Now that we've done some examples, we give an explicit definition of a derivation. The definition of a derivation is recursive. The explicit definition we give here is mainly for the sake of proving soundness in the next chapter. Although the actual definition is complicated, the basic idea is straightforward: A single SL sentence that's an assumption is a derivation (e.g., derivation 5), and any finite sequence of SL sentences every one of which (after the first) is either an assumption or sanctioned by some rule of SD is a derivation. We have to complicate this to handle the rule \rightarrow -Intro and *Assumption*, neither of which quite work like the other rules. To understand these rules we must first define what an *open assumption* is. An assumption is open iff it appears on an assumption line and is not in a box.

Definition 6.1. The following recursive clauses fix which finite sequences of derivation lines are *derivations* in SD:

Base Clause: For any SL sentence ϕ , the following single derivation line (i.e., sequence of derivation lines of length 1) is a derivation:

$$1. \quad \boxed{\phi} \quad \text{Assume}$$

Generating Clause:

Case 1: If you have some n -line derivation with k assumptions still open at line n , and in which the sentence ψ on n is sanctioned by rule R when applied to lines j_1, \dots, j_k ,

$$\begin{array}{l} 1. \\ \vdots \\ n. \quad \psi \quad R, j_1, \dots, j_k \end{array}$$

then the sequence of derivation lines you get by adding a new line $n + 1$ with k open assumptions and the sentence θ is a derivation,

$$\begin{array}{l} 1. \\ \vdots \\ n. \quad \psi \quad R, j_1, \dots, j_k \\ n + 1. \quad \theta \quad R', h_1, \dots, h_l \end{array}$$

so long as θ is sanctioned by some rule R' (other than \rightarrow -Intro) when applied to previous lines h_1, \dots, h_l already in the derivation.

Case 2: If you have some n -line derivation with k assumptions still open at line n , and in which the sentence ψ on n is sanctioned by rule R when applied to lines j_1, \dots, j_k ,

$$\begin{array}{c} 1. \\ \vdots \\ n. \quad \left| \psi \quad R, j_1, \dots, j_k \right. \end{array}$$

then, for any sentence θ , the sequence of derivation lines you get by adding a new line $n+1$ with $k+1$ open assumptions and θ sanctioned by *Assumption* is a derivation,

$$\begin{array}{c} 1. \\ \vdots \\ n. \quad \left| \psi \quad R, j_1, \dots, j_k \right. \\ n+1. \quad \left| \quad \left| \theta \quad \text{Assume} \right. \right. \end{array}$$

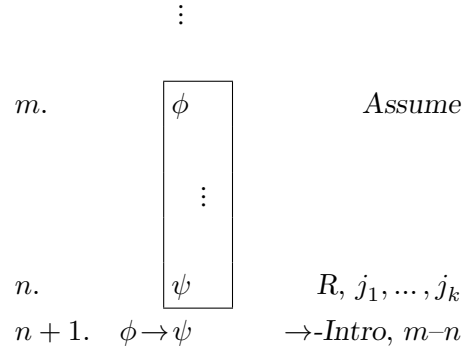
Case 3: If you have some n -line derivation with k assumptions still open at line n , and in which the sentence ψ on n is sanctioned by rule R when applied to lines j_1, \dots, j_k , and in which the k th assumption was opened on line m ,

$$\begin{array}{c} 1. \\ \vdots \\ m. \quad \left| \phi \quad \text{Assume} \right. \\ \vdots \\ n. \quad \left| \psi \quad R, j_1, \dots, j_k \right. \end{array}$$

then the sequence of derivation lines you get by adding a new line $n+1$

with $k - 1$ open assumptions and the sentence $\phi \rightarrow \psi$ is a derivation,

1.



so long as you close the assumption opened on line m by drawing a box around lines $m-n$ and write down $\rightarrow\text{-Intro, } m-n$ as the rule which sanctions line $n+1$.

Closure Clause: Nothing else is a derivation of SL.

Note that this definition of a derivation makes use of something, sanctioning, which we haven't yet defined explicitly. Definition 7.3 will make this explicit too. Also note that in the generating clauses we specify that we start with a derivation with k open assumptions on the last line. In cases 1 and 2 the schematic drawings given don't explicitly depict the k vertical lines that should be running between the line numbers and the sentences, but the drawings are not intended to suggest that you can write derivations without the running vertical lines which track open assumptions. The schematic drawing in case 3 similarly only depicts the last vertical assumption line (the one for the assumption opened on line m), but that's not to suggest the others aren't there or can be left off.

6.2.4 Restrictions on Applying Rules

Above we noted that every line of a derivation must either be an assumption, or there must be a rule which, applied to some previous unboxed sentences in the derivation, *sanctions* us to write down that sentence. (Extending a derivation with the introduction rule for \rightarrow , $\rightarrow\text{-Intro}$, as in case 3 of definition 6.1 is a special case, but still we say that the rule sanctions writing down the new sentence.) It's very important to be clear on just when a rule sanctions (or we might say *licenses*) writing down a sentence. The restriction, which we tacitly followed in the examples given above, is that a rule can be applied only if the connectives mentioned in the rule are the main connectives of the sentences to which that rule is being applied. That is, a rule can only be applied to whole sentences on a line—it can't be applied to subsentences on a line.

For example, we can only apply \rightarrow -Elim on two lines of a derivation if the sentence on one of the lines is a conditional $\phi \rightarrow \theta$ and the sentence on the other line is ϕ . If $\phi \rightarrow \theta$ is the sentence on one line and ϕ is merely contained as a subsentence in the sentence on the other (say the sentence has the form $\phi \wedge \psi$), then we cannot apply \rightarrow -Elim to those lines. Likewise, if a line has a sentence ϕ and the conditional $\phi \rightarrow \theta$ is merely contained as a subsentence in the sentence on another line (say the sentence has the form $(\phi \rightarrow \theta) \vee \psi$), then we cannot apply \rightarrow -Elim to those lines.

For a more concrete example, consider derivation 10, on page 133. Even though B appears on line 1 (as the conjunct of $A \wedge B$), we cannot apply \rightarrow -Elim to lines 1 and 2 to get $C \wedge D$. The fact that we can easily get B on its own line through \wedge -Elim doesn't matter. The rule \rightarrow -Elim will only sanction writing $C \wedge D$ on a line, given $B \rightarrow (C \wedge D)$ on line 2, if we have another line with the LHS of the conditional by itself. In just the same way we cannot apply \wedge -Elim to line 2 of derivation 10 to get C or D , since the conjunction in line 2 is not the main connective. Instead, it's the main connective of the RHS subsentence of $B \rightarrow (C \wedge D)$. Again it doesn't matter that we can get the conjunct by itself using \rightarrow -Elim (after using \wedge -Elim on line 1), the rule \wedge -Elim cannot be applied to a line unless the sentence on that line is a conjunction.

One reason for this restriction is because without it many applications of the rules would not be *truth-preserving*. One of the goals of SD is that our derivation system “mirrors” the results we could prove in SL. In other words, we want something to be derivable in SD iff we can prove the same result semantically in SL.

However, we are not yet ready to show that our derivation system has this property. We will not establish this result until next chapter. For now, we note merely that a new line is derivable solely in virtue of the *form* of previous lines, or in virtue of the rules for introducing a new line. Our derivation system is nothing more than a formal model, with *stipulated* restrictions regarding how to apply the rules we define. It does not, in and of itself, say anything about what is true or false in SL.

6.2.5 *Decidability

There are multiple algorithms to follow for applying the rules which, if there does exist a derivation, will halt when the last line written down is the sentence to be derived. For SD the algorithms are intuitive and straightforward (see Sec. 7.3, on page 188), while for QD (the basic derivation system for QL we'll introduce in Sec. 6.4, on page 164) they need to become much more complicated. In the next chapter (Sec. 7.4.3, on page 197) we will give one such algorithm for QD, which we call The Method. It will be crucial for showing some of the important connections between derivations and entailment mentioned in section 6.1, including completeness.

But although these algorithms are guaranteed to end in a derivation if the sentence can be derived, there are at least two reasons why you don't want to do most of your derivations using them. First, the derivations produced by them tend to be much longer and more complicated than is necessary. You will almost always be able to come up with a much shorter and more direct proof on your own. Second, at least

for QD (but not SD), although *if* there is a derivation the algorithms will “find it”, if there is *not* a derivation then the algorithms may never “find out”. That is, if there is not a derivation then the algorithms (any one you pick) do essentially one of two things: either they halt in a way that indicates there is no derivation, or they never halt. If you happen to be working on a problem in the latter case and you’re only following the algorithm, then you’ll never find out whether there is a derivation. (If a derivation system, like QD, has this feature, then it’s said to be *undecidable*. If there is an algorithm that always halts either in a derivation or with an indication that there’s no derivation (as in the case of SD), then the system is said to be *decidable* and the algorithm is said to be a *decision procedure*. Note that if we restrict QL to just 1-place predicates, then QD is decidable. See section 7.6, on page 209 for more details and discussion.) At any point in applying the algorithm you’ll have neither produced a derivation or reached any indication that there is no derivation.

Enderton (2010) provides a general, contemporary introduction to computability and decision procedures. Kleene (2002 [1967]: ch. 5) provides a lucid and concise discussion within roughly the framework devolved here.

6.2.6 Some Strategies

We don’t want to use these sorts of algorithms to find derivations, if we can avoid it. But there are general strategies we will use. For each logical connective there are two types of strategies: those for what to do if you already have sentences with that as their main connective, and those for what to do if you want to get a sentence with that as its main connective. We’ll call the first top-down strategies and the second bottom-up strategies. In many cases, doing a derivation is like planning a plane trip. The top-down method is like figuring out the nearest convenient airport from your current location, and the bottom-up method is like figuring out which airport is convenient for getting to your destination. In derivations, sometimes you have to go through intermediate sentences (as with intermediate cities in travel).

We will begin with some basic top-down and bottom-up strategies for each connective, adding more later in section 6.3.3 when we add shortcut rules.

Conjunction We start with the basic top-down and bottom-up strategies for conjunction. They are the most straightforward.

\wedge Top-down: If you have a sentence of the form $\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n$, then break it apart using \wedge -Elim to get each of the conjuncts ϕ_1 through ϕ_n , each on a new line.

\wedge Bottom-up: If you want to get a sentence of the form $\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n$, then derive each of ϕ_1 through ϕ_n individually and use \wedge -Intro to derive it from them.

Both strategies are exemplified in example derivation 7, on page 132. There we wanted to derive the sentence $C \wedge B$, so in line with the bottom-up strategy for \wedge we first derived both C and B and then used \wedge -Intro to derive $C \wedge B$. In line with the top-down strategy, we took our assumption $B \wedge C$ and broke it apart using \wedge -Elim (which

got us the sentences, C and B , we were looking to derive).

Conditionals The basic strategies for conditionals are also straightforward and have already been exemplified.

→ **Top-down:** If you have a sentence of the form $\phi \rightarrow \psi$, then first derive the LHS ϕ and then break it apart using \rightarrow -Elim to get the RHS ψ on a new line.

→ **Bottom-up:** If you want to get a sentence of the form $\phi \rightarrow \psi$, then assume the LHS ϕ , derive the RHS ψ , and then use \rightarrow -Intro to write the conditional on the next line.

In derivation 8, on page 132, we wanted to derive D . We saw that we had a conditional, $B \rightarrow (C \wedge D)$. In line with the top-down strategy, we derived its LHS B (in the process using the top-down strategy for \wedge), then used \rightarrow -Elim to get the RHS $(C \wedge D)$. We wanted $(C \wedge D)$, of course, because from it we could use \wedge -Elim to get D . In derivation 10, on page 133, we wanted to derive $(A \wedge B) \rightarrow ((B \rightarrow (C \wedge D)) \rightarrow D)$. In line with the bottom-up strategy, we assumed the LHS $(A \wedge B)$, derived the RHS $((B \rightarrow (C \wedge D)) \rightarrow D)$, and then used \rightarrow -Intro to write the conditional on the next line.

Biconditionals The basic strategies for biconditionals are similar to those for conditionals, as one might expect.

↔ **Top-down:** If you have a sentence of the form $\phi \leftrightarrow \psi$, then either

1. first derive the LHS ϕ and then break it apart using \leftrightarrow -Elim to get the RHS ψ on a new line,
2. first derive the RHS ψ and then break it apart using \leftrightarrow -Elim to get the LHS ϕ on a new line,
3. or do both.

↔ **Bottom-up:** If you want to get a sentence of the form $\phi \leftrightarrow \psi$, then first derive both $\phi \rightarrow \psi$ and $\psi \rightarrow \phi$ and then use \leftrightarrow -Intro to write the biconditional on the next line.

Negations In the case of negations, we don't have a basic top-down strategy, only a basic bottom-up. Later in this chapter we will develop shortcut rules which allow us to provide top-down strategies for negation.

~ **Bottom-up:** If you want to get a sentence of the form $\sim\phi$, then first assume ϕ , derive a contradiction $\psi \wedge \sim\psi$, and then in two separate steps use \rightarrow -Intro and \sim -Intro to write the negation on the next line.

For example, say we want to derive the sentence $\sim((A \rightarrow \sim B) \wedge (A \wedge B))$. In line with the basic bottom-up strategy for \sim , we first assume $((A \rightarrow \sim B) \wedge (A \wedge B))$ and try to derive a contradiction:

12.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
		\vdots	
	$n.$	$\psi \wedge \sim \psi$	

It should be clear that we can derive $B \wedge \sim B$, so that will be our goal:

13.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
		\vdots	
	$n.$	$B \wedge \sim B$	

The bottom-up strategy for \wedge says to get this we should derive both B and $\sim B$.

14.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
		\vdots	
	$n-2.$	B	
	$n-1.$	$\sim B$	
	$n.$	$B \wedge \sim B$	\wedge -Intro, $n-1, n-2$

The bottom-down strategy for \wedge is our only option at this point, so we break line 1 apart using \wedge -Elim.

15.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
	2.	$(A \rightarrow \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
		\vdots	
	$n-2.$	B	
	$n-1.$	$\sim B$	
	$n.$	$B \wedge \sim B$	\wedge -Intro, $n-1, n-2$

Now we continue to work bottom-down, using \wedge -Elim to break apart line 3. Note that in this step we've partway joined up the top and bottom of the proof, since breaking apart line 3 gets us what we were calling line $n-2$.

16.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
	2.	$(A \rightarrow \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	A	\wedge -Elim, 3
	5.	B	\wedge -Elim, 3
		\vdots	
	$n-1.$	$\sim B$	
	$n.$	$B \wedge \sim B$	\wedge -Intro, $n-1, n-2$

Next we see that we can work bottom-down from lines 2 and 4, breaking the conditional on line 2 apart. In doing so we finish the proof, since the result of doing this is $\sim B$, which is all that was left to get the contradiction.

17.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
	2.	$(A \rightarrow \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	A	\wedge -Elim, 3
	5.	B	\wedge -Elim, 3
	6.	$\sim B$	\rightarrow -Elim, 2,4
	7.	$B \wedge \sim B$	\wedge -Intro, 5,6

Of course, we haven't yet derived $\sim((A \rightarrow \sim B) \wedge (A \wedge B))$, but we can now do so by discharging the assumption through \rightarrow -Intro and then applying \sim -Intro.

18.	1.	$((A \rightarrow \sim B) \wedge (A \wedge B))$	<i>Assume</i>
	2.	$(A \rightarrow \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	A	\wedge -Elim, 3
	5.	B	\wedge -Elim, 3
	6.	$\sim B$	\rightarrow -Elim, 2,4
	7.	$B \wedge \sim B$	\wedge -Intro, 5,6

- $$\begin{array}{ll}
 8. & \overline{((A \rightarrow \sim B) \wedge (A \wedge B)) \rightarrow (B \wedge \sim B)} \quad \rightarrow\text{-Intro, 1-7} \\
 9. & \sim((A \rightarrow \sim B) \wedge (A \wedge B)) \quad \sim\text{-Intro, 8}
 \end{array}$$

Disjunctions Our last pair of strategies is for disjunctions. As with conjunctions, we give the strategies for the case where there are only two disjuncts. Generalizing the strategies for disjunctions with more than two disjuncts is left to the reader.

\vee **Top-down:** If you have a sentence of the form $\phi \vee \psi$ and you want to derive a sentence θ , first derive the conditionals $\phi \rightarrow \theta$ and $\psi \rightarrow \theta$, and then use \vee -*Elim* to write down θ on the next line. The order in which you derive the intermediate conditionals doesn't matter.

\vee **Bottom-up:** If you want a sentence of the form $\phi \vee \psi$, then first derive either ϕ or derive ψ , and then use \vee -*Intro* to write it down.

The basic bottom-up strategy isn't always the right tool for deriving disjunctions, because usually you can't derive one of the disjuncts. This is an important point: it might be that a disjunction is derivable even if neither disjunct is. It's important that we can derive disjunctions without first deriving one or the other disjunct, since $B \vee \sim B$ is a logical truth. We would like to be able to derive it, though neither B nor $\sim B$ is a logical truth.

Later on we'll get more useful bottom-up strategies for disjunctions. For now we'll focus on the top-down strategy.

As an example, say we want to derive $\sim((\sim A \vee \sim B) \wedge (A \wedge B))$. The whole sentence itself is a negation, so we start just as in the last example. So we need some contradiction to aim for. We'll try to derive $B \wedge \sim B$.

- $$\begin{array}{ll}
 19. & \begin{array}{l|l}
 1. & ((\sim A \vee \sim B) \wedge (A \wedge B)) \quad \textit{Assume} \\
 & \vdots \\
 n. & B \wedge \sim B
 \end{array}
 \end{array}$$

As in the last example the bottom-up strategy for \wedge has us try to derive both B and $\sim B$.

- $$\begin{array}{ll}
 20. & \begin{array}{l|l}
 1. & ((\sim A \vee \sim B) \wedge (A \wedge B)) \quad \textit{Assume} \\
 & \vdots
 \end{array}
 \end{array}$$

$n - 2.$	B	
$n - 1.$	$\sim B$	
$n.$	$B \wedge \sim B$	\wedge -Intro, $n - 2, n - 1$

And, the top-down strategy leads us to break apart the conjunction on line 1, which leads to another conjunction to break apart as well. This gets us one of the conjuncts of line n in the process.

21.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	Assume
	2.	$(\sim A \vee \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	B	\wedge -Elim, 3
		\vdots	
	$n - 1.$	$\sim B$	\sim -Intro, $n - 2$
	$n.$	$B \wedge \sim B$	\wedge -Intro, $n - 2, n - 1$

Now we only need to get $\sim B$. To do this, we follow the bottom-up strategy for negation. We assume B and try to get a contradiction.

22.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	Assume
	2.	$(\sim A \vee \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	B	\wedge -Elim, 3
	5.	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> B \vdots $\psi \wedge \sim \psi$ </div>	Assume
	$n - 3.$		
	$n - 2.$	$B \rightarrow (\psi \wedge \sim \psi)$	\rightarrow -Intro, $5 - n - 3$
	$n - 1.$	$\sim B$	\sim -Intro, $n - 2$
	$n.$	$B \wedge \sim B$	\wedge -Intro, $n - 2, n - 1$

Note that although it may look wrong to have an assumption line for B when we have already derived it, there's nothing wrong with this derivation. You can always assume

whatever you want, even if you already have it. Now we have two questions to think about: what contradiction could we get on line $n - 3$? And, how do we get it? The second question is straightforward. We want to get a contradiction, and at this point it's going to have to come from the disjunction on line 2. So we follow the top-down strategy for \vee which tells us how to get a sentence from a disjunction.

23.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	<i>Assume</i>
	2.	$(\sim A \vee \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	B	\wedge -Elim, 3
	5.	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> B \vdots $\sim B \rightarrow (\psi \wedge \sim \psi)$ $\sim A \rightarrow (\psi \wedge \sim \psi)$ $\psi \wedge \sim \psi$ </div>	<i>Assume</i>
	$n - 5.$	$\sim B \rightarrow (\psi \wedge \sim \psi)$	
	$n - 4.$	$\sim A \rightarrow (\psi \wedge \sim \psi)$	
	$n - 3.$	$\psi \wedge \sim \psi$	\vee -Elim, 2, $n - 5, n - 4$
	$n - 2.$	$B \rightarrow (\psi \wedge \sim \psi)$	\rightarrow -Intro, 5– $n - 3$
	$n - 1.$	$\sim B$	\sim -Intro, $n - 2$
	$n.$	$B \wedge \sim B$	\wedge -Intro, $n - 2, n - 1$

To get the conditionals on lines $n - 5$ and $n - 4$, we'll have to use \rightarrow -Intro. So the setup is:

24.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	<i>Assume</i>
	2.	$(\sim A \vee \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	B	\wedge -Elim, 3
	5.	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> B </div>	<i>Assume</i>
	6.	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> $\sim B$ \vdots $(\psi \wedge \sim \psi)$ </div>	<i>Assume</i>
	$m.$	$(\psi \wedge \sim \psi)$	
	$m + 1.$	$\sim B \rightarrow (\psi \wedge \sim \psi)$	\rightarrow -Intro, 6– m
	$m + 2.$	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> $\sim A$ </div>	<i>Assume</i>

			\vdots	
$n - 5.$			$(\psi \wedge \sim\psi)$	
$n - 4.$		$\sim A \rightarrow (\psi \wedge \sim\psi)$		$\rightarrow\text{-Intro}, m + 2 - n - 5$
$n - 3.$		$\psi \wedge \sim\psi$		$\vee\text{-Elim}, 2, m + 1, n - 4$
$n - 2.$	$B \rightarrow (\psi \wedge \sim\psi)$			$\rightarrow\text{-Intro}, 5 - n - 3$
$n - 1.$	$\sim B$			$\sim\text{-Intro}, n - 2$
$n.$	$B \wedge \sim B$			$\wedge\text{-Intro}, n - 2, n - 1$

Now we can decide what contradiction $(\psi \wedge \sim\psi)$ to aim for. If we choose $(B \wedge \sim B)$ again, then the first conditional will be easy. We will be able to get it by using $\wedge\text{-Intro}$ on lines 5 and 6.

25.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	Assume
	2.	$(\sim A \vee \sim B)$	$\wedge\text{-Elim}, 1$
	3.	$(A \wedge B)$	$\wedge\text{-Elim}, 1$
	4.	B	$\wedge\text{-Elim}, 3$
	5.	B	Assume
	6.	$\sim B$	Assume
	7.	$(B \wedge \sim B)$	$\wedge\text{-Intro}, 5, 6$
	8.	$\sim B \rightarrow (B \wedge \sim B)$	$\rightarrow\text{-Intro}, 6 - 7$
	9.	$\sim A$	Assume
		\vdots	
$n - 5.$		$(B \wedge \sim B)$	
$n - 4.$		$\sim A \rightarrow (B \wedge \sim B)$	$\rightarrow\text{-Intro}, 9 - n - 5$
$n - 3.$		$B \wedge \sim B$	$\vee\text{-Elim}, 2, 8, n - 4$
$n - 2.$	$B \rightarrow (B \wedge \sim B)$		$\rightarrow\text{-Intro}, 5 - n - 3$
$n - 1.$	$\sim B$		$\sim\text{-Intro}, n - 2$
$n.$	$B \wedge \sim B$		$\wedge\text{-Intro}, n - 2, n - 1$

This leaves us with just the second conditional, deriving $(B \wedge \sim B)$ from $\sim A$. At first glance this may seem impossible. We were able to derive $(B \wedge \sim B)$ from $\sim B$

because, given that we already had B , assuming $\sim B$ allowed us to use \wedge -Intro. But $\sim A$ obviously isn't sufficient to allow us to use \wedge -Intro, and there's no clear way to get what we need for \wedge -Intro, $\sim B$, from $\sim A$. So we won't be able to get $(B \wedge \sim B)$ by using \wedge -Intro.

Now so far we only have one strategy for getting a conjunction and that's to derive the conjuncts and use \wedge -Intro. But there's another strategy, not tied to any particular connective, which we can use here. This strategy is to use \sim -Elim. We will assume $\sim(B \wedge \sim B)$, derive a contradiction (any will do), and then use \rightarrow -Intro to get what we need. It should be reasonably clear that this strategy will work, since we obviously can derive the contradiction $(A \wedge \sim A)$.

26.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	Assume
	2.	$(\sim A \vee \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	B	\wedge -Elim, 3
	5.	B	Assume
	6.	$\sim B$	Assume
	7.	$(B \wedge \sim B)$	\wedge -Intro, 5,6
	8.	$\sim B \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 6–7
	9.	$\sim A$	Assume
	10.	$\sim(B \wedge \sim B)$	Assume
	11.	A	\wedge -Elim, 3
	12.	$A \wedge \sim A$	\wedge -Intro, 9,11
	13.	$\sim(B \wedge \sim B) \rightarrow (A \wedge \sim A)$	\rightarrow -Intro, 10–12
	14.	$(B \wedge \sim B)$	\sim -Elim, 13
	15.	$\sim A \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 9–14
	16.	$B \wedge \sim B$	\vee -Elim, 2,8,15
	17.	$B \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 5–16
	18.	$\sim B$	\sim -Intro, 17
	19.	$B \wedge \sim B$	\wedge -Intro, 4,18

Note that we didn't actually use the assumption $\sim(B \wedge \sim B)$ from line 10 in deriving the contradiction we end with on line 12, $A \wedge \sim A$. There is nothing wrong with this, since \rightarrow -Intro doesn't require that the sentence θ with which you started is actually used in deriving the sentence ψ with which you finished.

Now that we've derived a contradiction from $((\sim A \vee \sim B) \wedge (A \wedge B))$ we can finish the derivation by discharging the assumption with \rightarrow -Intro and then use \sim -Intro.

27.	1.	$((\sim A \vee \sim B) \wedge (A \wedge B))$	Assume
	2.	$(\sim A \vee \sim B)$	\wedge -Elim, 1
	3.	$(A \wedge B)$	\wedge -Elim, 1
	4.	B	\wedge -Elim, 3
	5.	B	Assume
	6.	$\sim B$	Assume
	7.	$(B \wedge \sim B)$	\wedge -Intro, 5,6
	8.	$\sim B \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 6–7
	9.	$\sim A$	Assume
	10.	$\sim(B \wedge \sim B)$	Assume
	11.	A	\wedge -Elim, 3
	12.	$A \wedge \sim A$	\wedge -Intro, 9,11
	13.	$\sim(B \wedge \sim B) \rightarrow (A \wedge \sim A)$	\rightarrow -Intro, 10–12
	14.	$(B \wedge \sim B)$	\sim -Elim, 13
	15.	$\sim A \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 9–14
	16.	$B \wedge \sim B$	\vee -Elim, 2,8,15
	17.	$B \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 5–16
	18.	$\sim B$	\sim -Intro, 17
	19.	$B \wedge \sim B$	\wedge -Intro, 4,18
	20.	$((\sim A \vee \sim B) \wedge (A \wedge B)) \rightarrow (B \wedge \sim B)$	\rightarrow -Intro, 1–19
	21.	$\sim((\sim A \vee \sim B) \wedge (A \wedge B))$	\sim -Intro, 20

Proof by Contradiction A general strategy involving negation was used in derivation 27, on the current page. This strategy formalizes an informal proof method often called proof by contradiction. In proof by contradiction, one proves that some sentence ϕ is true by assuming it's false and showing that a contradiction follows. The corresponding strategy in SD is:

Proof by Contradiction: If you want to get some sentence ϕ , then first derive $\sim\phi \rightarrow (\psi \wedge \sim\psi)$ and then use \sim -Elim to get ϕ from this conditional.

This strategy has the virtue that if you can derive ϕ (without any assumptions), then you will be able to derive a contradiction $(\psi \wedge \sim\psi)$ from $\sim\phi$ as an assumption. In other words, the strategy will always work. But despite this, there are usually much faster and better ways to derive a sentence. For example, write a derivation of $C \wedge B$ from $B \wedge C$ using proof by contradiction and then compare it to derivation 7, on page 132. There are some cases where proof by contradiction is the *only* strategy that will work. So there are two cases where it is a good idea to use the strategy: cases

where all other available strategies haven't worked (or where there aren't any other available strategies), and cases where you can see a straightforward way to use it.

Top-down and Bottom-up Finally, the reader should reflect on the general method we have been using. In slogan form, the method goes: work top-down *and* bottom-up. When writing derivation it's important to work not only from the assumptions, seeing how you can move down from them to the conclusion using the rules, but also to work up from the sentence you're trying to derive, looking for the different ways you can get there. Hence the grouping of strategies into top-down and bottom-up. The top-down strategies help guide what moves you can make as you work from the assumptions to the conclusion, while the bottom-up strategies help see what different paths there are to get to that conclusion.

As an aside, the top-down and bottom-up method is something useful outside of writing formal derivations. While most arguments, including those found in philosophy, mathematical proofs, and formal derivations, are presented as a series of steps from premises to conclusion, they are seldom devised that way. Usually a mathematician starts with a conjecture and tries to work "up" from it to results that have already been proven. Rarely does one from the start know what premises will be needed to prove a conjecture. Something similar goes for philosophy and other disciplines that rely on argument. So, the reader can think of the top-down, bottom-up method used in derivations as a reflection of how informal arguments and proofs are constructed in philosophy, mathematics, and other disciplines.

6.3 Shortcut Rules for SD

6.3.1 Standard Shortcut Rules

From derivation 27, on the previous page, we can see that derivations in SD of even simple looking sentences can be long and involve roundabout strategies. This is the main reason why we want to introduce shortcut rules. Shortcut rules can be thought of as ways of cutting out parts of derivations that we find ourselves doing repeatedly. The basic rules of SD plus the shortcut rules (both those in table 6.36 on the next page and table 6.37 on page 150) make up the derivation system which we call SD⁺.

For example, consider lines 10–14 of proof 27, on the previous page. Here we have two sentences, A and $\sim A$, and we used them to derive the sentence $B \wedge \sim B$. We can rewrite the relevant parts of the proof here, putting them in a less idiosyncratic order:

28.	1.	A	
	2.	$\sim A$	
	3.	$\sim(B \wedge \sim B)$	<i>Assume</i>
	4.	$A \wedge \sim A$	\wedge -Intro, 1,2
	5.	$\sim(B \wedge \sim B) \rightarrow (A \wedge \sim A)$	\rightarrow -Intro, 3–4
	6.	$B \wedge \sim B$	\sim -Elim

It should be clear from looking at this derivation that we can replace $B \wedge \sim B$ with any sentence ψ and the string of sentences that results from this replacement will also be a derivation. It should also be clear that we can replace A and $\sim A$ with any pair ϕ and $\sim\phi$ and the resulting string of sentences will be a derivation. That is,

29.

1.	ϕ	
2.	$\sim\phi$	
3.	$\sim\psi$	<i>Assume</i>
4.	$\phi \wedge \sim\phi$	\wedge -Intro, 1,2
5.	$\sim\psi \rightarrow (\phi \wedge \sim\phi)$	\rightarrow -Intro, 3–4
6.	ψ	\sim -Elim

will be a derivation of ψ for any sentences ϕ and ψ . More importantly, any time we're doing a derivation and we have two lines, one with a sentence ϕ and the other with its negation $\sim\phi$, we could insert a derivation of just this form to get a sentence ψ . So we can introduce a new rule which says that given two sentences ϕ and $\sim\phi$, we can add any sentence ψ . We call this new rule *Any Contradiction*, or *A.C.* for short. The key feature of this rule is that in any derivation where we use the rule, we could have gotten the same results without it. All we'd have to do is insert the appropriate instance of 29 into the derivation. (We leave it to the reader to rewrite derivation 27 on page 147 using *A.C.*)

Name	Given	May Add
<i>M.T.</i>	$\phi \rightarrow \theta, \sim\theta$	$\sim\phi$
<i>D.S.</i>	$\phi_1 \vee \dots \vee \phi_i \vee \dots \vee \phi_n, \sim\phi_i$	$\phi_1 \vee \dots \vee \phi_{i-1} \vee \phi_{i+1} \vee \dots \vee \phi_n$
	$\phi_1 \vee \dots \vee \sim\phi_i \vee \dots \vee \phi_n, \phi_i$	$\phi_1 \vee \dots \vee \phi_{i-1} \vee \phi_{i+1} \vee \dots \vee \phi_n$
<i>A.C.</i>	$\phi, \sim\phi$	ψ
\sim/\leftrightarrow -Intro	$\phi \leftrightarrow \psi$	$\sim\phi \leftrightarrow \sim\psi$
<i>Ext.</i> \wedge -Elim	$\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n$	Conjunction of any subset of the conjuncts

Table 6.36: Standard Shortcut Rules for SD

The idea will be the same for all the shortcut rules we'll introduce here. These come in two types, standard and exchange, and are listed in table 6.36, on this page and 6.37, on the next page. The idea is, (i) for each shortcut rule R , for given any

application of R we can derive, using only the basic rules, the sentence ϕ (which R allows us to write down) from the sentences ψ_1, \dots, ψ_n to which we applied R . And, (ii) in general, anything we can derive using a rule, any application of which can be derived using only basic rules, can itself be derived using only basic rules. So, (iii) anything we can derive using the basic rules of SD and any of the shortcut rules can be derived from just the basic rules alone. We have restated (ii) more precisely below as theorem 6.3 (on the next page), (i) as theorem 6.4 (on page 152), and (iii) as theorem 6.5 (on page 152). Exchange rules are short cut rules that work in both directions.

Name	Given	May Add
<i>DeM</i>	$\sim(\phi_1 \wedge \dots \wedge \phi_n)$	$\sim\phi_1 \vee \dots \vee \sim\phi_n$
	$\sim\phi_1 \vee \dots \vee \sim\phi_n$	$\sim(\phi_1 \wedge \dots \wedge \phi_n)$
	$\sim(\phi_1 \vee \dots \vee \phi_n)$	$\sim\phi_1 \wedge \dots \wedge \sim\phi_n$
	$\sim\phi_1 \wedge \dots \wedge \sim\phi_n$	$\sim(\phi_1 \vee \dots \vee \phi_n)$
<i>$\sim\sim$-Elim</i>	$\sim\sim\phi$	ϕ
<i>$\sim\sim$-Intro</i>	ϕ	$\sim\sim\phi$
<i>\rightarrow/\vee-Exch.</i>	$\phi \rightarrow \theta$	$\sim\phi \vee \theta$
	$\sim\phi \vee \theta$	$\phi \rightarrow \theta$
<i>Contraposition</i>	$\phi \rightarrow \theta$	$\sim\theta \rightarrow \sim\phi$
	$\sim\theta \rightarrow \sim\phi$	$\phi \rightarrow \theta$
<i>\sim/\rightarrow-Exch.</i>	$\sim(\phi \rightarrow \theta)$	$\phi \wedge \sim\theta$
	$\phi \wedge \sim\theta$	$\sim(\phi \rightarrow \theta)$

Table 6.37: Exchange Short-Cut Rules for SD

Continued next Page

Continued from Previous Page

Name	Given	May Add
<i>Distribution</i>	$\theta \wedge (\phi_1 \vee \dots \vee \phi_n)$	$(\theta \wedge \phi_1) \vee \dots \vee (\theta \wedge \phi_n)$
	$(\theta \wedge \phi_1) \vee \dots \vee (\theta \wedge \phi_n)$	$\theta \wedge (\phi_1 \vee \dots \vee \phi_n)$
	$(\phi_1 \vee \dots \vee \phi_n) \wedge \theta$	$(\phi_1 \wedge \theta) \vee \dots \vee (\phi_n \wedge \theta)$
	$(\phi_1 \wedge \theta) \vee \dots \vee (\phi_n \wedge \theta)$	$(\phi_1 \vee \dots \vee \phi_n) \wedge \theta$
	$\theta \vee (\phi_1 \wedge \dots \wedge \phi_n)$	$(\theta \vee \phi_1) \wedge \dots \wedge (\theta \vee \phi_n)$
	$(\theta \vee \phi_1) \wedge \dots \wedge (\theta \vee \phi_n)$	$\theta \vee (\phi_1 \wedge \dots \wedge \phi_n)$
	$(\phi_1 \wedge \dots \wedge \phi_n) \vee \theta$	$(\phi_1 \vee \theta) \wedge \dots \wedge (\phi_n \vee \theta)$
	$(\phi_1 \vee \theta) \wedge \dots \wedge (\phi_n \vee \theta)$	$(\phi_1 \wedge \dots \wedge \phi_n) \vee \theta$

Table 6.37: Exchange Short-Cut Rules for SD

Theorem 6.3 (on the current page) is a general claim that doesn't require a different proof for each new rule we introduce. The proof for it fills out the quick argument given when *A.C.* was introduced. There we argued that since any application of the rule can be derived using just the basic rules, we can eliminate that application of the rule from the proof by cutting and pasting the derivation of that application into the original proof.

Definition 6.2. We say that every application of a rule R_1 is derivable using the rules R_2, \dots, R_p and the basic rules of SD iff for all SL sentences ϕ_1, \dots, ϕ_m and ψ , if R_1 sanctions writing down ψ when applied to ϕ_1, \dots, ϕ_m on previous unboxed lines, then ψ can be derived from ϕ_1, \dots, ϕ_m using only rules R_2, \dots, R_p and the basic rules of SD.

Theorem 6.3. For all SL sentences $\theta_1, \dots, \theta_n, \delta$ and rules R_1, \dots, R_p , if

- (1) δ can be derived from $\theta_1, \dots, \theta_n$ using rules R_1, \dots, R_p and the basic rules of SD, and
- (2) every application of a rule R_1 is derivable using the rules R_2, \dots, R_p and the basic rules of SD,

then δ can be derived from $\theta_1, \dots, \theta_n$ using only rules R_2, \dots, R_p and the basic rules of SD.

Proof: Call the original derivation of δ from $\theta_1, \dots, \theta_n$ using rules R_1, \dots, R_p derivation D_1 . Say the first use of R_1 happens in D_1 on line q . Say in that case sentence ψ was written down (on line q) and the application of the rule used previous lines r_1, \dots, r_m

with sentences ϕ_1, \dots, ϕ_m , respectively, on them. By assumption, ψ can be derived from sentences ϕ_1, \dots, ϕ_m using only R_2, \dots, R_p and the basic rules of SD. Call this derivation D^* and assume ϕ_1, \dots, ϕ_m are, respectively, on lines 1 through m in D^* . Assume there are s more lines (call these the middle lines of D^*), and finally on line $m + s + 1$ of D^* is ψ .

Now in between lines $q - 1$ and q of D_1 insert s new lines. On these lines put the appropriate sentence from the s middle line of D^* . (So, put the sentence on the first middle line of D^* on the first new line inserted into D_1 , the second on the second, etc.) Write the same rules as justifications on these new lines as were on the middle lines of D^* and for each justification if t was the number of a line cited by that justification in D^* , cite line number r_t if $t \leq m$ and cite line number $(t - m) + (q - 1)$ if $t > m$. Next, on the line originally numbered q (now numbered $q + s$) erase rule R_1 as the justification and whatever line numbers t were cited and in place of that put whatever rule was used to justify the last line of D^* , citing instead lines r_t if $t \leq m$ and lines $(t - m) + (q - 1)$ if $t > m$. Finally, on all the lines of D_1 that were originally numbered q or higher (and now are numbered $q + s$ and higher), if the justification cites line t for $t < q$, do nothing. If it cites line t for $t \geq q$, replace t with $t + s$.

Note that we've now produced a derivation D_2 of δ from $\theta_1, \dots, \theta_n$ that has one less application of R_1 than D_1 had. Now if there is an application of rule R_1 in D_2 , repeat exactly this procedure for D_2 . Repeating the procedure will lead to a derivation D_3 with one less application of R_1 than D_2 had. We can continue this, producing some series D_1, D_2, \dots, D_l of derivations which will eventually end in a derivation D_l that has no applications of R_1 . (This procedure must end, since there could only have been a finite number of applications of R_1 in D_1 .) Thus, D_l will be a derivation of δ from $\theta_1, \dots, \theta_n$ that uses only rules R_2, \dots, R_p and the basic rules of SD. ■

Theorem 6.4. *For all standard and exchange shortcut rules R (see tables 6.36 and 6.37), every application of R is derivable using the basic rules of SD.*

Proof: See the discussion immediately following the proof of theorem 6.5. ■

Theorem 6.5. *Shortcut Rule Elimination Theorem: For all SL sentences ϕ_1, \dots, ϕ_m and ψ , if ψ can be derived from ϕ_1, \dots, ϕ_m in SD^+ (that is, using the basic rules of SD and any of the standard and exchange shortcut rules), then ψ can be derived from ϕ_1, \dots, ϕ_m in SD (that is, using only the basic rules).*

Proof: Assume that ψ can be derived from ϕ_1, \dots, ϕ_m using the basic rules of SD and the standard and exchange shortcut rules. Consider any of the shortcut rules, say $M.T.$ Let R_1 be $M.T.$ and rules R_2 through R_{25} be the other standard and exchange rules. By assumption, condition (1) in theorem 6.3 (on the previous page) holds for these sentences, while by theorem 6.4 (on the current page) condition (2) in theorem 6.3 holds for $M.T.$ So, it follows from theorem 6.3 that ψ can be derived from ϕ_1, \dots, ϕ_m using the basic rules of SD and all the standard and exchange shortcut rules besides

M.T. By reapplying theorem 6.3 in just the same way to all the standard and exchange shortcut rules, we get that ψ can be derived from ϕ_1, \dots, ϕ_m using only the basic rules of SD. (That is, we reapply theorem 6.3 twenty four more times, each time showing that another shortcut rule wasn't needed.) ■

Unlike theorem 6.3, for theorem 6.4 we need a separate argument for each shortcut rule (both standard and exchange). We've already done one rule, *Any Contradiction*. Our argument in this case was that any application of the rule will involve writing some sentence ψ on a new line from sentences ϕ and $\sim\phi$. But whatever sentences ψ and ϕ we pick, if we substitute them into 29, the result will be a derivation of ψ from ϕ and $\sim\phi$.

Before continuing, it's important to note that, strictly speaking, 29 is *not* a derivation. This is because a derivation, as we've defined it, is a series of SL sentences. The strings of symbols on each line of 29 are not SL sentences because they contain MathEnglish variables for SL sentences. Instead, they are sentence schemas. But, as we've done in 29, nothing stops us from treating sentence schemas like the ones in 29 as SL sentences and applying rules to them. The key fact—why this is useful—is that what we get when we do this becomes a derivation whenever we substitute SL sentences in for the MathEnglish variables. For this reason we'll call these *derivation schemas* instead of derivations.

Returning to theorem 6.4 (on the previous page), we can handle the other rules in just the same way we handled *Any Contradiction*. For example, if we write a derivation schema of $\sim\phi$ from $\phi \rightarrow \theta$ and $\sim\theta$ using only basic rules of SD, then this will be sufficient to show that any application of the rule *M.T.* (*Modus Tollens*) can be derived using only basic rules of SD.⁵ (Recall def. 6.2, on page 151: Saying that an application of a rule can be derived using only basic rules is a shorthand way of saying that the sentence written down on a new line, in some application of the rule, can be derived using only basic rules from the sentences to which the rule was applied.) So, in order to complete the proof for theorem 6.4, we need to write derivation schemas for all the standard and exchange rules (for all the rules in tables 6.36 and 6.37). That is, for each rule we need to write a derivation schema that has the given schemas of the rule as premises and the may-add schema of the rule as conclusion.

Note that once we have shown that a shortcut rule can be eliminated from a proof (i.e., once we've shown that theorem 6.4, holds at least for that rule), then we can use that shortcut rule in derivation schemas for new shortcut rules we haven't yet shown can be eliminated. If we write a derivation schema for some new shortcut rule we're trying to show can be eliminated and that schema uses a previous shortcut rule, then any derivation got from the schema will contain an application of the previous rule.

⁵For those keeping track of the use/mention distinction, here we have mentioned the MathEnglish variables ' ϕ ' and ' θ ' as well as the strings of symbols ' $\sim\phi$ ', ' $\phi \rightarrow \theta$ ', and ' $\sim\theta$ '. So, strictly speaking, we should have put them all in quotes. This is different how we normally use these symbols, since we're normally actually using them (as variables) instead of mentioning them (as the objects of derivation schemas).

But we already know that these applications of the previous rule can be eliminated, so that's not a problem.

Most of the derivation schemas for the shortcut rules (both standard and exchange) are left to the reader as exercises. (See section 6.5.2.) There is one complication though. We cannot actually write a single derivation schema for *D.S.* (*Disjunctive Syllogism*), *DeM* (*DeMorgans*), or *Distribution*. This is because these rules mention arbitrarily long conjunctions and disjunctions and we can only write a derivation schema involving conjunctions and disjunctions of definite, fixed (and finite) length. A less rigorous option is to write a few derivation schemas for *D.S.*, *DeM*, and *Distribution* for the cases when the conjunctions and disjunctions are small (say 2- or 3-place) and convince ourselves that we can keep writing similar schemas no matter how large the conjunctions and disjunctions get. A more rigorous option is to write the derivation schema for the 2-place case and then use mathematical induction on the length of the conjunctions and disjunctions to show derivation schemas can be written for all lengths. In the exercises, in section 6.5.2, we only ask the reader to write the derivation schemas for these three rules for the cases where the conjunctions and disjunctions are 2-place.

Here we write the derivation schema needed for the last of the four *DeMorgans* rules in table 6.37 (on page 150), assuming that the conjunction and disjunction are only 2-place. So we need to derive $\sim(\phi \vee \theta)$ from $\sim\phi \wedge \sim\theta$. The sentence we want is a negation, so we use our basic bottom-up strategy for negation:

30.	1.	$\sim\phi \wedge \sim\theta$	<i>Assume</i>
	2.	$(\phi \vee \theta)$	<i>Assume</i>
		\vdots	
	$n - 2.$	$\psi \wedge \sim\psi$	
	$n - 1.$	$(\phi \vee \theta) \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, $2 - n - 2$
	$n.$	$\sim(\phi \vee \theta)$	\sim -Intro, $n - 1$

As always with \sim -Intro, we need some contradiction $\psi \wedge \sim\psi$. Before we had to be careful about setting up the right contradiction (recall derivation 27). But now that we have rule *A.C.*, we don't need to be so careful. So long as we can get a contradiction, we can always use *A.C.* to get whatever other contradiction we need to make the derivation work.

Returning to the proof, we need to work top-down from the disjunction on line 2 to get to a contradiction. So we use the basic strategy:

31.	1.	$\sim\phi \wedge \sim\theta$	<i>Assume</i>
	2.	$(\phi \vee \theta)$	<i>Assume</i>
		\vdots	
	$n - 4.$	$\phi \rightarrow (\psi \wedge \sim\psi)$	
	$n - 3.$	$\theta \rightarrow (\psi \wedge \sim\psi)$	
	$n - 2.$	$\psi \wedge \sim\psi$	\vee -Elim, 2, $n - 4, n - 3$
	$n - 1.$	$(\phi \vee \theta) \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, 2– $n - 2$
	$n.$	$\sim(\phi \vee \theta)$	\sim -Intro, $n - 1$

And from here we need to work bottom-up from the conditionals on lines $n - 4$ and $n - 3$.

32.	1.	$\sim\phi \wedge \sim\theta$	<i>Assume</i>
	2.	$(\phi \vee \theta)$	<i>Assume</i>
	3.	ϕ	<i>Assume</i>
		\vdots	
	$m.$	$(\psi \wedge \sim\psi)$	
	$m + 1.$	$\phi \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, 3– m
	$m + 2.$	θ	<i>Assume</i>
		\vdots	
	$n - 4.$	$(\psi \wedge \sim\psi)$	
	$n - 3.$	$\theta \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, $m + 2$ – $n - 4$
	$n - 2.$	$\psi \wedge \sim\psi$	\vee -Elim, 2, $m + 1, n - 3$
	$n - 1.$	$(\phi \vee \theta) \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, 2– $n - 2$
	$n.$	$\sim(\phi \vee \theta)$	\sim -Intro, $n - 1$

And now we can finish the proof by working top-down, breaking apart the conjunction on line 1 and using *A.C.*

33.	1.	$\sim\phi \wedge \sim\theta$	<i>Assume</i>
	2.	$(\phi \vee \theta)$	<i>Assume</i>
	3.	ϕ	<i>Assume</i>
	4.	$\sim\phi$	\wedge -Elim, 1
	5.	$\sim\phi \wedge \phi$	\wedge -Intro, 3,4
	6.	$(\psi \wedge \sim\psi)$	A.C., 5
	7.	$\phi \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, 3–6
	8.	θ	<i>Assume</i>
	9.	$\sim\theta$	\wedge -Elim, 1
	10.	$\sim\theta \wedge \theta$	\wedge -Intro, 8,9
	11.	$(\psi \wedge \sim\psi)$	A.C., 10
	12.	$\theta \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, 8–11
	13.	$\psi \wedge \sim\psi$	\vee -Elim, 2,7,12
	14.	$(\phi \vee \theta) \rightarrow (\psi \wedge \sim\psi)$	\rightarrow -Intro, 2–13
	15.	$\sim(\phi \vee \theta)$	\sim -Intro, 14

And so we now have a derivation schema showing how to derive a sentence of the form $\sim(\phi \vee \theta)$ from one of the form $\sim\phi \wedge \sim\theta$. Note that we could have slightly shortened the derivation schema by doing \rightarrow -Intro on lines 3–5 instead of 3–6, using $\sim\phi \wedge \phi$ as our contradiction instead of $\phi \rightarrow (\psi \wedge \sim\psi)$. In this case would have then used A.C. on line 11 to get $\sim\phi \wedge \phi$. Similarly, we could have also used $\sim\theta \wedge \theta$ as our contradiction. We leave it to the reader to show what slight modifications would be needed to the derivation schema in this case.

Finally, we should note that every application of every shortcut rule (whether a standard or exchange shortcut rule) is truth-preserving. (Recall def. 7.1, on page 181.) Although you might be tempted to try to prove this as a corollary to theorem 7.4, on page 181, and theorem 6.4, on page 152, there's no easy way to get from (i) the claim that every application of every basic rule of SD is truth-preserving and (ii) the claim that every application of every rule of SD^+ is derivable in SD, to the further claim that every application of every rule of SD^+ is truth-preserving. Instead, the easiest way to prove this result more or less follows the lines of the proof for theorem 7.4 (on page 181).

Theorem 6.6. *Every application of every rule of SD^+ , including both standard and exchange shortcut rules, is truth-preserving.*

Proof: Adjusting the proof of theorem 7.4 to work for the rules of SD^+ is left for the reader. The key is extending the truth-preservation lemma mentioned there to the

rules of SD^+ . We asked the reader to show this in exercises 2.8.8 (on page 56) and 2.8.9 (on page 57). ■

6.3.2 Exchange Shortcut Rules

Recall our restriction in section 6.2.4 on the basic rules of SD , which said that a rule only sanctions writing down a sentence if the connectives it mentions are the main connectives of sentences on the lines to which it's applied. We put this restriction in place because not every application of the basic rules is truth-preserving without it. But with the restriction in place, every application of the basic rules becomes truth-preserving (theorem 7.4, on page 181). When we introduced the shortcut rules we carried over the restriction because, again, without it there are some shortcut rules with nontruth-preserving applications. But for some of our shortcut rules, the exchange shortcut rules (table 6.37, on page 150), every application is truth-preserving even without the restriction. To be explicit, for the exchange shortcut rules we can use the following definition of sanctioning (while still using def. 7.3 on page 181 for the basic rules and the standard exchange rules):

Definition 6.7. An exchange shortcut rule R from SD^+ , applied to a line with sentence ψ , *sanctions* writing down sentence ψ^* iff

- (1) there is some substitution of SL sentences that, for the given schema of R , results in a sentence ϕ and, for the may-add schema, results in a sentence ϕ^* ,
- (2) ϕ is a subsentence of ψ , and
- (3) ψ^* is the SL sentence you get when you replace one instance (token) of ϕ with an instance (token) of ϕ^* in ψ .

Note that right below derivation 35 (on page 159) is a detailed example of how the definition works in practice. With a moments thought it should be clear that if ϕ actually is ψ , then (recalling that all the exchange rules only have one given schema) this definition lines up with the original definition 7.3 (on page 181). This is just what we would expect.

Theorem 6.8. *Every application of every exchange shortcut rule from SD^+ is truth-preserving, even if we extend the notion of sanctioning for them with definition 6.7.*

Note that this result was not already stated in theorem 6.6 (on the previous page). Although theorem 6.6 says that every application of every rule of SD^+ is truth-preserving, truth-preservation is defined in terms of sanctioning (def. 7.1). And, in theorem 6.6 it was assumed that the exchange shortcut rules only sanctioned writing down a sentence if they were applied to whole sentences on lines. But now we're extending the notion of sanctioning for the exchange shortcut rules with definition 6.7. Theorem 6.8 notes that even if we do this, the exchange shortcut rules are still truth-preserving.

We will use theorem 2.70 (on page 46) and the following theorem to prove theorem 6.8. This theorem will play the same role as the truth-preservation lemma from the

proof of theorem 7.4 (on page 181). Theorem 2.70 is actually stronger than what we need, but since we already have it we will use it.

Theorem 6.9. *For all exchange shortcut rules R from SD^+ , if ϕ and ϕ^* are the sentences you get after substituting SL sentences into the given and may-add schemas of R , respectively, then ϕ and ϕ^* are truth functionally equivalent.*

Proof: This theorem follows immediately from what the reader showed in exercises 2.8.9 (on page 57). ■

Proof of Thm. 6.8: Consider some arbitrary application of some exchange shortcut rule R from SD^+ . Say that in this application R is applied to sentence ψ and permits, or sanctions, you to write down ψ^* . By definition 6.7 (on the previous page), (i) there is some substitution of SL sentences that, for the given schema of R , results in a sentence ϕ and, for the may-add schema, results in a sentence ϕ^* , (ii) ϕ is a subsentence of ψ , and (iii) ψ^* is the SL sentence you get when you replace one instance (token) of ϕ with an instance (token) of ϕ^* in ψ . From (i) and theorem 6.9, ϕ and ϕ^* are truth functionally equivalent. So, from theorem 2.70 (on page 46) it follows that ψ and ψ^* are also truth functionally equivalent. From the definition 2.50 (on page 35) of truth functional equivalence it follows that $\psi \models \psi^*$. So, by definition 7.1 (on page 181), this application is truth-preserving. ■

The intuitive idea behind definition 6.7 (on the previous page) is that the exchange shortcut rules can be applied to subsentences on lines (while the basic rules and standard shortcut rules can only be applied to whole sentences on lines). Some examples will hopefully make things more clear. Recall derivation 27, on page 147, which showed that $\vdash \sim((\sim A \vee \sim B) \wedge (A \wedge B))$. First, consider how we might rewrite this proof using *DeMorgans*, in addition to the basic rules of SD , but only applying it to the whole sentence on a line.

- | | | | |
|-----|----|---|---------------------------|
| 34. | 1. | $(\sim A \vee \sim B) \wedge (A \wedge B)$ | <i>Assume</i> |
| | 2. | $(\sim A \vee \sim B)$ | \wedge -Elim, 1 |
| | 3. | $(A \wedge B)$ | \wedge -Elim, 1 |
| | 4. | $\sim(A \wedge B)$ | <i>DeM</i> , 2 |
| | 5. | $(A \wedge B) \wedge \sim(A \wedge B)$ | \wedge -Intro, 3,4 |
| | 6. | $((\sim A \vee \sim B) \wedge (A \wedge B)) \rightarrow ((A \wedge B) \wedge \sim(A \wedge B))$ | \rightarrow -Intro, 1–5 |
| | 7. | $\sim((\sim A \vee \sim B) \wedge (A \wedge B))$ | \sim -Intro, 6 |

Obviously this is much shorter than 27, and much shorter than 27 would be even if we rewrote it using *A.C.* (as we suggested the reader do above). Here we have applied *DeMorgans* to line 2. But if we allow ourselves to apply *DeMorgans* to subsentences

on lines in addition to whole sentences, then the proof can be made even shorter.

- 35.
- | | | |
|----|---|---------------------------|
| 1. | $(\sim A \vee \sim B) \wedge (A \wedge B)$ | <i>Assume</i> |
| 2. | $\sim(A \wedge B) \wedge (A \wedge B)$ | <i>DeM, 1</i> |
| 3. | $((\sim A \vee \sim B) \wedge (A \wedge B)) \rightarrow (\sim(A \wedge B) \wedge (A \wedge B))$ | \rightarrow -Intro, 1–2 |
| 4. | $\sim((\sim A \vee \sim B) \wedge (A \wedge B))$ | \sim -Intro, 3 |

Unlike in 34, here we did not need to break line 1 apart into its conjuncts. We simply applied *DeMorgans* to the first conjunct of line 1 and wrote the result down as line 2. To see how definition 6.7 (on page 157) captures this intuitive idea, consider how we would show, directly from the definition, that *DeMorgans* sanctions writing $\psi^* = \sim(A \wedge B) \wedge (A \wedge B)$ on line 2 when applied to $\psi = (\sim A \vee \sim B) \wedge (A \wedge B)$ on line 1. The given schema for *DeMorgans* relevant to line 1 is $\sim\phi_1 \vee \sim\phi_2$, while the relevant may-add schema is $\sim(\phi_1 \wedge \phi_2)$. The substitution we want for clause (1) of definition 6.7 is $\phi_1 = A$ and $\phi_2 = B$. This substitution results in $\phi = (\sim A \vee \sim B)$ and $\phi^* = \sim(A \wedge B)$. In line with clause (2) of the definition, $\phi = (\sim A \vee \sim B)$ is a subsentence of $\psi = (\sim A \vee \sim B) \wedge (A \wedge B)$. And, in line with clause (3), $\psi^* = \sim(A \wedge B) \wedge (A \wedge B)$ is the SL sentence you get when you replace one instance (token) of $\phi = (\sim A \vee \sim B)$ with an instance (token) of $\phi^* = \sim(A \wedge B)$ in $\psi = (\sim A \vee \sim B) \wedge (A \wedge B)$.

We have shown that the exchange shortcut rules, more liberally allowed to be applied to subsentences on a line, still have truth-preserving applications (Thm. 6.8). In the last section, 6.3.1 (on page 148), we showed that anything we can derive using the standard and exchange shortcut rules from SD^+ can be derived using only the basic rules of SD (Thm. 6.5, on page 152). But to prove this we used theorem 6.3 (on page 151) and theorem 6.4 and we need to think about how more liberally allowing the exchange shortcut rules to be applied to subsentences on a line affects the proofs of these two theorems. That is, if we want to show that theorem 6.5 still holds, we need to show that the theorems we used to prove it still hold.

It should not be hard to see that the change from definition 7.3 (on page 181) to 6.7 (on page 157), i.e. the move to more liberal applications of exchange shortcut rules, does not affect the proof of 6.3 (on page 151). But it does affect the proof of 6.4. We proved this theorem (or, rather, asked the reader to prove most of the cases) by giving, for each rule R , a derivation schema using only the basic rules of SD that has the given schemas of R as the premises and the may-add schema of R as the conclusion (as the last line). This was sufficient to show that every application of the rules of SD^+ , if restricted to whole sentences on lines, is derivable from the basic rules of SD because if the rules are only being applied to whole sentences, then these derivation schema will yield derivations that use only basic rules for every application. But this will not work if we're more liberally allowing the exchange rules to be applied to subsentences on a line.

For example, consider derivation schema 33 (on page 156) for *DeMorgans*. Say that we have the sentence $A \vee (\sim B \wedge \sim C)$ on some line of a derivation on which we're working and we want to apply *DeMorgans* to the right disjunct. Under the more liberal definition 6.7 (on page 157) we can do this, and *DeMorgans* will permit, or sanction, us to write down $A \vee \sim(B \vee C)$. But it should be clear that substituting $\phi = B$ and $\theta = C$ into derivation schema 33 will not result in a derivation of $A \vee \sim(B \vee C)$ from $A \vee (\sim B \wedge \sim C)$.

To show that theorem 6.4 still holds for our more liberal use of exchange shortcut rules we will rely on two facts. First, definition 6.7 ensures that if an exchange shortcut rule, applied to a sentence ψ , sanctions writing down ψ^* , then there's a specific relationship between ψ and ψ^* . Specifically, there are two sentences ϕ and ϕ^* , got by substituting sentences into the given and may-add schemas of the rules, and ψ^* is ψ with ϕ replaced with ϕ^* . The second fact is that these sentences ϕ and ϕ^* are always *provably equivalent*, or as we might say *derivationally equivalent*.⁶ (This is proved in exercise 6.5.2, on page 176.)

Definition 6.10. Two sentences of SL are *provably equivalent* iff one of the following two equivalent conditions holds:

- (1) both $\phi \vdash \psi$ and $\psi \vdash \phi$, or
- (2) $\vdash \phi \leftrightarrow \psi$.

These two facts, along with the following theorem,⁷ are enough to show that theorem 6.4 still holds for our more liberal use of exchange shortcut rules. (The reader should make sure they are convinced of that.)

Theorem 6.11. Restricted Replacement Theorem for SD: *For all sentences ψ of SL: if*

- (1) ϕ and ϕ^* are SL sentences such that $\phi \vdash \phi^*$ and $\phi^* \vdash \phi$, and
- (2) if ϕ is a subsentence of ψ , then ψ^* is the SL sentence you get when you replace one instance (token) of ϕ with an instance (token) of ϕ^* , and ψ^* is ψ if not,

then ψ^ can be derived from ψ using only the basic rules of SD, i.e. $\psi \vdash \psi^*$.*

Proof: Assume that ϕ and ϕ^* are two SL sentences such that $\phi \vdash \phi^*$ and $\phi^* \vdash \phi$.

⁶Compare this definition with definition 6.19 (on page 172), which generalizes it for formulas of QL.

⁷Compare this theorem to theorem 6.17 (on page 172), which is a stronger version of it generalized to QD. Note that the Restricted Replacement Theorem for SD follows immediately from the generalized QD version (but we provide a separate proof here). Also note that the proof of theorem 6.17 uses the One-step Replacement Lemmas (Thm. 6.20, on page 173), and the proofs for these involve constructing derivation schemas. The proof given here for the Restricted Replacement Theorem for SD also involves the construction of derivation schemas, but does so by giving instructions in the inheritance step for how to write the relevant derivations instead of explicitly writing them out in a separate lemma.

Base Step: If ψ is atomic, then it's just a sentence letter. So, if ϕ is a subsentence of ψ , it itself must just be that same sentence letter. So, ϕ is the same sentence as ψ , and ψ^* is the same as ϕ^* . Since $\phi \vdash \phi^*$, it follows immediately that $\psi \vdash \psi^*$.

Inheritance Step: For the recursive hypothesis, assume that the theorem holds for the sentences $\theta, \theta_1, \dots, \theta_n, \delta$.

Conjunction: Assume that ψ is the conjunction $\theta_1 \wedge \dots \wedge \theta_n$. Either ϕ is not a subsentence of ψ , it's a subsentence of ψ but not the same as ψ , or is the same as ψ . If it's not a subsentence of ψ or it's the same as ψ , then just as in the base step it follows immediately that $\psi \vdash \psi^*$.

So assume that ϕ is a subsentence of ψ but not the same as it. Then ϕ is a subsentence of one of the conjuncts θ_i of ψ and ψ^* is the conjunction $\theta_1 \wedge \dots \wedge \theta_i^* \wedge \dots \wedge \theta_n$. By the recursive hypothesis, $\theta_i \vdash \theta_i^*$. It should be clear that by using \wedge -Elim we have that $\psi \vdash \theta_1, \dots, \psi \vdash \theta_i, \dots, \psi \vdash \theta_n$. By transitivity, $\psi \vdash \theta_i^*$. And, it should be clear that if each of $\theta_1, \dots, \theta_i^*, \dots, \theta_n$ can be derived from ψ , then by using \wedge -Intro we can derive their conjunction $\theta_1 \wedge \dots \wedge \theta_i^* \wedge \dots \wedge \theta_n$ from ψ . But this conjunction just is ψ^* , so $\psi \vdash \psi^*$.

Disjunction: Assume that ψ is the disjunction $\theta_1 \vee \dots \vee \theta_n$. Either ϕ is not a subsentence of ψ , it's a subsentence of ψ but not the same as ψ , or is the same as ψ . If it's not a subsentence of ψ or it's the same as ψ , then just as in the base step it follows immediately that $\psi \vdash \psi^*$.

So assume that ϕ is a subsentence of ψ but not the same as it. Then ϕ is a subsentence of one of the disjuncts θ_i of ψ and ψ^* is the disjunction $\theta_1 \vee \dots \vee \theta_i^* \vee \dots \vee \theta_n$.

We want to show that $\psi \vdash \psi^*$, i.e. that $\theta_1 \vee \dots \vee \theta_n \vdash \theta_1 \vee \dots \vee \theta_i^* \vee \dots \vee \theta_n$. Using the proof by contradiction strategy, we write ψ on the first line and write $\sim\psi^*$ on line 2 as an assumption with the goal of deriving a contradiction. Now we apply *DeMorgans* to line 2, getting $\sim\theta_1 \wedge \dots \wedge \sim\theta_i^* \wedge \dots \wedge \sim\theta_n$. Now using \wedge -Elim we break apart this conjunction, getting each conjunct $\sim\theta_1, \dots, \sim\theta_i^*, \dots, \sim\theta_n$ on a separate line. Then using these conjuncts and *Disjunctive Syllogism* on line 1 to get θ_i on its own line. By the recursive hypothesis, $\theta_i \vdash \theta_i^*$, so from the line with θ_i we can derive θ_i^* . Since we already have $\sim\theta_i^*$ on its own line, we can use \wedge -Intro to get $\theta_i^* \wedge \sim\theta_i^*$. We then close the assumption on line 2 by using \rightarrow -Intro to get $\sim\psi^* \rightarrow (\theta_i^* \wedge \sim\theta_i^*)$. We finally use \sim -Elim to get ψ^* .

Negation: Assume that ψ is the negation $\sim\theta$. Either ϕ is not a subsentence of ψ , it's a subsentence of ψ but not the same as ψ , or is the same as ψ . If it's not a subsentence of ψ or it's the same as ψ , then just as in the base step it follows immediately that $\psi \vdash \psi^*$.

So assume that ϕ is a subsentence of ψ but not the same as it. Then ϕ is a subsentence of θ and ψ^* is the negation $\sim\theta^*$.

We want to show that $\psi \vdash \psi^*$, i.e. that $\sim\theta \vdash \sim\theta^*$. Using our usual basic bottom-up strategy for negation, we set up this proof by putting $\sim\theta$ on the first line and assuming θ^* with the goal of deriving a contradiction. But, by the recursive hypothesis, $\theta^* \vdash \theta$. So we know that from assumption θ^* we can derive θ , and at that point we'll have $\sim\theta$ on one line and θ on another. Using \wedge -Intro we can get $\theta \wedge \sim\theta$. Then with \rightarrow -Intro we'll get $\theta^* \rightarrow (\theta \wedge \sim\theta)$, and applying \sim -Intro to this sentence will get us $\sim\theta^*$.

Conditional: Left to the reader as an exercise.

Biconditional: Also left to the reader as an exercise.

Closure Step: Since the inheritance step covers all the ways to generate SL sentences, we've shown that the theorem holds for all SL sentences ψ . ■

Thus we have shown that anything we can derive in SD^+ can be derived in SD .

6.3.3 Shortcut Rule Strategies

As we discussed in section 6.2.6, for each logical connective there are two types of strategies: those for what to do if you already have sentences with that as their main connective (top-down strategies), and those for what to do if you want to get a sentence with that as its main connective (bottom-up strategies). In section 6.2.6 we covered basic top-down and bottom-up strategies for each connective. We now add new strategies based on shortcut rules to this basic stock. (Since shortcut rules don't divide nicely by connective, and often involve schemas with multiple connectives, some of the groupings here a bit arbitrary; but this isn't a substantial issue.)

Conjunction

DeM Top-down: If you have a sentence of the form $\sim\phi_1 \wedge \dots \wedge \sim\phi_n$, then convert it using DeM to get $\sim(\phi_1 \vee \dots \vee \phi_n)$ on a new line.

\sim/\rightarrow -**Exchange Top-down:** If you have a sentence of the form $\phi \wedge \sim\psi$, then convert it using \sim/\rightarrow -Exchange to get $\sim\phi \rightarrow \psi$ on a new line.

Disjunction

D.S. Top-down: If you have a sentence of the form $\phi_1 \vee \dots \vee \phi_i \vee \dots \vee \phi_n$, and another sentence of the form $\sim\phi_i$, then eliminate one of the disjuncts using D.S. to get $\phi_1 \vee \dots \vee \phi_{i-1} \vee \phi_{i+1} \vee \dots \vee \phi_n$ on a new line.

DeM Top-down: If you have a sentence of the form $\sim\phi_1 \vee \dots \vee \sim\phi_n$, then convert it using DeM to get $\sim(\phi_1 \wedge \dots \wedge \phi_n)$ on a new line.

\rightarrow/\sim -**Exchange Top-down:** If you have a sentence of the form $\sim\phi \vee \psi$, then convert it using \rightarrow/\sim -Exchange to get $\phi \rightarrow \psi$ on a new line.

Negation

DeM Top-down: If you have a sentence of the form $\sim(\phi_1 \wedge \dots \wedge \phi_n)$, then convert it using *DeM* to get $\sim\phi_1 \vee \dots \vee \sim\phi_n$ on a new line.

DeM Top-down: If you have a sentence of the form $\sim(\phi_1 \vee \dots \vee \phi_n)$, then convert it using *DeM* to get $\sim\phi_1 \wedge \dots \wedge \sim\phi_n$ on a new line.

$\sim\sim$ -Elim Top-down: If you have a sentence of the form $\sim\sim\phi$, then convert it using $\sim\sim$ -Elim to get ϕ on a new line.

\sim/\rightarrow -Exchange Top-down: If you have a sentence of the form $\sim(\phi \rightarrow \psi)$, then convert it using \sim/\rightarrow -Exchange to get $\phi \wedge \sim\psi$ on a new line.

$\sim\sim$ -Intro Bottom-up: If you want to get a sentence of the form $\sim\sim\phi$, then first derive ϕ and then use $\sim\sim$ -Intro to derive it.

Conditionals

M.T. Top-down: If you have a sentence of the form $\phi \rightarrow \psi$, and another $\sim\psi$, then break the conditional apart using *M.T.* to get $\sim\phi$ on a new line.

\rightarrow/\sim -Exchange Top-down: If you have a sentence of the form $\phi \rightarrow \psi$, then convert it using \rightarrow/\sim -Exchange to get $\sim\phi \vee \psi$ on a new line.

Contraposition Top-down: If you have a sentence of the form $\phi \rightarrow \psi$, then convert it using *Contraposition* to get $\sim\psi \rightarrow \sim\phi$ on a new line.

Biconditionals

\sim/\leftrightarrow -Intro Top-down: If you have a sentence of the form $\phi \leftrightarrow \psi$, then convert it using \sim/\leftrightarrow -Intro to get $\sim\phi \leftrightarrow \sim\psi$ on a new line.

Misc

A.C. Bottom-up: If you want to get a sentence of the form ψ , then first derive both ϕ and $\sim\phi$ and then use *A.C.* to derive it from them.

Note that we've left strategies derived from *Distribution*. This isn't because they're not useful (quite the contrary). Instead, we leave it to the reader to place the appropriate Top-down strategies from *Distribution* under conjunction and disjunction.

Also note that we've presented most of the strategies as top-down. For any of the strategies based on an exchange shortcut rule, it should be clear that you can "reverse" the strategy and read it as bottom-up.

6.4 QD

6.4.1 Introduction and Elimination Rules

Now our goal is to extend SD to natural deduction system that will allow us to write derivations for QL. This system, which we call *Quantificational Derivation System*, or QD, consists of all the rules of SD, plus an introduction and elimination rule for each quantifier (given in table 6.44). Like SD, we will extend QD by adding shortcut rules. First, we can extend QD by adding all the shortcut rules we gave for SD. Second, we can extend QD by adding shortcut rules specifically for the quantifiers (see table 6.62). We'll call the system we get by adding all the shortcut rules from SD (tables 6.36 and 6.36) and the new shortcut rules for the quantifiers (table 6.62) QD^+ . Note that some of the rules for QD have special restrictions. We explain these in the examples below.

Name	Given	May Add
\forall -Elim	$\forall\beta\phi$	$\phi a/\beta$, for 'a' any individual constant
\forall -Intro	$\phi a/\beta$	$\forall\beta\phi$, iff 'a' does not occur in ϕ nor in any unboxed assumption
\exists -Intro	$\phi a/\beta$	$\exists\beta\phi$
\exists -Elim	$\exists\beta\phi, \phi a/\beta \rightarrow \theta$	θ , iff 'a' does not occur in ϕ or θ , nor in any unboxed assumption

Table 6.44: (New) Basic Rules for QD

The mechanics of writing proofs in QD are no different than the mechanics of writing proofs in SD, but we do have to slightly adapt the definition for sanctioning (Def. 7.3, on page 181).

Definition 6.12. A rule R , applied to unboxed lines m_1, \dots, m_j with, respectively, sentences ψ_1, \dots, ψ_j , *sanctions* writing the sentence ϕ iff there's some substitution of QL *formulas* that, for the given schemas of R , results in ψ_1, \dots, ψ_j and, for the may-add schema, results in ϕ .

Another difference is that there are four new rules available. We now look at four examples which highlight each rule.

Our first example demonstrates \forall -Elim. Say we want to show that $\forall x(Ax \rightarrow Bx)$, $Ad \vdash Bd$. We start by setting the two sentences on the LHS of the turnstile as assumptions:

- 36.
- | | | |
|----|--------------------------------|---------------|
| 1. | $\forall x(Ax \rightarrow Bx)$ | <i>Assume</i> |
| 2. | Ad | <i>Assume</i> |

Next we use \forall -*Elim* on line 1. Note that there are no restrictions on the use of \forall -*Elim*.

- 37.
- | | | |
|----|--------------------------------|-----------------------------|
| 1. | $\forall x(Ax \rightarrow Bx)$ | <i>Assume</i> |
| 2. | Ad | <i>Assume</i> |
| 3. | $Ad \rightarrow Bd$ | \forall - <i>Elim</i> , 1 |

Although we could have substituted any constant for x in line 3, we chose d so that we can next apply \rightarrow -*Elim*.

- 38.
- | | | |
|----|--------------------------------|-----------------------------------|
| 1. | $\forall x(Ax \rightarrow Bx)$ | <i>Assume</i> |
| 2. | Ad | <i>Assume</i> |
| 3. | $Ad \rightarrow Bd$ | \forall - <i>Elim</i> , 1 |
| 4. | Bd | \rightarrow - <i>Elim</i> , 2,3 |

And this completes the derivation.

The next example demonstrates \exists -*Intro*. Here we will show that $\forall y(\exists xDxy \rightarrow By), Dab \vdash Bb$. As before, we start with the assumptions.

- 39.
- | | | |
|----|--|---------------|
| 1. | $\forall y(\exists xDxy \rightarrow By)$ | <i>Assume</i> |
| 2. | Dab | <i>Assume</i> |

Again as before, we need to use \forall -*Elim* so we can eventually use \rightarrow -*Elim* to finish.

- 40.
- | | | |
|----|--|-----------------------------|
| 1. | $\forall y(\exists xDxy \rightarrow By)$ | <i>Assume</i> |
| 2. | Dab | <i>Assume</i> |
| 3. | $\exists xDxb \rightarrow Bb$ | \forall - <i>Elim</i> , 1 |

Again we strategically chose the constant we did, b , so that using \rightarrow -*Elim* will get us the right result. But we can't apply \rightarrow -*Elim* yet, since the LHS of the horseshoe in line 3, $\exists xDxb$, is an existential sentence, while what we have on line 2, Dab , is not. But, by using \exists -*Intro* we can easily get what we need:

41.	1.	$\forall y(\exists x Dxy \rightarrow By)$	<i>Assume</i>
	2.	Dab	<i>Assume</i>
	3.	$\exists x Dxb \rightarrow Bb$	\forall - <i>Elim</i> , 1
	4.	$\exists x Dxb$	\exists - <i>Intro</i> , 2

Note that just as with \forall -*Elim* there are no restrictions on \rightarrow -*Elim*. Also note that we could have used \rightarrow -*Elim* to generalize the constant b , getting $\exists x Dax$, but that wouldn't have helped us. Also, we could have generalized using a different variable, getting, say $\exists z Dzb$. But again that wouldn't have helped us. Any constant is legal to instantiate with \forall -*Elim*, but often only one is a wise choice.

We now have the right setup for \rightarrow -*Elim*:

42.	1.	$\forall y(\exists x Dxy \rightarrow By)$	<i>Assume</i>
	2.	Dab	<i>Assume</i>
	3.	$\exists x Dxb \rightarrow Bb$	\forall - <i>Elim</i> , 1
	4.	$\exists x Dxb$	\exists - <i>Intro</i> , 2
	5.	Bb	\rightarrow - <i>Elim</i> , 3,4

And this completes the derivation.

The next example demonstrates \exists -*Elim*. This is our first rule with restrictions. We will show that $\forall x \sim Qxb, \exists z(Qzb \vee Gz) \vdash \exists x Gx$. As before, we start by setting out the assumptions.

43.	1.	$\forall x \sim Qxb$	<i>Assume</i>
	2.	$\exists z(Qzb \vee Gz)$	<i>Assume</i>

When using both \exists -*Elim* and \forall -*Elim*, it's generally best to do what needs to be done for \exists -*Elim* first and use \forall -*Elim* only when necessary. The reason for this has to do with the restriction on \exists -*Elim* and will become apparent with some practice. Now, according to \exists -*Elim*, we can get a sentence θ in this case by showing that $(Qtb \vee Gt) \rightarrow \theta$, for some constant t that fits the restriction on \exists -*Elim*. (Just what constants will work and what sentence θ we want will take some thought.) So, we need to use \rightarrow -*Intro*.

44.	1.	$\forall x \sim Qxb$	<i>Assume</i>
	2.	$\exists z(Qzb \vee Gz)$	<i>Assume</i>
	3.	$Qab \vee Ga$	<i>Assume</i>
		\vdots	

$n.$		θ	
$n + 1.$		$(Qab \vee Ga) \rightarrow \theta$	$\rightarrow\text{-Intro}, 3\text{--}n$
$n + 2.$		θ	$\exists\text{-Elim}, 2, n + 1$

Here we have chosen a to substitute for z in line 3 because (so long as we pick θ correctly) it will meet our restriction. According to the restriction on $\exists\text{-Elim}$, the constant we pick can't appear in any open assumptions (the assumption used to derive the needed conditional, in this case line 3, is not open by the time we apply the rule). It also can't appear in the scope of the existential quantifier, which in this case is $(Qzb \vee Gz)$. Since a does not appear in any open assumptions and does not appear in $(Qzb \vee Gz)$, it will meet our restriction. (Clearly other constants would have also met the restriction.) Now we have to decide what sentence θ will enable us to finish the derivation. Note that if we derive Gt for any constant $t \neq a$, then we can use $\exists\text{-Intro}$ to get the needed sentence $\exists xGx$. (We will set this up, but we will see in a moment that this first guess won't work.) We can't pick $t = a$ because then we couldn't apply $\exists\text{-Elim}$ on line $n + 2$.

45.	1.	$\forall x \sim Qxb$	<i>Assume</i>
	2.	$\exists z(Qzb \vee Gz)$	<i>Assume</i>
	3.	$Qab \vee Ga$	<i>Assume</i>
		\vdots	
	$n.$	Gc	
	$n + 1.$	$(Qab \vee Ga) \rightarrow Gc$	$\rightarrow\text{-Intro}, 3\text{--}n$
	$n + 2.$	Gc	$\exists\text{-Elim}, 2, n + 1$
	$n + 3.$	$\exists xGx$	$\exists\text{-Intro}, n + 2$

Now we only need to complete the derivation by deriving Gc from $(Qab \vee Ga)$. We can *try* do this by using $\forall\text{-Elim}$ and *D.S.*, but it becomes clear at once that this won't work.

46.	1.	$\forall x \sim Qxb$	<i>Assume</i>
	2.	$\exists z(Qzb \vee Gz)$	<i>Assume</i>
	3.	$Qab \vee Ga$	<i>Assume</i>
	4.	$\sim Qab$	$\forall\text{-Elim}, 1$
	5	Ga	<i>D.S.</i> , 3,4

			\vdots	
$n.$			Gc	
$n + 1.$		$(Qab \vee Ga) \rightarrow Gc$		$\rightarrow\text{-Intro}, 3\text{--}n$
$n + 2.$		Gc		$\exists\text{-Elim}, 2, n + 1$
$n + 3.$		$\exists xGx$		$\exists\text{-Intro}, n + 2$

It should be clear that this $\forall\text{-Elim}/D.S.$ strategy won't work, since for it to work we'd need the token of G that appears in line 3 to be followed by the *same* constant that follows the token of G that appears in line n . But that constant is the constant we substitute in for $\exists\text{-Elim}$, and we would then violate the restriction for the rule.

So we need to go back to θ and consider another strategy. This time we'll try letting $\theta = \exists xGx$.

47.	1.	$\forall x \sim Qxb$		<i>Assume</i>
	2.	$\exists z(Qzb \vee Gz)$		<i>Assume</i>
	3.	$Qab \vee Ga$		<i>Assume</i>
		\vdots		
	$n.$	$\exists xGx$		
	$n + 1.$	$(Qab \vee Ga) \rightarrow \exists xGx$		$\rightarrow\text{-Intro}, 3\text{--}n$
	$n + 2.$	$\exists xGx$		$\exists\text{-Elim}, 2, n + 1$

Note that we still keep our choice of a on line 3 within the restrictions of $\exists\text{-Elim}$. Now we can try again at finishing the proof. This time we can:

48.	1.	$\forall x \sim Qxb$		<i>Assume</i>
	2.	$\exists z(Qzb \vee Gz)$		<i>Assume</i>
	3.	$Qab \vee Ga$		<i>Assume</i>
	4.	$\sim Qab$		$\forall\text{-Elim}, 1$
	5.	Ga		<i>D.S.</i> , 3,4
	6.	$\exists xGx$		$\exists\text{-Intro}, 5$
	7.	$(Qab \vee Ga) \rightarrow \exists xGx$		$\rightarrow\text{-Intro}, 3\text{--}6$
	8.	$\exists xGx$		$\exists\text{-Elim}, 2, 7$

And this completes the derivation.

The final example demonstrates \forall -Intro. This rule also has restrictions. We will show that $\forall x(Ax \rightarrow Bx), \forall x(Bx \rightarrow Hx) \vdash \forall x(Ax \rightarrow Hx)$. It's worth mentioning that the fact that all the examples so far have two sentences on the LHS of the turnstile is just a coincidence. As before, we start by setting out the assumptions.

49.	1.	$\forall x(Ax \rightarrow Bx)$	Assume
	2.	$\forall x(Bx \rightarrow Hx)$	Assume

Before moving forward, we need to think about how \forall -Intro works. According to the rule, if we want to get $\forall x(Ax \rightarrow Hx)$ by using \forall -Intro, we need to first get $At \rightarrow Ht$ on some line, where t is some constant that does not appear in any unboxed assumptions, or in $Ax \rightarrow Hx$. Since no constants appear in $Ax \rightarrow Hx$, it provides no constraints. Further, there are no constants in either unboxed assumption in derivation 49, so we are free to choose any constant we like. Pick a . So we want to derive $Aa \rightarrow Ha$.

50.	1.	$\forall x(Ax \rightarrow Bx)$	Assume
	2.	$\forall x(Bx \rightarrow Hx)$	Assume
	3.	Aa	Assume
		\vdots	
	n .	Ha	
	$n + 1$.	$Aa \rightarrow Ha$	\rightarrow -Intro, 3– n
	$n + 2$.	$\forall x(Ax \rightarrow Hx)$	\forall -Intro, $n + 1$

Now we just have to finish the derivation of Ha from Aa without introducing any new unboxed assumptions with constant a . (Of course, if we did the bottom half of this proof wouldn't work, since we couldn't close the assumption on line 3 with \rightarrow -Intro if unboxed assumptions appeared after line 3.) A natural next step is to use \forall -Elim on lines 1 and 2.

51.	1.	$\forall x(Ax \rightarrow Bx)$	Assume
	2.	$\forall x(Bx \rightarrow Hx)$	Assume
	3.	Aa	Assume
	4.	$Aa \rightarrow Ba$	\forall -Elim, 1
	5.	$Ba \rightarrow Ha$	\forall -Elim, 2

			\vdots	
$n.$			Ha	
$n + 1.$		$Aa \rightarrow Ha$		$\rightarrow\text{-Intro}, 3\text{--}n$
$n + 2.$		$\forall x(Ax \rightarrow Hx)$		$\forall\text{-Intro}, n + 1$

Now getting to Ha is just a matter of using $\rightarrow\text{-Elim}$:

52.	1.	$\forall x(Ax \rightarrow Bx)$	Assume
	2.	$\forall x(Bx \rightarrow Hx)$	Assume
	3.	Aa	Assume
	4.	$Aa \rightarrow Ba$	$\forall\text{-Elim}, 1$
	5.	$Ba \rightarrow Ha$	$\forall\text{-Elim}, 2$
	6.	Ba	$\rightarrow\text{-Elim}, 3, 4$
	7.	Ha	$\rightarrow\text{-Elim}, 5, 6$
	8.	$Aa \rightarrow Ha$	$\rightarrow\text{-Intro}, 3\text{--}7$
	9.	$\forall x(Ax \rightarrow Hx)$	$\forall\text{-Intro}, 8$

And this completes the derivation. Notice that since there are infinitely many constants in QL, and any of them would have worked in the derivation, there are infinitely many derivations of this sentence.

6.4.2 Shortcut Rules for QD

All of the shortcut rules for SD, both the standard and exchange rules, can be carried over as shortcut rules for QD. In addition, there are four new shortcut rules for QD, the quantifier negation rules. These are found in table 6.62.

Name	Given	May Add
QN	$\sim\forall\beta\phi$	$\exists\beta\sim\phi$
	$\exists\beta\sim\phi$	$\sim\forall\beta\phi$
	$\sim\exists\beta\phi$	$\forall\beta\sim\phi$
	$\forall\beta\sim\phi$	$\sim\exists\beta\phi$

Table 6.62: Exchange Short-Cut Rules for QD

Just as we had to slightly modify the definition of sanctioning used in SD for the basic (intro and elimination) rules and standard shortcut rules for QD, we also have to slightly modify the definition of sanctioning used in SD^+ (Def. 6.7, on page 157) for the exchange shortcut rules for QD that make up QD^+ .

Definition 6.13. An exchange shortcut rule R of QD^+ (a rule from table 6.37, on page 150, or table 6.62, on the previous page), applied to a line with a QL sentence ψ , *sanctions* writing down sentence ψ^* iff

- (1) there is some substitution of QL formulas that, for the given schema of R , results in a formula ϕ and, for the may-add schema, results in a formula ϕ^* ,
- (2) ϕ is a subformula of ψ , and
- (3) ψ^* is the SL sentence you get when you replace one instance (token) of ϕ with an instance (token) of ϕ^* in ψ .

6.4.3 Shortcut Rule Elimination Theorem for QD

In this section we want to extend the Shortcut Rule Elimination Theorem (Thm. 6.5, on page 152) to QD.

Theorem 6.14. Shortcut Rule Elimination Theorem for QD^+ : *For all QL sentences ϕ_1, \dots, ϕ_m and ψ , if ψ can be derived from ϕ_1, \dots, ϕ_m in QD^+ , then ψ can be derived from ϕ_1, \dots, ϕ_m in QD.*

The same proof we used for Shortcut Rule Elimination Theorem for SD (Thm. 6.5) will work for this version, so long as appropriate versions of theorems 6.3 (on page 151) and 6.4 (on page 152) hold for the shortcut rules of QD. Specifically:

Theorem 6.15. *For all QL sentences $\theta_1, \dots, \theta_n, \delta$ and rules R_1, \dots, R_p , if*

- (1) δ can be derived from $\theta_1, \dots, \theta_n$ using rules R_1, \dots, R_p and the basic rules of SD, and
- (2) every application of a rule R_1 is derivable using the rules R_2, \dots, R_p and the basic rules of QD (recall Def. 6.2, on page 151),

then δ can be derived from $\theta_1, \dots, \theta_n$ using only rules R_2, \dots, R_p and the basic rules of QD.

Theorem 6.16. *For all standard and exchange shortcut rules R (see tables 6.36, 6.37, and 6.62), every application of R is derivable using the basic rules of QD (see tables 6.1 and 6.44).*

We leave it to the reader to prove the Shortcut Rule Elimination Theorem (Thm. 6.14) using theorems 6.15 and 6.16.

Turning to the proofs for theorems 6.15 and 6.16, note that nothing in the proof of 6.3 depended on any special features of SD. Thus, the proof of theorem 6.3 can be

adapted to 6.15 just by changing all the references to SD to references to QD. But unfortunately theorem 6.16 is not nearly as straightforward.

It is helpful to break theorem 6.16 into two parts: (i) the claim that, for all standard shortcut rules (table 6.36), every application is derivable using the basic rules of QD, and (ii) the claim that, for all the exchange shortcut rules (tables 6.37 and 6.62), every application is derivable using the basic rules of QD. Proving part (i) of theorem 6.16 is no different from proving it for theorem 6.4; the same arguments using the derivation schemas written for theorem 6.4 will work. But nothing done so far will help with part (ii). This is because applications of exchange shortcut rules in QD^+ , even those shared with SD^+ (see table 6.37), use a different definition of sanctioning than was used in SD^+ (compare Def. 6.7 and 6.13). According to definition 6.13 (on the previous page), in QD^+ exchange rules can be applied not only to subsentences, but also to subformulas. We have to show that allowing exchange rules to be applied not only to subsentences, but also to subformulas, doesn't prevent us from deriving their applications using only the basic rules.

To do this we will use (1) an extended (and generalized) version of the Restricted Replacement Theorem for SD (Thm. 6.11, on page 160) and (2) the fact that any two formulas got by substituting QL formulas into the may-add and given schemas of the exchange shortcut rules for QD are provably equivalent. (We ask the reader to prove this second fact in exercise 6.5.3, on page 177.)

Theorem 6.17. *The Replacement Theorem for QD: If ϕ and ϕ^* are provably equivalent formulas of QL, and θ and θ^* differ only in that θ contains the subformula ϕ in one place where θ^* contains the subformula ϕ^* , then θ and θ^* are provably equivalent.*

Before proving this theorem, we need to define when two *formulas* of QL are provably equivalent. The definition given for SL sentences (def. 6.10, on page 160) will not carry over to QL formulas, since formulas just aren't the sort of thing which can be derived. For example, we would like to be able to say that $(Qx \rightarrow Gy)$ and $(\sim Qx \vee Gy)$ are provably equivalent even though we cannot derive the formula $(Qx \rightarrow Gy) \leftrightarrow (\sim Qx \vee Gy)$ (because it's not a sentence).

The way we extend the notion of provable equivalence is through the universal closure of a formula.

Definition 6.18. The *universal closure* of a formula θ , written $\forall\theta$, is the sentence that results by prefixing universal quantifiers in alphabetical order for all free variables of θ .

E.g., $\forall((Qx \rightarrow Gy) \leftrightarrow (\sim Qx \vee Gy))$, the universal closure of $((Qx \rightarrow Gy) \leftrightarrow (\sim Qx \vee Gy))$, is $\forall x \forall y((Qx \rightarrow Gy) \leftrightarrow (\sim Qx \vee Gy))$.

We can now define provable equivalence for QL formulas using the universal closure:

Definition 6.19. Two QL formulas θ and ϕ are *provably equivalent* iff the universal closure of the formula that has a biconditional as main connective and θ and ϕ as immediate constituents is derivable in QD; in other words, iff $\vdash \forall(\theta \leftrightarrow \phi)$ in QD.

It's important to note that this definition really is a generalization of definition 6.10 (on page 160). If θ and ϕ are sentences of SL (recall that every sentence of SL is also a formula of QL), then they are provably equivalent on definition 6.10 iff they are provably equivalent on definition 6.19.

Before moving to the proof of the Replacement Theorem for QD (Thm. 6.17 on the previous page), it will be convenient to prove the following one-step Replacement Lemmas:

Theorem 6.20. One-step Replacement Lemmas: *If $\vdash \forall(\phi \leftrightarrow \phi^*)$, then:*

- (1) $\vdash \forall(\sim\phi \leftrightarrow \sim\phi^*)$
- (2) $\vdash \forall((\phi \wedge \phi_1 \wedge \dots \wedge \phi_p) \leftrightarrow (\phi^* \wedge \phi_1 \wedge \dots \wedge \phi_p))$
- \vdots
- (3) $\vdash \forall((\phi_1 \wedge \dots \wedge \phi_p \wedge \phi) \leftrightarrow (\phi_1 \wedge \dots \wedge \phi_p \wedge \phi^*))$
- (4) $\vdash \forall((\phi \vee \phi_1 \vee \dots \vee \phi_p) \leftrightarrow (\phi^* \vee \phi_1 \vee \dots \vee \phi_p))$
- \vdots
- (5) $\vdash \forall((\phi_1 \vee \dots \vee \phi_p \vee \phi) \leftrightarrow (\phi_1 \vee \dots \vee \phi_p \vee \phi^*))$
- (6) $\vdash \forall((\phi \rightarrow \psi) \leftrightarrow (\phi^* \rightarrow \psi))$
- (7) $\vdash \forall((\psi \rightarrow \phi) \leftrightarrow (\psi \rightarrow \phi^*))$
- (8) $\vdash \forall((\phi \leftrightarrow \psi) \leftrightarrow (\phi^* \leftrightarrow \psi))$
- (9) $\vdash \forall((\psi \leftrightarrow \phi) \leftrightarrow (\psi \leftrightarrow \phi^*))$

And, if $\vdash \forall\forall\beta(\phi \leftrightarrow \phi^*)$, then:

- (10) $\vdash \forall(\forall\beta\phi \leftrightarrow \forall\beta\phi^*)$
- (11) $\vdash \forall(\exists\beta\phi \leftrightarrow \exists\beta\phi^*)$

We will prove (2) for the case of a 2-place conjunction and leave the rest to the reader to prove in a similar way. We will use the following notation. If ϕ is a QL formula, then let x_1, \dots, x_m be the complete list of free variables in ϕ . Further, let $\phi c_1 \dots c_m / x_1 \dots x_m$ be the formula you get by substituting c_1 for x_1 , ..., and c_m for x_m .

Proof of Thm. 6.20, (2), for 2-place Conjunctions: Assume that $\vdash \forall(\phi \leftrightarrow \phi^*)$. Then consider some derivation D in QD of $\forall(\phi \leftrightarrow \phi^*)$. The basic idea will be to extend this derivation to a derivation of $\forall((\phi \wedge \psi) \leftrightarrow (\phi^* \wedge \psi))$ by first stripping away the initial quantifiers, then manipulating the truth functional connectives, and, finally, restoring the quantifiers. In detail, the new extended derivation should go (to save space when numbering lines, let $q = n + m$):

	\vdots	
$n.$	$\forall(\phi \leftrightarrow \phi^*)$	last line of D
$n + 1.$	$\forall[(\phi \leftrightarrow \phi^*)c_1/x_1]$	\forall -Elim, n
	\vdots	
$n + m.$	$(\phi \leftrightarrow \phi^*)c_1 \dots c_m/x_1 \dots x_m$	\forall -Elim, $n + m - 1$
$q + 1.$	$(\phi \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	Assume
$q + 2.$	$\phi c_1 \dots c_m/x_1 \dots x_m$	\wedge -Elim, $q + 1$
$q + 3.$	$\phi^* c_1 \dots c_m/x_1 \dots x_m$	\leftrightarrow -Elim, $q, q + 2$
$q + 4.$	$\psi c_1 \dots c_m/x_1 \dots x_m$	\wedge -Elim, $q + 1$
$q + 5.$	$(\phi^* \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	\wedge -Intro, $q + 3, q + 4$
$q + 6.$	$(\phi \wedge \psi)c_1 \dots c_m/x_1 \dots x_m \rightarrow$ $(\phi^* \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	\rightarrow -Intro, $q + 1 - q + 5$
$q + 7.$	$(\phi^* \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	Assume
$q + 8.$	$\phi^* c_1 \dots c_m/x_1 \dots x_m$	\wedge -Elim, $q + 7$
$q + 9.$	$\psi c_1 \dots c_m/x_1 \dots x_m$	\wedge -Elim, $q + 7$
$q + 10.$	$\phi c_1 \dots c_m/x_1 \dots x_m$	\leftrightarrow -Elim, $q, q + 8$
$q + 11.$	$(\phi \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	\wedge -Intro, $q + 9, q + 10$
$q + 12.$	$(\phi^* \wedge \psi)c_1 \dots c_m/x_1 \dots x_m \rightarrow$ $(\phi \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	\rightarrow -Intro, $q + 7 - q + 11$
$q + 13.$	$[(\phi \wedge \psi) \leftrightarrow$ $(\phi^* \wedge \psi)]c_1 \dots c_m/x_1 \dots x_m$	\leftrightarrow -Intro, $q + 6, q + 12$
$q + 14.$	$\forall[(\phi \wedge \psi) \leftrightarrow$ $(\phi^* \wedge \psi)]c_1 \dots c_{m-1}/x_1 \dots x_{m-1}$	\forall -Intro, $q + 13$
	\vdots	
$n + 2m.$	$\forall((\phi \wedge \psi) \leftrightarrow (\phi^* \wedge \psi))$	\forall -Intro, $n + 2m - 13$

It is important to note that all the constants introduced on lines $n + 1$ through $n + m$ need to be new constants that do not appear in any previous lines. If not, then there's no guarantee that we'll be able to do \forall -Intro on the end lines. ■

Proof of Thm. 6.17: Just as with the proof of the Restricted Replacement Theorem for SD (Thm. 6.11, on page 160), the proof for the Replacement Theorem for QD is a recursive proof. But, since the definition of provably equivalent is different (we've extended it to formulas of QL) we can't simply extend the proof of theorem 6.11 by adding new cases to the inheritance step for the quantifiers.

Assume that ϕ and ϕ^* are provably equivalent formulas of QL (assume that $\vdash \forall(\phi \leftrightarrow \phi^*)$), that ϕ is a subformula of θ , and that θ^* is the result of replacing ϕ with

ϕ^* in θ .

Base Step: Similar to the base step in the proof of theorem 6.11, in the base case θ is atomic and so has no subformula other than itself. So, if ϕ is a subformula of θ , then $\phi = \theta$. Hence $\theta^* = \phi^*$. Since ϕ and ϕ^* are provably equivalent, it follows immediately that θ and θ^* are provably equivalent.

Inheritance Step:

Recursive Assumption: Assume that the theorem holds for formulas $\psi, \psi_1, \dots, \psi_k$; that is, assume that if ψ^* is the result of replacing ϕ with ϕ^* , then $\vdash \forall(\psi \leftrightarrow \psi^*)$, and similarly for the others.

Negation: Assume that $\theta = \sim\psi$. Either $\phi = \theta$, in which case it trivially follows that $\vdash \forall(\theta \leftrightarrow \theta^*)$, or ϕ is a subformula of ψ (and hence $\theta^* = \sim\psi^*$). By the recursive assumption, $\vdash \forall(\psi \leftrightarrow \psi^*)$. It follows by the One-step Replacement Lemma (Thm. 6.20) that $\vdash \forall(\sim\psi \leftrightarrow \sim\psi^*)$.

Conjunction: Assume that $\theta = \psi_1 \wedge \dots \wedge \psi_k$. Either $\phi = \theta$, in which case it trivially follows that $\vdash \forall(\theta \leftrightarrow \theta^*)$, or ϕ is a subformula of one of the conjuncts ψ_i (and hence $\theta^* = \psi_1 \wedge \dots \wedge \psi_i^* \wedge \dots \wedge \psi_k$). By the recursive assumption, $\vdash \forall(\psi_i \leftrightarrow \psi_i^*)$. It follows by the One-step Replacement Lemma (Thm. 6.20) that $\vdash ((\psi_1 \wedge \dots \wedge \psi_i \wedge \dots \wedge \psi_k) \leftrightarrow (\psi_1 \wedge \dots \wedge \psi_i^* \wedge \dots \wedge \psi_k))$.

Disjunction: This case is left to the reader.

Conditional: This case is also left to the reader.

Biconditional: This case is also left to the reader.

Universal: Assume that $\theta = \forall\beta\psi$. Either $\phi = \theta$, in which case it trivially follows that $\vdash \forall(\theta \leftrightarrow \theta^*)$, or ϕ is a subformula of ψ (and hence $\theta^* = \forall\beta\psi^*$). By the recursive assumption, $\vdash \forall(\psi \leftrightarrow \psi^*)$. To derive $\forall(\forall\beta\psi \leftrightarrow \forall\beta\psi^*)$, start with the derivation of $\forall(\psi \leftrightarrow \psi^*)$. Extend it by adding as many steps of \forall -Elim as needed to get to the sentence $(\psi \leftrightarrow \psi^*)_{c_1 \dots c_m / x_1 \dots x_m}$. Then using \forall -Intro first on β , then on the others we can get $\forall\beta\psi(\psi \leftrightarrow \psi^*)$ on the line. Hence $\vdash \forall\beta\psi(\psi \leftrightarrow \psi^*)$, so by the One-step Replacement Lemma (Thm. 6.20) we get that $\vdash \forall(\forall\beta\psi \leftrightarrow \forall\beta\psi^*)$.

Existential: This case is exactly the same, except that a different result from the One-step Replacement Lemma is used.

Closure Step: Since the inheritance step covers all the ways to generate QL formulas, we've shown that the theorem holds for all QL formulas θ . ■

Proof of Thm. 6.16, Part (ii): Since any two formulas ϕ and ϕ^* got by substituting QL formulas into the may-add and given schemas of the exchange shortcut rules from

QD^+ (tables 6.37 and 6.62) are provably equivalent, it follows from the Replacement Theorem for QD (Thm. 6.17 on page 172) that if θ^* is a sentence sanctioned by an exchange rule applied to some sentence θ , then θ and θ^* are provably equivalent. That is, $\vdash \forall(\theta \leftrightarrow \theta^*)$. Since θ and θ^* are sentences, the universal closure of their biconditional $\theta \leftrightarrow \theta^*$ is just the biconditional itself, so $\vdash (\theta \leftrightarrow \theta^*)$. But since $\vdash (\theta \leftrightarrow \theta^*)$, it should be clear that $\theta \vdash \theta^*$. Thus, any application of an exchange rule from QD^+ is derivable using the basic rules of QD alone. ■

6.5 Exercises

6.5.1 SD Practice Problems

Write derivations for each of the following using only the rules specified by the instructor. It is probably a good idea to do the problems in order, as the earlier ones tend to be easier than the later ones.

1. $\vdash A \rightarrow (B \rightarrow A)$
2. $\vdash (A \rightarrow B) \rightarrow ((B \rightarrow C) \rightarrow (A \rightarrow C))$
3. $\vdash (A \rightarrow B) \rightarrow ((A \vee C) \rightarrow (B \vee C \vee D))$
4. $\vdash ((A \rightarrow B) \wedge C \wedge (C \leftrightarrow A)) \rightarrow B$
5. $\vdash \sim A \vee (B \rightarrow A)$
6. $\vdash (A \rightarrow B) \vee (B \rightarrow C)$
7. $\vdash (\sim C \wedge ((A \rightarrow C) \vee (B \rightarrow C))) \rightarrow \sim(A \wedge B)$

6.5.2 SD Shortcut Rules

Finish the proof of theorem 6.4 (on page 152). To do this, write derivation schemas for each of the following using only the basic rules of SD or shortcut rules for which you've already done a derivation schema. Note that this is also sufficient to show that sentences got by substituting into the given and may-add schemas of the SD exchange shortcut rules are provably equivalent (Def. 6.10, on page 160). Note that a star * has been placed next to the ones that have already been done in the text above. (They are left in the list for completeness.) Hint: derivations 17 and 26 should be helpful in doing two of the problems below.

M.T.

1. $\phi \rightarrow \theta, \sim \theta \vdash \sim \phi$

D.S.

2. $\phi \vee \theta, \sim \phi \vdash \theta$
3. $\sim \phi \vee \theta, \phi \vdash \theta$

A.C.

4. $\phi, \sim \phi \vdash \psi^*$ (derivation 29)

\sim / \leftrightarrow -Intro

5. $\phi \leftrightarrow \psi \vdash \sim \phi \leftrightarrow \sim \psi$

DeM

6. $\sim(\phi \wedge \theta) \vdash (\sim \phi \vee \sim \theta)$
7. $\sim \phi \vee \sim \theta \vdash \sim(\phi \wedge \theta)$
8. $\sim(\phi \vee \theta) \vdash (\sim \phi \wedge \sim \theta)$

9. $\sim\phi \wedge \sim\theta \vdash \sim(\phi \vee \theta)$ * (derivation 33)

13. $\sim\phi \vee \theta \vdash \phi \rightarrow \theta$

$\sim\sim$ -Elim

10. $\sim\sim\phi \vdash \phi$

Contraposition

14. $\phi \rightarrow \theta \vdash \sim\theta \rightarrow \sim\phi$

$\sim\sim$ -Intro

11. $\phi \vdash \sim\sim\phi$

15. $\sim\theta \rightarrow \sim\phi \vdash \phi \rightarrow \theta$

\rightarrow / \vee -Exchange

12. $\phi \rightarrow \theta \vdash \sim\phi \vee \theta$

\sim / \rightarrow -Exchange

16. $\sim(\phi \rightarrow \theta) \vdash \phi \wedge \sim\theta$

17. $\phi \wedge \sim\theta \vdash \sim(\phi \rightarrow \theta)$

Distribution

18. $\theta \wedge (\phi_1 \vee \phi_2) \vdash (\theta \wedge \phi_1) \vee (\theta \wedge \phi_2)$

19. $(\theta \wedge \phi_1) \vee (\theta \wedge \phi_2) \vdash \theta \wedge (\phi_1 \vee \phi_2)$

20. $(\phi_1 \vee \phi_2) \wedge \theta \vdash (\phi_1 \wedge \theta) \vee (\phi_2 \wedge \theta)$

21. $(\phi_1 \wedge \theta) \vee (\phi_2 \wedge \theta) \vdash (\phi_1 \vee \phi_2) \wedge \theta$

22. $\theta \vee (\phi_1 \wedge \phi_2) \vdash (\theta \vee \phi_1) \wedge (\theta \vee \phi_2)$

23. $(\theta \vee \phi_1) \wedge (\theta \vee \phi_2) \vdash \theta \vee (\phi_1 \wedge \phi_2)$

24. $(\phi_1 \wedge \phi_2) \vee \theta \vdash (\phi_1 \vee \theta) \wedge (\phi_2 \vee \theta)$

25. $(\phi_1 \vee \theta) \wedge (\phi_2 \vee \theta) \vdash (\phi_1 \wedge \phi_2) \vee \theta$

26. $\theta \leftrightarrow \psi \vdash (\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)$

27. $(\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi) \vdash \theta \leftrightarrow \psi$

6.5.3 QD Shortcut Rules

Prove that any two formulas got by substituting QL formulas into the may-add and given schemas of the exchange shortcut rules for QD are provably equivalent (Def. 6.19, on page 172). That is, show that the following hold for all QL formulas $\phi, \phi_1, \phi_2, \theta, \psi$ by writing the appropriate derivation schemas. Note that all but (7) and (8) deal with exchange shortcut rules from SD^+ . For these virtually all the work has been done in exercise 6.5.2; all you need to do is show how to put the derivation schemas done there together and how to remove and put back on the quantifiers needed to make the universal closure.

1. $\vdash \forall[\sim(\phi_1 \wedge \phi_2) \leftrightarrow (\sim\phi_1 \vee \sim\phi_2)]$

5. $\vdash \forall[(\phi \rightarrow \theta) \leftrightarrow (\sim\theta \rightarrow \sim\phi)]$

2. $\vdash \forall[\sim(\phi_1 \vee \phi_2) \leftrightarrow (\sim\phi_1 \wedge \sim\phi_2)]$

6. $\vdash \forall[\sim(\phi \rightarrow \theta) \leftrightarrow (\phi \wedge \sim\theta)]$

3. $\vdash \forall[\sim\sim\phi \leftrightarrow \phi]$

7. $\vdash \forall[\sim\forall\beta\phi \leftrightarrow \exists\beta\sim\phi]$

4. $\vdash \forall[(\phi \rightarrow \theta) \leftrightarrow (\sim\phi \vee \theta)]$

8. $\vdash \forall[\sim\exists\beta\phi \leftrightarrow \forall\beta\sim\phi]$

9. $\vdash \forall[(\theta \wedge (\phi_1 \vee \phi_2)) \leftrightarrow ((\theta \wedge \phi_1) \vee (\theta \wedge \phi_2))]$
10. $\vdash \forall[(\phi_1 \vee \phi_2) \wedge \theta \leftrightarrow ((\phi_1 \wedge \theta) \vee (\phi_2 \wedge \theta))]$
11. $\vdash \forall[(\theta \vee (\phi_1 \wedge \phi_2)) \leftrightarrow ((\theta \vee \phi_1) \wedge (\theta \vee \phi_2))]$
12. $\vdash \forall[(\phi_1 \wedge \phi_2) \vee \theta \leftrightarrow ((\phi_1 \vee \theta) \wedge (\phi_2 \vee \theta))]$
13. $\vdash \forall[(\theta \leftrightarrow \psi) \leftrightarrow ((\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi))]$

6.5.4 QD Practice Problems

Write derivations for each of the following using only the rules specified by the instructor. It is probably a good idea to do the problems in order, as the earlier ones tend to be easier than the later ones.

1. $\vdash \forall x \forall y Qxy \rightarrow \forall z Qzz$
2. $\vdash \forall x \forall y Qxy \rightarrow \forall x \forall y Qyx$
3. $\vdash \forall x(Qx \wedge Gx) \rightarrow (\forall x Qx \wedge \forall x Gx)$
4. $\vdash (\forall x Qx \wedge \forall x Gx) \rightarrow \forall x(Qx \wedge Gx)$
5. $\vdash (\forall x Qx \vee \forall x Gx) \rightarrow \forall x(Qx \vee Gx)$
6. $\vdash \forall x(Qx \rightarrow Gx) \rightarrow (\forall x Qx \rightarrow \forall x Gx)$
7. $\vdash \forall x(P \wedge Qx) \rightarrow (P \wedge \forall x Qx)$
8. $\vdash (P \wedge \forall x Qx) \rightarrow \forall x(P \wedge Qx)$
9. $\vdash \forall x(P \vee Qx) \rightarrow (P \vee \forall x Qx)$
10. $\vdash (P \vee \forall x Qx) \rightarrow \forall x(P \vee Qx)$
11. $\vdash \forall x(P \rightarrow Qx) \rightarrow (P \rightarrow \forall x Qx)$
12. $\vdash (P \rightarrow \forall x Qx) \rightarrow \forall x(P \rightarrow Qx)$
13. $\vdash \exists x \forall y Qxy \rightarrow \forall y \exists x Qxy$
14. $\vdash \forall x(Qx \rightarrow Gx) \rightarrow (\exists x Qx \rightarrow \exists x Gx)$
15. $\vdash \exists x(Qx \wedge Gx) \rightarrow (\exists x Qx \wedge \exists x Gx)$
16. $\vdash (\exists x Qx \vee \exists x Gx) \rightarrow \exists x(Qx \vee Gx)$
17. $\vdash \exists x(P \wedge Qx) \rightarrow (P \wedge \exists x Qx)$
18. $\vdash (P \wedge \exists x Qx) \rightarrow \exists x(P \wedge Qx)$
19. $\vdash \exists x(P \vee Qx) \rightarrow (P \vee \exists x Qx)$
20. $\vdash (P \vee \exists x Qx) \rightarrow \exists x(P \vee Qx)$
21. $\vdash \exists x(P \rightarrow Qx) \rightarrow (P \rightarrow \exists x Qx)$
22. $\vdash (P \rightarrow \exists x Qx) \rightarrow \exists x(P \rightarrow Qx)$
23. $\vdash \forall x(Qx \rightarrow P) \rightarrow (\exists x Qx \rightarrow P)$
24. $\vdash (\exists x Qx \rightarrow P) \rightarrow \forall x(Qx \rightarrow P)$
25. $\vdash \forall x \exists y(Qx \wedge Gy) \rightarrow (\forall x Qx \wedge \exists y Gy)$
26. $\vdash (\forall x Qx \wedge \exists y Gy) \rightarrow \forall x \exists y(Qx \wedge Gy)$
27. $\vdash \forall x \exists y(Qx \wedge Gy) \rightarrow \exists y \forall x(Qx \wedge Gy)$
28. $\vdash \forall x \exists y(Qx \vee Gy) \rightarrow (\forall x Qx \vee \exists y Gy)$
29. $\vdash (\exists y Gy \vee \forall x Qx) \rightarrow \forall x \exists y(Qx \vee Gy)$
30. $\vdash \exists y \forall x(Qx \vee Gy) \rightarrow \forall x \exists y(Qx \vee Gy)$

31. $\vdash \exists y \forall x (Qx \rightarrow Gy) \rightarrow \forall x \exists y (Qx \rightarrow Gy)$
32. $\vdash \forall x \exists y (Qx \vee Gy) \rightarrow \exists y \forall x (Qx \vee Gy)$
33. $\vdash (\exists y Qy \rightarrow \exists x Gx) \rightarrow \exists y \forall x (Qx \rightarrow Gy)$
34. $\vdash \exists y \forall x (Qx \rightarrow Gy) \rightarrow (\exists x Qx \rightarrow \exists x Gx)$
35. $\vdash \forall x \exists y (Qx \rightarrow Gy) \rightarrow \exists y \forall x (Qx \rightarrow Gy)$
36. $\vdash \exists x \exists y (Qx \wedge \sim Qy) \rightarrow (\exists x Qx \wedge \exists x \sim Qx)$
37. $\vdash \exists x \forall y (Qx \rightarrow Gy) \rightarrow (\forall x Qx \rightarrow \forall x Gx)$
38. $\vdash (\exists x Qx \wedge \exists x \sim Qx) \rightarrow \exists x \exists y (Qx \wedge \sim Qy)$
39. $\vdash (\forall x Qx \rightarrow \forall x Gx) \rightarrow \exists x \forall y (Qx \rightarrow Gy)$
40. $\vdash (\sim \exists x Qx \vee \forall x Qx) \rightarrow \forall x \forall y (Qx \rightarrow Qy)$

Chapter 7

Soundness and Completeness

7.1 Introduction

In the previous chapter, we stipulated restrictions for the rule applications of SD (and QD) so that the rules would be *truth-preserving*.

Definition 7.1. A rule is *truth-preserving* iff the sentence or sentences to which the rule is applied entail any sentence which the rule sanctions you to write as the next step.

Definition 7.2. A *formal derivation rule* is a sequence of sentence schemas, the first through the second last of which is called the *given schemas* and the last is called the *may-add schema*.

As the names suggest, table 6.1 (on page 129) lists rules by putting the first through second to last schemas in the left column, labeled “Given”, and the last schema in the right column, labeled “May Add”. We only bring out that we can think of rules as sequences of sentence schemas, and call them the given schemas and the may-add schema so the next definition is easier to state.

Definition 7.3. A rule R , applied to unboxed lines m_1, \dots, m_j with, respectively, sentences ψ_1, \dots, ψ_j , *sanctions* writing the sentence ϕ iff there’s some substitution of SL sentences that, for the given schemas of R , results in ψ_1, \dots, ψ_j and, for the may-add schema, results in ϕ .

As an example, consider again derivation 10, on page 133. The rule \rightarrow -Elim was applied to line 2, which had sentence $B \rightarrow (C \wedge D)$, and line 3, which had sentence B , to get line 4, which had sentence $C \wedge D$. Using definition 7.3 we can show that this move is sanctioned by \rightarrow -Elim by noting, from table 6.1 (on page 129), that \rightarrow -Elim has two given schemas, $\theta \rightarrow \psi$ and θ , and the may-add scheme ψ . Substituting $\theta = B$ and $\psi = C \wedge D$ in the given schemas gets us lines 2 and 3, while making this same substitution in the may-add schema gets us line 4.

Lastly, we end with the following theorem:

Theorem 7.4. *Every application of every basic rule of SD is truth-preserving.*

Proof: It can be shown that: for any basic rule R of SD, if some substitution of SL sentences into the given schema of R results in SL sentences ψ_1, \dots, ψ_n and that

same substitution into the may-add schema of R results in the SL sentence ϕ , then $\psi_1, \dots, \psi_n \models \phi$. Call this the truth-preservation lemma. (We asked the reader to show that the lemma is true in exercises 2.8.8, on page 56.) Now consider some arbitrary application of some basic rule R of SD. Say that in this application R is applied to sentences $\theta_1, \dots, \theta_m$ and permits, or sanctions, you to write down δ . By definition 7.3 (on the previous page), there's some substitution of SL sentences that, for the given schemas of R , results in $\theta_1, \dots, \theta_m$ and, for the may-add schema, results in δ . By the truth preservation lemma, $\theta_1, \dots, \theta_m \models \delta$. By definition 7.1 (on the previous page), this application is truth-preserving.

This proof doesn't cover \rightarrow -Intro or Assume. We will have to give these rules special treatment. The former is the only rule that eliminates an assumption, and the latter is the only rule that adds an assumption, so they each have a special role. ■

Recall from section 6.1 (on page 127) that we want to use derivations as a way of showing that a sentence is a logical truth, or of showing that some set of sentences entails some other sentence. Specifically, we want to use derivations in SD and QD to show that sentences of SL and QL are TFT and QT, or to show entailments between sentences of SL or between sentences of QL. But derivations can only fill this role if our derivation systems are sound. Generally speaking, a derivation system is only totally satisfactory if it is also complete. Let L be some formal language for which we have defined some kind of models.

Definition 7.5. A derivation system D for L is *sound* iff for every set Δ of sentences of L and every sentence ϕ of L, if $\Delta \vdash \phi$, then $\Delta \models \phi$.

7.2 Soundness

7.2.1 Soundness of SD

We begin by proving the soundness of SD.

Theorem 7.6. SD Soundness Theorem: *SD is sound; i.e., for every set Δ of sentences of SL and every sentence ϕ of SL, if $\Delta \vdash \phi$ in SD, then $\Delta \models \phi$.*

We will first prove the following result about derivations:

Theorem 7.7. Soundness Lemma: *For any sequence of derivation lines that is a derivation (see definition 6.1, on page 134), the sentence ϕ on the last line is entailed by the set Δ of sentences that are on unboxed lines and are sanctioned by Assumption.*

Since the definition of a derivation (def. 6.1, on page 134) is a recursive definition, the most natural way to prove theorem 7.7 is through a recursive proof. The recursive proof we give will use three easily proved lemmas (proofs are left to the reader; the first lemma, on monotonicity, is also used to prove thm. 7.6). Two are facts about entailment and one is about derivation.

Theorem 7.8. Monotonicity of Entailment: *For all SL sentences $\phi_1, \dots, \phi_n, \theta, \psi$:*

If $\phi_1, \phi_2, \dots, \phi_n \models \psi$, then $\phi_1, \phi_2, \dots, \phi_n, \theta \models \psi$

Theorem 7.9. Transitivity of Entailment: *For all SL sentences ϕ_1, \dots, ϕ_n , θ , and ψ_1, \dots, ψ_k :*

*If $\phi_1, \phi_2, \dots, \phi_n \models \psi_1$ and
 $\phi_1, \phi_2, \dots, \phi_n \models \psi_2$ and
 \vdots
 $\phi_1, \phi_2, \dots, \phi_n \models \psi_k$ and
 $\psi_1, \psi_2, \dots, \psi_k \models \theta$ then:
 $\phi_1, \phi_2, \dots, \phi_n \models \theta$*

Theorem 7.10. Non-decreasing Assumption Principle (NDAP): *If Δ_1 is the set of assumptions of an unboxed line and Δ_2 is the set of assumptions of a later unboxed line, then Δ_1 is a subset of Δ_2 , i.e., $\Delta_1 \subseteq \Delta_2$.*

Proof of Thm. 7.7, Soundness Lemma:

Base Step: The base case is a single-line derivation D sanctioned by the rule *Assumption*. Say the sentence on that line is ϕ . We have to show that the sentence on the last line is entailed by all the sentences, on unboxed lines, that are sanctioned by *Assumption*. But in this case the sentence on the last line is ϕ , and the set of unboxed sentences sanctioned by *Assumption* only contains ϕ . Obviously $\phi \models \phi$, so the theorem holds in the base case.

Inheritance Step: In the inheritance step we start with a derivation D . Say Δ is the set of unboxed assumptions occurring in D , and Δ_i is the set of unboxed assumptions occurring in D up to (and including) line number i . We then want to show that if we add another line to D with sentence ϕ sanctioned by rule R , then $\Delta^* \models \phi$, where Δ^* is the set of unboxed assumptions for the new line. (Notice that this requires more than the fact that the rules are truth preserving; we also have to attend to how we define a derivation. The fact that the rules are truth preserving is essential of course.) We need to consider each rule R of SD as its own case.

Recursive Assumption: The recursive assumption is that for all lines L_i in the derivation D , if ϕ is the sentence on the line, then $\Delta_i \models \phi$.

Assumption: Say we add another line to D with sentence ϕ sanctioned by *Assumption*. Note that the set Δ^* of unboxed assumptions for this new line are those in Δ plus ϕ . We know $\phi \models \phi$, and $\Delta, \phi \models \phi$ follows from this by monotonicity.

Repetition: Say ϕ already occurs somewhere in D , say on line number i . Then $\Delta_i \models \phi$ by the recursive assumption. Suppose we add another line to D with ϕ sanctioned by *Repetition*. By NDAP, $\Delta_i \subseteq \Delta^*$. So by monotonicity, $\Delta^* \models \phi$.

\vee -Intro: Assume we add another line to D with sentence ϕ sanctioned by \vee -Intro. Then there's some earlier line i with the sentence θ and ϕ is a disjunction with θ as one disjunct. We have that Δ_i is the set of unboxed assumptions of line i , and by NDAP $\Delta_i \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \theta$. So by monotonicity, $\Delta^* \models \theta$. Because the rule is truth preserving we know that ϕ is a disjunction with θ as one disjunct, $\theta \models \phi$. So by transitivity, $\Delta^* \models \phi$.

\wedge -Elim: Say we add another line to D with sentence ϕ sanctioned by \wedge -Elim. Then there's some earlier line i with the sentence $\theta_1 \wedge \dots \wedge \theta_m$ and ϕ is one of the conjuncts. As before, by NDAP $\Delta_i \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \theta_1 \wedge \dots \wedge \theta_m$. So by monotonicity, $\Delta^* \models \theta_1 \wedge \dots \wedge \theta_m$. And ϕ is one of the conjuncts of $\theta_1 \wedge \dots \wedge \theta_m$, so it follows that $\theta_1 \wedge \dots \wedge \theta_m \models \phi$. So by transitivity, $\Delta^* \models \phi$.

\sim -Elim: Suppose we add another line to D with sentence ϕ sanctioned by \sim -Elim. Then there's some earlier line i with the sentence $\sim\phi \rightarrow (\psi \wedge \sim\psi)$. As before, by NDAP $\Delta_i \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \sim\phi \rightarrow (\psi \wedge \sim\psi)$. So by monotonicity, $\Delta^* \models \sim\phi \rightarrow (\psi \wedge \sim\psi)$. Since the RHS of $\sim\phi \rightarrow (\psi \wedge \sim\psi)$ is false in all models (it's TFF), the conditional is true in a model \mathbf{m} only if the LHS is false in \mathbf{m} . So if the conditional is true in a model \mathbf{m} , ϕ is true in \mathbf{m} . In other words, $\sim\phi \rightarrow (\psi \wedge \sim\psi) \models \phi$. So by transitivity, $\Delta^* \models \phi$.

\sim -Intro: This case is very similar to the last and is left to the reader.

\rightarrow -Elim: Assume we add another line to D with sentence ϕ sanctioned by \rightarrow -Elim. Then there's two earlier lines i and j , and (say) line i has a sentence $\theta \rightarrow \phi$ and line j has sentence θ . By NDAP we have that $\Delta_i \subseteq \Delta^*$ and $\Delta_j \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \theta \rightarrow \phi$ and $\Delta_j \models \theta$. By monotonicity, $\Delta^* \models \theta \rightarrow \phi$ and $\Delta^* \models \theta$. Because the rule is truth preserving we know that $\theta, \theta \rightarrow \phi \models \phi$. So by transitivity, $\Delta^* \models \phi$.

\leftrightarrow -Elim: The argument for each of the two versions of \leftrightarrow -Elim is the same as that for \rightarrow -Elim.

\leftrightarrow -Intro: Say we add another line to D with sentence $\phi = \phi \leftrightarrow \theta$ sanctioned by \leftrightarrow -Intro. Then there's two earlier lines i and j , and (say) line i has a sentence $\theta \rightarrow \phi$ and line j has sentence $\phi \rightarrow \theta$. By NDAP we have that $\Delta_i \subseteq \Delta^*$ and $\Delta_j \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \theta \rightarrow \phi$ and $\Delta_j \models \phi \rightarrow \theta$. By monotonicity, $\Delta^* \models \theta \rightarrow \phi$ and $\Delta^* \models \phi \rightarrow \theta$. Because the rule is truth preserving we know that $\phi \rightarrow \theta, \theta \rightarrow \phi \models \phi \leftrightarrow \theta$. So by transitivity, $\Delta^* \models \phi$.

\wedge -Intro: Suppose we add another line to D with sentence $\phi = \phi_1 \wedge \dots \wedge \phi_m$ sanctioned by \wedge -Intro. Then there are m earlier lines numbered i_1, \dots, i_m

with, respectively, sentences ϕ_1, \dots, ϕ_m . By NDAP we have that $\Delta_{i_1} \subseteq \Delta^*, \dots, \Delta_{i_m} \subseteq \Delta^*$. By the recursive assumption, $\Delta_{i_1} \models \phi_1, \dots, \Delta_{i_m} \models \phi_m$. So by monotonicity, $\Delta^* \models \phi_1, \dots, \Delta^* \models \phi_m$. We now observe that $\phi_1, \dots, \phi_m \models \phi \wedge \dots \wedge \phi_m$. So by transitivity, $\Delta^* \models \phi$.

\vee -Elim: Assume we add another line to D with sentence ϕ sanctioned by \vee -Elim. Then there are $m + 1$ earlier lines numbered i_1, \dots, i_m, i_{m+1} with, respectively, sentences $\theta_1 \rightarrow \phi, \dots, \theta_m \rightarrow \phi$, and $\theta_1 \vee \dots \vee \theta_m$. By NDAP we have that $\Delta_{i_1} \subseteq \Delta^*, \dots, \Delta_{i_m} \subseteq \Delta^*$ and $\Delta_{i_{m+1}} \subseteq \Delta^*$. By the recursive assumption, $\Delta_{i_1} \models \theta_1 \rightarrow \phi, \dots, \Delta_{i_m} \models \theta_m \rightarrow \phi$ and $\Delta_{i_{m+1}} \models \theta_1 \vee \dots \vee \theta_m$. By monotonicity, $\Delta^* \models \theta_1 \rightarrow \phi, \dots, \Delta^* \models \theta_m \rightarrow \phi$ and $\Delta^* \models \theta_1 \vee \dots \vee \theta_m$. We now observe that $\theta_1 \rightarrow \phi, \dots, \theta_m \rightarrow \phi, \theta_1 \vee \dots \vee \theta_m \models \phi$. So by transitivity, $\Delta^* \models \phi$.

\rightarrow -Intro: Like *Assumption*, the assumptions change in \rightarrow -Intro. If the new line sanctioned by \rightarrow -Intro has sentence $\phi \rightarrow \theta$ and unboxed assumptions Δ^* , then earlier we have an assumption line (now in a box) that starts with ϕ and Δ^* as its other assumptions, and we have a line (now at the bottom of the box) with θ on it with assumptions ϕ, Δ^* . By the recursive assumption we have that $\phi, \Delta^* \models \theta$. Consider any model \mathbf{m} that makes Δ^* true; if it also makes ϕ true, then θ is true in \mathbf{m} as well and so is $\phi \rightarrow \theta$. If \mathbf{m} makes ϕ false, then $\phi \rightarrow \theta$ is true. (Notice that this step only works because we defined the conditional to be true when the LHS is false.) So if \mathbf{m} makes Δ^* true, it makes $\phi \rightarrow \theta$ true too. So, $\Delta^* \models \phi \rightarrow \theta$.

Closure Step: We have now covered all the generating cases for derivations. By the closure clause of the definition, we have proved soundness for all derivations. ■

Proof of Thm. 7.6, SL Soundness Theorem: Assume that Δ is a set of SL sentences. Assume $\Delta \vdash \phi$ and consider some derivation D of ϕ from Δ . Let Δ' be the set of sentences in Δ that appear as unboxed assumptions in D . By the soundness lemma (Thm. 7.7, on page 182), $\Delta' \models \phi$. It follows immediately by monotonicity that $\Delta \models \phi$. ■

7.2.2 Soundness of QD

In this section we prove that QD is also sound.

Theorem 7.11. QD Soundness Theorem: *QD is sound; i.e., for every set Δ of sentences of QL and every sentence ϕ of QL, if $\Delta \vdash \phi$ in SD, then $\Delta \models \phi$.*

The proof given in the last section of the SL Soundness Theorem (Thm. 7.6, on page 182) can be carried over to the QL Soundness Theorem. That proof relied on the monotonicity of entailment and the soundness lemma (Thm. 7.7, on page 182). It should be clear that entailment is also monotonic in the case of QL. Since QD is

just an extension of SD (it's just SD plus the rules for the quantifiers in table 6.44 on page 164), all we need to do to show that the soundness lemma holds for QD is add a case, for each new rule of QD, to the inheritance step of the proof of the soundness lemma for SD.

Proof of Thm. 7.7 for GQD:

Base Step: The base case has been covered in the proof for SD.

Inheritance Step: Just as in the proof for SD, in the inheritance step we start with a derivation D . Say Δ is the set of unboxed assumptions occurring in D , and Δ_i is the set of unboxed assumptions occurring in D up to (and including) line number i . We then want to show that if we add another line to D with sentence ϕ sanctioned by rule R , then $\Delta^* \models \phi$, where Δ^* is the set of unboxed assumptions for the new line. Again we need to consider each rule R of QD as its own case. Most of the rules have already been covered in the proof of SD, so we only need to cover the introduction and elimination rules for the quantifiers.

Recursive Assumption: The recursive assumption, as in the proof for SD, is that for all lines L_i in the derivation D , if ϕ is the sentence on the line, then $\Delta_i \models \phi$.

\forall -Elim: Say we add another line to D with sentence $\phi s/\beta$ sanctioned by \forall -Elim. Then there's some earlier line i with the sentence $\forall\beta\phi$. We have that Δ_i is the set of unboxed assumptions of line i , and by NDAP $\Delta_i \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \forall\beta\phi$. So by monotonicity, $\Delta^* \models \forall\beta\phi$.

Assume some model \mathbf{m} such that $\forall\beta\phi$ is true. By the def. of truth for \forall , $\phi t/\beta$ is true on all t -variants of \mathbf{m} . Notice that $\phi t/\beta$ and $\phi s/\beta$ are exactly the same, except that the latter has s substituted for t . These sentences satisfy condition (1) of Dragnet.

Now let's consider, in particular, the t -variant that assigns to t what \mathbf{m} assigns to s . Name that t -variant \mathbf{m}^t . The models \mathbf{m} and \mathbf{m}^t meet Dragnet condition (2). Thus, by Dragnet, $\phi t/\beta$ is true on \mathbf{m}^t iff $\phi s/\beta$ is true on \mathbf{m} . Therefore, $\phi s/\beta$ is true on \mathbf{m} .

Any model such that $\forall\beta\phi$ is true also makes $\phi s/\beta$ true. Thus, $\forall\beta\phi \models \phi s/\beta$. So by transitivity, $\Delta^* \models \phi s/\beta$.

\exists -Intro: Say we add another line to D with sentence $\exists\beta\phi$ sanctioned by \exists -Intro. Then there's some earlier line i with the sentence $\phi s/\beta$. Again we have that Δ_i is the set of unboxed assumptions of line i , and by NDAP $\Delta_i \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \phi s/\beta$. By monotonicity, $\Delta^* \models \phi s/\beta$.

Assume some model \mathbf{m} such that $\exists\beta\phi$ is false. By the def. of truth for \exists , there is no t -variant of \mathbf{m} that makes $\phi t/\beta$ true. Notice that $\phi t/\beta$ and $\phi s/\beta$ are exactly the same, except that the latter has s substituted for t . These sentences satisfy condition (1) of Dragnet.

Now let's consider, in particular, the t -variant that assigns to t what \mathbf{m} assigns to s . Name that t -variant \mathbf{m}^t . The models \mathbf{m} and \mathbf{m}^t meet Dragnet condition (2). Thus, by Dragnet, $\phi t/\beta$ is true on \mathbf{m}^t iff $\phi s/\beta$ is true on \mathbf{m} . Therefore, $\phi s/\beta$ is false on \mathbf{m} .

Any model that makes $\exists\beta\phi$ false also makes $\phi s/\beta$ false. Thus, $\phi s/\beta \models \exists\beta\phi$. So by transitivity, $\Delta^* \models \exists\beta\phi$.

\forall -Intro: Say we add another line to D with sentence $\forall\beta\phi$ sanctioned by \forall -Intro.

Then there's some earlier line i with the sentence $\phi s/\beta$. Again we have that Δ_i is the set of unboxed assumptions of line i , and by NDAP $\Delta_i \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \phi s/\beta$. By monotonicity, $\Delta^* \models \phi s/\beta$. However we know that $\phi s/\beta$ does not entail $\forall\beta\phi$, so we have to do some extra work and make use of the restrictions on the rule \forall -Intro.

Let \mathbf{m} be some model that makes all of Δ^* true; and let's assume for *reductio* that \mathbf{m} that makes $\forall\beta\phi$ false. One of the restrictions for \forall -Intro is that s must not occur in $\forall\beta\phi$. Hence, by the Free Choice Theorem, $\phi s/\beta$ is false on some s -variant of \mathbf{m} . Let's name that s -variant \mathbf{m}^s .

The other rule restriction for \forall -Intro is that s must not occur in Δ^* . The variant \mathbf{m}^s differs from \mathbf{m} only on the assignment to s ; otherwise, they make all the same assignments. Because s doesn't occur in Δ^* and \mathbf{m} makes all of Δ^* true, \mathbf{m}^s also makes all of Δ^* true. The assignment \mathbf{m}^s makes to s doesn't matter for this result.

Because $\Delta^* \models \phi s/\beta$, \mathbf{m}^s makes $\phi s/\beta$ true. But we had concluded that $\phi s/\beta$ is false on \mathbf{m}^s . We've inferred a contradiction. Our assumption that \mathbf{m} makes $\forall\beta\phi$ false must be wrong.

Therefore, if \mathbf{m} is a model that makes all of Δ^* true, then \mathbf{m} makes $\forall\beta\phi$ true as well. So, $\Delta^* \models \forall\beta\phi$.

\exists -Elim: Say we add another line to D with sentence θ sanctioned by \exists -Elim. Then there's some earlier line i with the sentence $\phi s/\beta \rightarrow \theta$ and an earlier line j with the sentence $\exists\beta\phi$. Again we have that Δ_i is the set of unboxed assumptions of line i and Δ_j the unboxed assumptions of line j . By NDAP $\Delta_i \subseteq \Delta^*$ and $\Delta_j \subseteq \Delta^*$. By the recursive assumption, $\Delta_i \models \phi s/\beta \rightarrow \theta$ and $\Delta_j \models \exists\beta\phi$. By monotonicity, $\Delta^* \models \phi s/\beta \rightarrow \theta$ and $\Delta^* \models \exists\beta\phi$. Again we have to do some extra work and make use of the restrictions on the rule \exists -Elim to show that $\Delta^* \models \theta$.

Let \mathbf{m} be some model that makes all of Δ^* true. Because $\Delta^* \models \exists\beta\phi$, \mathbf{m} also makes $\exists\beta\phi$ true. One of the rule restrictions for \exists -Elim is that s must not occur in $\exists\beta\phi$. Hence, by the Free Choice theorem, $\phi s/\beta$ is true on some s -variant of \mathbf{m} . Name that s -variant \mathbf{m}^s .

Another of the rule restrictions for \exists -Elim is that s must not occur in Δ^* . The variant \mathbf{m}^s makes all the same assignments as \mathbf{m} except in what it assigns to s . Because \mathbf{m} makes Δ^* true and Δ^* doesn't contain s , \mathbf{m}^s also makes Δ^* true. The assignment \mathbf{m}^s makes to s doesn't make any difference.

Thus, because $\Delta^* \models \phi s/\beta \rightarrow \theta$, \mathbf{m}^s makes $\phi s/\beta \rightarrow \theta$ true. We saw earlier that \mathbf{m}^s makes $\phi s/\beta$ true, so \mathbf{m}^s makes θ true as well (def. of truth, \rightarrow).

According to the third rule restriction for \exists -Elim, s must not occur in θ . Because \mathbf{m}^s makes θ true and s isn't in θ , \mathbf{m} also makes θ true. The assignment that \mathbf{m}^s makes to s is irrelevant.

So, we have shown that $\Delta^* \models \theta$.

Closure Step: We have now covered all the generating cases for derivations. By the closure clause of the definition, we have proved soundness for all derivations in QD.

■

7.3 Completeness

In this section we first prove the completeness of SD.

Definition 7.12. A derivation system D for L is *complete* iff for every finite set Δ of sentences of L and every sentence ϕ of L, if $\Delta \models \phi$, then $\Delta \vdash \phi$.

If we limit Δ to just the empty set, we get weak completeness:

Definition 7.13. A derivation system D for L is *weakly complete* iff for every sentence ϕ of L, if $\models \phi$, then $\vdash \phi$.

The following theorem can be proved using basic results we already have. For other systems of logic, what we are calling completeness and weak completeness are not equivalent. Because they are equivalent in our systems, we will not always distinguish them in what follows. Strong completeness also holds for our systems, but is not trivially equivalent to completeness. As in the first cases, there are systems that are complete but not strongly complete.

Theorem 7.14. *SD is weakly complete iff it's complete; and likewise for QD.*

Proof: (\Leftarrow) This direction of the biconditional is trivial. Assume that SD/QD is complete. Then for any finite set Δ , if $\Delta \models \phi$, then $\Delta \vdash \phi$. By definition, this includes the case when Δ is the empty set. Hence SD/QD is weakly complete.

(\Rightarrow) Assume that SD/QD is weakly complete: for any sentence ϕ , if $\models \phi$, then $\vdash \phi$. Now assume that, for some finite set Δ of sentences and sentence ϕ , $\Delta \models \phi$. Since Δ is finite, we can consider the conjunction of all the sentences in Δ . Let δ be this conjunction. We want to show that $\models \delta \rightarrow \phi$; to do so, assume that there's some model \mathbf{m} that makes $\delta \rightarrow \phi$ false. By the definition of truth for \rightarrow and \wedge , it follows that \mathbf{m} makes all the conjuncts of δ true and ϕ false. But that would mean that \mathbf{m} makes all the sentences in Δ true and ϕ false. But we assumed that $\Delta \models \phi$, so there's no model \mathbf{m} that makes $\delta \rightarrow \phi$ false. Hence $\models \delta \rightarrow \phi$, and so by weak completeness,

$\vdash \delta \rightarrow \phi$. It should be clear to the reader that if $\vdash \delta \rightarrow \phi$, then $\delta \vdash \phi$. Hence, $\delta \vdash \phi$. Finally, since $\Delta \vdash \delta$ and \vdash is transitive, $\Delta \vdash \phi$. ■

Definition 7.15. A derivation system D for L is *strongly complete* iff for every set Δ of sentences of L and every sentence ϕ of L , if $\Delta \models \phi$, then $\Delta \vdash \phi$.

Note that in the definitions the RHS of the biconditionals must hold even in the special case when Δ is the empty set and the special case when Δ is infinite. If we limit Δ so that it must be finite (but still allow it to be empty), we get (regular) completeness. Note that both SD and QD are strongly complete, but there is no simple theorem that uses results we already have which extends weak completeness to strong completeness in the way this theorem (Thm. 7.14) extends weak completeness to (regular) completeness.

Returning to strong completeness, letting Δ be infinite may seem problematic, since as we've defined them (def. 6.1, on page 134) derivations can only have finitely many lines. Hence, a derivation can only have finitely many assumptions. And, as we've defined the single turnstile, $\Delta \vdash \phi$ iff there's a derivation of ϕ from the sentences in Δ . But there's nothing problematic about letting Δ be infinite, because showing that there's a derivation of ϕ from the sentences in Δ doesn't require that the derivation use *all* the sentences in Δ as assumptions. In general, even when Δ is finite, any derivation of ϕ from some subset of sentences in Δ will show that $\Delta \vdash \phi$. So, if Δ is infinite and $\Delta \models \phi$, if the derivation system D is complete we'll know that ϕ can be derived from some finite subset of sentences of Δ .¹

We'll prove that SD is weakly complete and then show that, for SD, completeness and weak completeness are equivalent. By theorem 7.14 (on the previous page), this will be sufficient to show that SD is complete. The equivalent statement is:

Theorem 7.16. The SD Weak Completeness Lemma: *For any sentence ϕ of SD, either $\phi \vdash A \wedge \sim A$, or ϕ is true in some model \mathbf{m} .*

Before proving the theorem, it will be useful to introduce a new exchange rule for QD and then show that anything we can derive using QD and this rule can be derived using QD alone. We call the rule \leftrightarrow -Exchange. (Note that every application

¹Before turning to the proofs of these theorems, some historical background might be of interest. As mentioned above (Sec. 4.1.1), quantificational languages were first developed by Frege, Peirce and Mitchell in the 1870's and 1880's. But it wasn't until David Hilbert and Wilhelm Ackermann published their hugely influential text *Grundzüge der theoretischen Logik* (Principles of Mathematical Logic) in 1928 that the question of completeness was clearly formulated. While Kurt Gödel, in his 1929 doctoral dissertation (republished in 1930), is widely accepted as the first person to prove that quantificational logic is strongly complete, Church (1956: 291, fn.464) reports that the Jacques Herbrand's dissertation in 1930 had the essential material for the same proof. Further, completeness follows from results of Skolem (1967 [1928]), but since the question of completeness hadn't been clearly raised yet no one seems to have noticed. Leon Henkin (1949) later developed a method of proving completeness different from Gödel's. Henkin's approach is probably the most common one used today in logic textbooks, but the proof we give here is a constructive proof closer to Gödel's original. (Ours owes much to Willard Quine's completeness proof (1982).)

of \leftrightarrow -Exchange is truth preserving, as the last problem in exercise 2.8.9, on page 57, extends theorem 6.9, on page 158, to it.) It's given in table 7.1. Recall from section

Name	Given	May Add
\leftrightarrow -Exchange	$\theta \leftrightarrow \psi$	$(\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)$
	$(\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)$	$\theta \leftrightarrow \psi$

Table 7.1: \leftrightarrow -Exchange

6.4.3 that all we need to do to show that anything that can be derived using QD and this rule can be derived using just QD is to prove the following:

Theorem 7.17. *Any two QL formulas got by substituting other QL formulas into the may-add and given schemas of \leftrightarrow -Exchange are provably equivalent; that is, $\vdash \forall((\theta \leftrightarrow \psi) \leftrightarrow ((\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)))$.*

Proof: We show that $\vdash \forall((\theta \leftrightarrow \psi) \leftrightarrow ((\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)))$ by giving a derivation schema, which for any two formulas θ and ψ will result in the needed derivation. (Note that to save space $q = n + m$.)

1.	$(\theta \leftrightarrow \psi)c_1 \dots c_m/x_1 \dots x_m$	Assume
2.	$(\sim\theta \leftrightarrow \sim\psi)c_1 \dots c_m/x_1 \dots x_m$	\sim/\leftrightarrow -Intro, 1
3.	$\sim(\theta \wedge \psi)c_1 \dots c_m/x_1 \dots x_m$	Assume
4.	$(\sim\theta \vee \sim\psi)c_1 \dots c_m/x_1 \dots x_m$	DeM, 3
5.	$\sim\theta c_1 \dots c_m/x_1 \dots x_m$	Assume
6.	$\sim\psi c_1 \dots c_m/x_1 \dots x_m$	\leftrightarrow -Elim, 2, 5
7.	$(\sim\theta \wedge \sim\psi)c_1 \dots c_m/x_1 \dots x_m$	\wedge -Intro, 5, 6
8.	$(\sim\theta \rightarrow (\sim\theta \wedge \sim\psi))c_1 \dots c_m/x_1 \dots x_m$	\rightarrow -Intro, 5–7
9.	$\sim\psi c_1 \dots c_m/x_1 \dots x_m$	Assume
10.	$\sim\theta c_1 \dots c_m/x_1 \dots x_m$	\leftrightarrow -Elim, 2, 9
11.	$(\sim\theta \wedge \sim\psi)c_1 \dots c_m/x_1 \dots x_m$	\wedge -Intro, 9, 10
12.	$(\sim\psi \rightarrow (\sim\theta \wedge \sim\psi))c_1 \dots c_m/x_1 \dots x_m$	\rightarrow -Intro, 9–11
13.	$(\sim\theta \wedge \sim\psi)c_1 \dots c_m/x_1 \dots x_m$	\vee -Intro, 4, 8, 12
14.	$(\sim(\theta \wedge \psi) \rightarrow (\sim\theta \wedge \sim\psi))c_1 \dots c_m/x_1 \dots x_m$	\rightarrow -Intro, 3–13
15.	$(\sim\sim(\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi))c_1 \dots c_m/x_1 \dots x_m$	\rightarrow/\vee -Exch., 14
16.	$((\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi))c_1 \dots c_m/x_1 \dots x_m$	$\sim\sim$ -Elim, 15

17.	$[(\theta \leftrightarrow \psi) \rightarrow$	
	$((\theta \wedge \psi) \vee (\sim \theta \wedge \sim \psi))]c_1 \dots c_m / x_1 \dots x_m$	\rightarrow -Intro, 1–16
18.	<div style="border: 1px solid black; padding: 10px; display: inline-block;"> $((\theta \wedge \psi) \vee (\sim \theta \wedge \sim \psi))c_1 \dots c_m / x_1 \dots x_m$ \vdots $(\theta \leftrightarrow \psi)c_1 \dots c_m / x_1 \dots x_m$ </div>	Assume
n .		
$n + 1$.	$[(\theta \wedge \psi) \vee (\sim \theta \wedge \sim \psi)) \rightarrow$	
	$(\theta \leftrightarrow \psi)]c_1 \dots c_m / x_1 \dots x_m$	\rightarrow -Intro, 18– n
$n + 2$.	$[(\theta \leftrightarrow \psi) \leftrightarrow$	\leftrightarrow -Intro, 17,
	$((\theta \wedge \psi) \vee (\sim \theta \wedge \sim \psi))]c_1 \dots c_m / x_1 \dots x_m$	$n + 1$
	\vdots	
$q + 2$.	$\forall((\theta \leftrightarrow \psi) \leftrightarrow ((\theta \wedge \psi) \vee (\sim \theta \wedge \sim \psi)))$	\forall -Intro, $q + 1$

Note that we have left steps 18– n for the reader; this is just the derivation of the other conditional needed for \leftrightarrow -Intro on line $n + 2$. Also note that the last steps, lines $n + 3$ to the end, are all \forall -Intro meant to eliminate the constants c_1, \dots, c_m . ■

Proof of Thm. 7.16: To prove the theorem, we shall describe an algorithm for applying the rules of SD^+ and \leftrightarrow -Exchange that takes a SL sentence ϕ and either halts in a derivation of $A \wedge \sim A$, or halts with a sentence in DNF for which there is some model \mathbf{m} that makes ϕ true. Since a sentence can be derived using the rules of SD^+ and \leftrightarrow -Exchange iff it can be derived using the basic rules of SD, this will be sufficient to prove the theorem.

The algorithm begins with ϕ as an assumption on line 1. The algorithm then applies the method studied earlier in section 2.6.2 (on page 48) to produce a sentence ϕ' in DNF that's TFE to ϕ . We have to show that each step of the earlier method can be carried out in steps using the rules of SD^+ and \leftrightarrow -Exchange. The earlier method proceeded in three stages.

Step A:

- (1) If a subsentence of ϕ has \rightarrow as its main connective, i.e. if $\phi = \theta \rightarrow \psi$, replace the subsentence by $\sim \theta \vee \psi$. Repeat as necessary to obtain a sentence ϕ^* without conditionals. Each of these steps are sanctioned by \rightarrow/\vee -Exchange.
- (2) If a subsentence of ϕ has \leftrightarrow as its main connective, i.e. if $\phi = \theta \leftrightarrow \psi$, it is replaced with the subsentence $(\theta \wedge \psi) \vee (\sim \theta \wedge \sim \psi)$. Repeat as necessary

to obtain a sentence ϕ^{**} without biconditionals. Each of these steps are sanctioned by \leftrightarrow -*Exchange*.

Step B: In the case where ϕ^{**} contains a subsentence whose main connective is negation and which contains other connectives, we replace that subsentence by the following steps:

- (1) Replace $\sim\sim\theta$ by θ ; this step is sanctioned by $\sim\sim$ -*Elim*.
- (2) Replace $\sim(\theta \wedge \psi)$ by $\sim\theta \vee \sim\psi$; this step is sanctioned by *DeM*.
- (3) Replace $\sim(\theta \vee \psi)$ by $\sim\theta \wedge \sim\psi$; this step is sanctioned by *DeM*.

Repeat as necessary to obtain a sentence ϕ^{***} in which negations govern nothing but sentence letters.

Step C: The only thing that could prevent ϕ^{***} from being in DNF is that some conjunctions govern some disjunctions, i.e., there is a subsentence of the form $\theta \wedge (\psi_1 \vee \dots \vee \psi_n)$, or the reverse $(\psi_1 \vee \dots \vee \psi_n) \wedge \theta$. Those subsentences can each be replaced by the equivalent sentence $(\theta \wedge \psi_1) \vee \dots \vee (\theta \wedge \psi_n)$ or $(\psi_1 \wedge \theta) \vee \dots \vee (\psi_n \wedge \theta)$. These steps are sanctioned by *Distribution*.

Applying the above steps A, B, and C will provide us a derivation starting with ϕ as an assumption (and no other assumptions) and ending with a DNF sentence that's TFE to ϕ . We now have two possibilities:

Case 1: Every disjunct contains a sentence letter and the negation of that sentence letter. That is, each disjunction has the form $(\psi_1 \wedge \dots \wedge \psi_i \wedge \dots \wedge \sim\psi_i \wedge \dots \wedge \psi_n)$; for example: $(A \wedge B \wedge C \wedge \sim E \wedge \sim B \wedge \sim K)$.

Case 2: At least one disjunct contains no sentence letter such that the negation of the sentence letter is also in the disjunct.

We can show in case 1 that the original sentence leads to a contradiction. First, we observe that any conjunction that contains a sentence letter and its negation leads to a contradiction by repeated steps of \wedge -*Elim*. Thus we can derive the negation of any such conjunction using \sim -*Intro*. So, if the last line of our derivation so far is of the form $(\psi_1 \vee \dots \vee \psi_n)$ and each ψ_i contains a sentence letter and the negation of that sentence letter, then we can add to the derivation lines that establish the negation of each ψ_i . Thus by $n - 1$ steps of *D.S.* we get a single ψ_i by itself with only the first line as an assumption. Since by hypothesis in this case ψ_i leads to a contradiction, we can show the initial assumption leads to a contradiction.

Generally, the procedure will look like this:

ϕ	<i>Assume</i>
\vdots	
$\psi_1 \vee \dots \vee \psi_n$	
<div style="border: 1px solid black; padding: 5px; display: inline-block;"> ψ_1 \vdots $\theta_1 \wedge \sim\theta_1$ </div>	<i>Assume</i>
$\psi_1 \rightarrow (\theta_1 \wedge \sim\theta_1)$	\rightarrow -Intro
$\sim\psi_1$	\sim -Intro
\vdots	
<div style="border: 1px solid black; padding: 5px; display: inline-block;"> ψ_n \vdots $\theta_n \wedge \sim\theta_n$ </div>	<i>Assume</i>
$\psi_n \rightarrow (\theta_n \wedge \sim\theta_n)$	\rightarrow -Intro
$\sim\psi_n$	\sim -Intro
$\psi_1 \vee \dots \vee \psi_{n-1}$	<i>D.S.</i>
$\psi_1 \vee \dots \vee \psi_{n-2}$	<i>D.S.</i>
$\psi_1 \vee \dots \vee \psi_{n-3}$	<i>D.S.</i>
\vdots	
$\psi_1 \vee \psi_2$	<i>D.S.</i>
ψ_1	<i>D.S.</i>
$\sim\psi_1$	<i>Rep.</i>
$A \wedge \sim A$	<i>A.C.</i>

We can show in case 2 that we can find a model that makes all sentences in the derivation true, starting with the last. We choose the disjunction that does not contain

a sentence letter and its negation (if there is more than one, it doesn't matter which we choose), and we construct a model \mathbf{m} by assigning T to each sentence letter that occurs positively (without a negation in front) in the conjunction and F to each sentence letter that occurs negatively (with a negation in front). We can do this since none occur in both modes.

This model makes each element of the conjunction true and thus makes the entire conjunction true. Since the sentence containing it is a disjunction, this is sufficient to make the entire sentence true. Thus we can make the last line of the derivation true. Observe now that all of the steps we used in the derivation were replacement of provably equivalence sentences; that is, they used exchange shortcut rules. Thus, we know that we could also construct a derivation by 'turning this proof upside down', so to speak. In other words, we could construct a new derivation, with the last step of the original derivation as the initial assumption step. Then we could use the exchange shortcut rules to work back to ϕ of the original derivation.

Thus, by soundness, we know that if the first sentence of the 'upside-down derivation' (the sentence in DNF that was at the bottom) is true in a model, then so is everything that can be derived from it, including our original sentence ϕ that is now at the end of the inverted derivation. Therefore, ϕ is true in some model. ■

Theorem 7.18. Weak SD Completeness Theorem: *For all SL sentences ϕ : if $\models \phi$, then $\vdash \phi$ in SD.*

Proof: We apply the method above from the SD Completeness Lemma to the negation of ϕ . This either produces a derivation of a contradiction from $\sim\phi$, in which case we can prove ϕ by adding two more steps justified by \rightarrow -Intro and \sim -Elim, or it produces a model that makes $\sim\phi$ true and that therefore makes ϕ false. So, ϕ is either false in some model or is derivable in SD. ■

Finally, as a corollary we get:

Theorem 7.19. SD Completeness Theorem: *For every finite set Δ of sentences of SL and every sentence ϕ of SL, if $\Delta \models \phi$, then $\Delta \vdash \phi$ in SD.*

Proof: This follows immediately from the Weak SD Completeness Theorem and theorem 7.14 (on page 188). ■

7.4 Completeness of QD

In this section we shall prove that QD is complete. We would like to use the same kind of strategy for QD we did for SD, so we have deal with the quantifiers. Unfortunately, the quantifiers prevent us from proving the analogue of DNF for QL. For example: $\forall x(Hx \vee Gx)$ is not equivalent to $\forall xHx \vee \forall xGx$.

To get around problems like this, we must show we can move all of the quantifiers to the front of the sentence. This has the advantage of separating the quantifier parts of the logical structure from the SL parts.

7.4.1 Prenex Definition and Steps

Definition 7.20. A sentence ϕ of QL is in *prenex normal form* iff every quantifier is in initial position, or, in other words, the scope of all quantifiers is greater than that of any non-quantifier connective.

Theorem 7.21. Prenex Normal Form Theorem: *For all sentence θ of QL, there is a provably equivalent sentence θ^* in prenex normal form; that is, θ^* is in prenex normal form and $\vdash \theta \leftrightarrow \theta^*$ in QD.*

Proof: As with DNF we have a set of steps for turning sentence θ into a sentence θ^* in prenex normal form. First we give the steps, and then show that each step can be sanctioned either by QN, \leftrightarrow -Exchange, or an exchange rule that can be introduced. (We'll call these new exchange rules the *Prenex Exchange Rules*.) Because all the steps in the process are justified by exchange rules, we can either read the resulting series of steps top-down as a derivation of θ^* from θ , or bottom-up as a derivation of θ from θ^* . So, we'll have shown that $\vdash \theta \leftrightarrow \theta^*$ in the derivation system consisting of \leftrightarrow -Exchange and the Prenex Exchange Rules. But, as with all the other exchange rules anything that can be derived using the Prenex Exchange Rules can be derived in QD alone; so, this will be sufficient to show that $\vdash \theta \leftrightarrow \theta^*$ in QD. First, the steps are:

- (1) Replace biconditionals with disjunctions of conjunctions; i.e. replace $\phi \leftrightarrow \psi$ with $(\phi \wedge \psi) \vee (\sim\phi \wedge \sim\psi)$.
- (2) Rewrite any variables that occur bound by more than one quantifier.
- (3) Move the first quantifier not in prenex position one step towards the front by the following principles. Repeat this step as often as necessary. Keep in mind that you can't move forward a quantifier that binds the variable ' x ' if it will now have within its scope a new subformula that has a free ' x '. But we have prevented that problem by eliminating potentially clashing variables in Step 2.

Replace	by
$((\#x)\theta \wedge \psi)$	$(\#x)(\theta \wedge \psi)$
$(\theta \wedge (\#x)\psi)$	$(\#x)(\theta \wedge \psi)$
$((\#x)\theta \vee \psi)$	$(\#x)(\theta \vee \psi)$
$(\theta \vee (\#x)\psi)$	$(\#x)(\theta \vee \psi)$
$(\theta \rightarrow (\#x)\psi)$	$(\#x)(\theta \rightarrow \psi)$
$(\exists x\theta \rightarrow \psi)$	$\forall x(\theta \rightarrow \psi)$
$(\forall x\theta \rightarrow \psi)$	$\exists x(\theta \rightarrow \psi)$

$\sim\exists x\theta$	$\forall x\sim\theta$
$\sim\forall x\theta$	$\exists x\sim\theta$

Note that $(\#x)$ is just a dummy quantifier standing for either; replacement is the same for both quantifiers. Also, we use ‘ x ’ in the chart above, but the same principles hold for quantifiers with any other variable.

After applying these steps to a sentence θ we will get a sentence θ^* that is in prenex normal form.² We now have to show that each step can be sanctioned by an exchange rule. Step (1) is straightforward, since obviously it will be sanctioned by \leftrightarrow -*Exchange*. But steps (2) and (3) we need new rules (although the replacements involving negations in (3) can be handled with *QN*). The most straightforward strategy is to read the needed exchange rules right off the steps. Thus, the Prenex Exchange Rules are given in the following chart.

Name	Given	May Add
α/β - <i>Exch</i>	$(\#\alpha)\phi$	$(\#\beta)\phi\beta/\alpha$
<i>Q Shuffling</i>	$((\#x)\theta \wedge \psi)$	$(\#x)(\theta \wedge \psi)$
	$(\theta \wedge (\#x)\psi)$	$(\#x)(\theta \wedge \psi)$
	$((\#x)\theta \vee \psi)$	$(\#x)(\theta \vee \psi)$
	$(\theta \vee (\#x)\psi)$	$(\#x)(\theta \vee \psi)$
	$(\theta \rightarrow (\#x)\psi)$	$(\#x)(\theta \rightarrow \psi)$
	$(\exists x\theta \rightarrow \psi)$	$\forall x(\theta \rightarrow \psi)$
	$(\forall x\theta \rightarrow \psi)$	$\exists x(\theta \rightarrow \psi)$

Table 7.5: Prenex Exchange Shortcut Rules for QD

Now all that’s left to show is that anything that can be derived using the Prenex Exchange Rules can be derived using the basic rules of QD alone. Recall from section 6.4.3 that all we need to do to show this is to prove the following:

Theorem 7.22. *For all Prenex Exchange Rules R , any two QL formulas got by substituting other QL formulas into the may-add and given schemas of R are provably equivalent.*

We leave the proof of this theorem to the reader, since as with the other exchange

²For more discussion of Prenex Form, see Kleene 2002 [1967]: 132, Hodges 2001b: 54, 2001a: 30.

rules it just involves writing down the appropriate derivation schemas. ■

7.4.2 The Strategy for Proving QD Completeness

Our goal is to prove the strong completeness of QD; that is, we want to prove that for any set Δ of QL sentences and QL sentence ϕ , if $\Delta \models \phi$, then $\Delta \vdash \phi$. Our strategy will be to first prove the completeness of QD and then show how to modify the method to prove the strong completeness. Our strategy for proving completeness will be to show that for any sentence we can either (1) find a derivation of it or (2) prove that there is a model that makes it false. (This part of the strategy is more or less the same as what we did to show that SD is complete.) In other words, ϕ is either derivable or not quantificationally true, from which it immediately follows that if ϕ is quantificationally true, then it is derivable.

The method, in brief, is to negate the sentence ϕ and begin a derivation. Then we transform the negation of the sentence into prenex normal form, using the steps outlined in section 7.4.1. Next we transform the inner part of the sentence (remember the quantifiers are all up front) into DNF form, using our standard method for that (see Sec. 2.6.2, on page 48). This will not introduce any new assumptions. We then systematically take instances of the bound variables and try to derive a contradiction. If we can derive a contradiction we can then (assuming all goes well) use \sim -Elim to obtain a derivation of ϕ .

We must be very systematic since we have to be sure that if we get a contradiction we can derive it from the initial sentence ϕ , and that if we do not get a contradiction we have not overlooked anything and that we can show the existence of a model making the sentence on the first line, ϕ , true.

It is important that we know the form of the sentence we have reached and are able to prescribe a uniform systematic method. The sentence has been highly standardized; there are no biconditionals or conditionals (these have been eliminated in early steps of the transformation), negations govern only atomic sentences, and conjunctions govern only atomic sentences or their negations. These last, atomic sentence and their negations, are called *ions*. We say an ion occurs *positively* iff it's an atomic sentence without a negation, and it occurs *negatively* iff it's a negated atomic sentence. We will call the quantifier free part of the original sentence the *matrix*. It is usually not a sentence since it may have free variables. We will call the sentences that are obtained from the matrix by substitution in the process of constructing the derivation *matrix instances*. To put some of our jargon together, the matrix of the sentence will consist of disjunctions of conjunctions of ions.

7.4.3 The Method and Completeness Lemmas

In this section we describe the method sketched above. Given a sentence θ , the Method either produces a derivation of θ or indicates a model that makes it false:

Step 0: Write $\sim\theta$ on line 1 as an assumption. Then first apply the prenex steps to

put $\sim\theta$ in Prenex Normal Form (PNF). Next, apply the disjunctive normal form steps to the inner, quantifier-free part of the sentence until it's in DNF. At this point we'll have a sentence $(\sim\theta)^*$ that's in what we'll call *prenex disjunctive normal form* (PDFNF).

Step 1: We continue the derivation operating on $(\sim\theta)^*$, the PDFNF of the sentence we're concerned with. If this PDFNF is a universal statement and contains no constants we write as the next line the instance of it we obtain by eliminating the quantifier and substituting the constant a for the previously bound variable; these steps are sanctioned by \forall -Elim. This step is only done once, whereas the next three steps generally require repeated recursive applications.

Step 2: For every universal sentence that appears thus far in the derivation, we add *all new instances* that can be formed with constants that occur earlier in the derivation; these steps are sanctioned by \forall -Elim. E.g., if $\forall x\phi$ appears on a line and the constant c appears anywhere (earlier) in the derivation, then if we have not taken an instance of ϕ with c yet (i.e., $\phi c/x$), we do so. As a practical matter, this means that it is useful in following the method to keep track somewhere of the constants used at each stage and of which constants have been used to instantiate which universal statements. Note that in this step we are taking new instances with old constants and that we are not adding any new assumptions. We may, however, be adding new existentials.

Step 3: For every existential sentence that appears in the derivation for which no instance has been added yet, add an instance using the first constant which *does not occur in any previous assumption*. Note that 'instance' is to be taken very strictly here. The fact that we instantiated $\exists xKxa$ with Kba takes care of that existential, but if we later add the sentence $\exists xKxb$ then we must add an instance of it. The rule which sanctions these steps will be *Assumption*. We will eventually discharge these premises by \rightarrow -Elim and \exists -Elim if we get to a contradiction. It is in anticipation of this eventuality that we carefully chose a constant which does not occur in any previous assumption. Note that with this step we are adding new instances with new constants in new assumptions.

Step 4: Determine whether the conjunction of the *instances* of the matrix in the derivation thus far are contradictory. Officially, the way to do this is to take the conjunction of them all by \wedge -Intro, use *Distribution* to get the conjunction into DNF and check whether every disjunct contains a contradiction. If so, then by a process of \forall -Elim and *Any Contradiction* we can eventually produce the line $A \wedge \sim A$.

This step can be cumbersome in practice. We will give some unofficial short cuts to make this more manageable soon.

Step 5:

- (1) If the matrices are contradictory (see step 4) we stop.
- (2) Or if the conjunction of the matrix instances is consistent and the last applications of Steps 2 and 3 produce no new sentences, we stop.
- (3) Or if the conjunction of the matrix instance is consistent and the last applications of Steps 2 and 3 produced new sentences, then we return to Step 2 and reapply those steps.

There are three possible outcomes of applying this method to a sentence:

- (1) The method might reach a contradiction.
- (2) The method might stop without a contradiction.
- (3) The method might generate new sentences perpetually without contradiction.

We will show first that if a contradiction is reached we can construct a derivation of θ .

Theorem 7.23. Derivational Lemma: *If the Method starts with $\sim\theta$ and produces a contradiction, then there is a derivation of θ .*

Proof: Step 3 left us with $A \wedge \sim A$ on a line with its assumptions being those of the matrices. We want to shift those assumptions so that we end up with the contradiction from the first assumption, $\sim\theta$, alone. We know by considering our method that other assumptions entered only by Step 2, where we added instances of existentials using new constants. We eliminate the last assumption by a \rightarrow -Intro. We know that this eliminated assumption introduced a *new* constant from an existential. Therefore we know that this constant did not appear in any earlier assumption or in the existential of which we are taking an instance. It also (obviously) does not occur in $A \wedge \sim A$. Thus we are allowed to use the rule \exists -Elim on the existential claim from which we assumed that instance.

So we derive the contradiction $A \wedge \sim A$ again, by \exists -Elim. We continue this process, repeating $A \wedge \sim A$ as often as necessary to shift the dependence back to the assumption on line 1. This gives us a derivation of $A \wedge \sim A$ from the first assumption, $\sim\theta$, only.

We then add two more lines: $\sim\theta \rightarrow (A \wedge \sim A)$, sanctioned by \rightarrow -Intro, and θ , sanctioned by \sim -Elim. Thus we have a derivation of θ from no assumptions. ■

We have shown that if we obtain a contradiction in the derivation process we can derive the original sentence that interests us.

We must now show that if we do not obtain a contradiction (whether or not the method stops), then there is a model that makes $\sim\theta$ true (and hence makes θ false).

Before giving the rigorous version of the construction of the model, we will present some of the ideas in a more concrete context. If we consider a sentence such as $(Ka \wedge \sim Gb) \vee (\sim Kb \wedge Hc)$ we can observe several things. First, each disjunct is satisfiable iff no ion occurs both positively and negatively in it. It is obvious that a conjunction that includes a sentence and its negation cannot be satisfied, but we can

show for a conjunction of ions that that is the only way in which it can fail to be satisfiable. For example, we can make Ka and $\sim Gb$ true by letting K be interpreted as the set of even numbers, G the set of numbers divisible by 10 and letting 'a' be assigned 2 and 'b' be assigned 7.

Of course several such sentences taken together produce different results. E.g., as we saw above $(Ka \wedge \sim Kb \wedge \sim Gc) \vee (Gb \wedge \sim Gc)$ is satisfiable, as is $(Kb \wedge \sim Kc \wedge \sim Gb) \vee (Gc \wedge \sim Gd)$, but the two together (taken as a conjunction) are not. The reason is that while each disjunct of the first sentence is self-consistent, it cannot be true simultaneously with either of the disjuncts of the second sentence. If we have a series of disjunctions then they are simultaneously satisfiable only if we can find a way of picking a disjunct from each one in such a way that all the chosen disjuncts can be true together.

This is relevant to the task at hand because we know that all of the non-quantified sentences in our derivation are in DNF and are thus disjunctions of conjunctions of ions. We are calling the quantifier free part of the original sentence the matrix. It is usually not a sentence since it may have free variables. The sentences that are obtained from the matrix by substitution in the process of constructing the derivation are the matrix instances. We will use the notation $M_{i,j}$ for the disjuncts of the matrix instances, specifically the disjuncts of the first matrix instance will be $M_{1,1}, M_{1,2}, \dots, M_{1,m}$. Thus the first matrix instance is $M_{1,1} \vee M_{1,2} \vee \dots \vee M_{1,m}$. The matrix instances that appear in the derivation can be listed in an array:

$$\begin{array}{c} M_{1,1} \vee M_{1,2} \vee \dots \vee M_{1,m} \\ M_{2,1} \vee M_{2,2} \vee \dots \vee M_{2,m} \\ \vdots \\ M_{n,1} \vee M_{n,2} \vee \dots \vee M_{n,m} \end{array}$$

Note that if the the method never stops, then this array will be infinitely long.

The matrices are jointly consistent iff there is a way of picking an $M_{i,j}$ from each matrix instance so that the conjunction of those $M_{i,j}$ contains no atomic sentence and its negation. In one direction this is easy to see: if there is no way of choosing a disjunct from each matrix instance that does not end up with an atomic sentence and its negation among the chosen sentences then the set of instances is inconsistent.

To show that all instances are satisfiable when such a selection can be made without choosing a sentence and its negation will take some proving. In order to do this we will need to define the *master matrix list* M . We will first choose (if there is more than one) a set of disjuncts $M_{i,j}$ (including one from each matrix instance M_i) that does not contain any atomic sentence and its negation. This will be a set of conjunctions of atomic sentences and negations of atomic sentences. Our master matrix list M simply

consists of all these atomic sentences and negated atomic sentences. Note that since the $M_{i,j}$ selections must be consistent no atomic sentence that appears unnegated also appears negated.

Definition 7.24. Given a master matrix list M , the *matrix model of M* is the model \mathbf{m}_M such that:

- (1) The universe of \mathbf{m}_M contains one natural number for each constant that appears in M , and $\mathbf{m}_M(a) = 1$, $\mathbf{m}_M(b) = 2$, $\mathbf{m}_M(c) = 3$, $\mathbf{m}_M(d) = 4$, and so on; $\mathbf{m}_M(t) = 1$ for any constant t that doesn't appear in M .
- (2) For each m -place predicate P , $\mathbf{m}_M(P)$ is the set of m -tuples of natural numbers $\langle n_1, \dots, n_m \rangle$ such that $\mathbf{m}_M(t_1) = n_1, \dots, \mathbf{m}_M(t_m) = n_m$ and $Pt_1 \dots t_m$ appears on the list M .
- (3) Assignments are only made if justified by these principles.

A bit more informally, we list the constants that occur on the master matrix list. \mathbf{m}_M has a universe that contains as many natural numbers as constants used. We assign to each constant that occurs on the list the natural number that indicates its place in the order, i.e. 1 to a , 2 to b , and so on. Note that this produces a *census*. Each 1-place predicate is assigned the set of numbers associated with the constants such that an instance of the predicate followed by that constant appears on the master matrix list M . Each 2-place predicate is assigned the set of pairs of numbers associated with constants such that an instance of the predicate followed by that pair of constants appears on the master matrix list M . E.g., if Kab , Kbc , and Kde appear on M , then $\mathbf{m}_M(K)$ is assigned $\{\langle 1, 2 \rangle, \langle 2, 3 \rangle, \langle 4, 5 \rangle\}$. Assignments are made in a similar fashion for n -placed predicates for $n > 2$.³

Theorem 7.25. The Method Lemma 1: *The matrix model \mathbf{m}_M makes true all sentences on the master matrix list M .*

Proof: By construction, if an atomic sentence appears on the list we decided to put the relevant pair, triple, or whatever, of numbers in the set assigned to the predicate letter. For each negated atomic sentence on the list we know that we would not put the relevant pair, triple, or whatever, in the model of the predicate letter unless the atomic sentence which is being negated also appeared. But that never happened because M is consistent by hypothesis. ■

Theorem 7.26. The Method Lemma 2: *All matrix instances in the derivation are true in the matrix model \mathbf{m}_M .*

³Note that if the method never stops, then the master matrix list M will be infinite and we won't actually be able to write down the matrix model \mathbf{m}_M . But this isn't a problem, the matrix model \mathbf{m}_M still exists, even if we can't write it down.

Proof: By Lemma 1 (Thm. 7.25), all sentences on the master matrix list M are true, and we included all the conjuncts of at least one disjunct $M_{i,j}$ from each matrix instance in forming the master list. ■

Theorem 7.27. The Method Lemma 3: *All quantified sentences in the derivation are true in the matrix model \mathbf{m}_M .*

Informally, every universal has been instantiated with all the relevant constants, and every existential by at least one. Since \mathbf{m}_M is a census, it follows that all the sentences in the derivation are true.

More rigorously...

Proof: We prove this lemma using a recursive proof on the number of quantifiers in each sentence.

Base Case: The base case is the case of the sentences with $k = 0$ quantifiers. But these sentences are just the matrix instances in the derivation. We already proved in The Method Lemma 2 (Thm. 7.26) that all these sentences are true the matrix model \mathbf{m}_M , so the base case is complete.

Inheritance Step:

Recursive Assumption: Our recursive assumption is that all sentences in the derivation with less than k quantifiers are true in the matrix model \mathbf{m}_M .

Existential Quantifier: Say θ is a sentence appearing in the derivation of the form $\exists\alpha\phi$, where ϕ is a formula with $k - 1$ quantifiers. Step 3 of the method guarantees that the sentence $\phi t/\alpha$, for some constant t , appears somewhere in the derivation. This sentence $\phi t/\alpha$ has $k - 1$ quantifiers, so by the recursive assumption it is true in the matrix model \mathbf{m}_M . But then the existentially quantified sentence $\exists\alpha\phi$ is true in \mathbf{m}_M as well. (To show this rigorously, consider the s -variant of \mathbf{m}_M , \mathbf{m}^{s_1} , that assigns the same element of the universe of \mathbf{m}_M to s as \mathbf{m}_M assigns to the constant t . Now by the Dragnet Theorem, Thm. 4.9 on page 95, since \mathbf{m}_M makes $\phi t/\alpha$ true, the sentence $\phi s/\alpha$ is true on \mathbf{m}^{s_1} . It follows from this that $\exists\alpha\phi$ is true on \mathbf{m}_M .)

Universal Quantifier: Say θ is a sentence appearing in the derivation of the form $\forall\alpha\phi$, where ϕ is a formula with $k - 1$ quantifiers. All instances $\phi t/\alpha$ which appear in the derivation have $k - 1$ quantifiers, and so by the recursive hypothesis are true in the matrix model \mathbf{m}_M . If we consider any s -variant of \mathbf{m}_M , we know that what it assigns to s must be a number from the universe of \mathbf{m}_M ; we also know from the way that we constructed the matrix model \mathbf{m}_M that a number was included in the universe of \mathbf{m}_M only if it was assigned to some constant that occurred in the derivation. Let what's assigned to s by some s -variant, \mathbf{m}^{s_2} , be the number associated with the constant c . Because the universal statement $\forall\alpha\phi$ occurred in the derivation, we know that we took all instances of it, including $\phi c/\alpha$. As already stated,

all instances $\phi t/\alpha$ are true in \mathbf{m}_M , including $\phi c/\alpha$. By the DragNet Theorem (Thm. 4.9 on page 95), since $\phi c/\alpha$ is true in \mathbf{m}_M it follows that $\phi s/\alpha$ is true on \mathbf{m}^{s_2} . But the exact same argument will work for every s -variant of \mathbf{m}_M ; so $\phi s/\alpha$ is true on every s -variant of \mathbf{m}_M . It follows that $\forall \alpha \phi$ is true on the matrix model \mathbf{m}_M .

Closure Step: Every sentence in the derivation is true in the matrix model \mathbf{m}_M , which is what was to be shown. ■

7.4.4 Proving Completeness

In this section we put together all the pieces from the last section to prove that QD is complete.

Theorem 7.28. Main Weak QD Completeness Lemma: *For all sentences θ of QL, if the method is applied to $\sim\theta$ then either: (a) the method produces a derivation of θ in QD^+ , or (b) there is some model \mathbf{m} that makes θ false.*

Proof: If the method is applied to $\sim\theta$, then either (1) it will produce a contradiction $A \wedge \sim A$, (2) the method halts without a contradiction, or (3) the method never halts (and hence never halts in a contradiction). If (1), then by the Derivational Lemma (Thm. 7.23, on page 199) there is a derivation of θ .

If either (2) or (3) is the case, then by the Method Lemma 3 (Thm. 7.27, on the previous page) we know that all sentences in the derivation starting with $(\sim\theta)^*$, the prenex disjunctive normal form sentence produced in Step 0 of the method from the sentence $\sim\theta$ on line 1, are true in the matrix model \mathbf{m}_M . Note that all the steps in the derivation of $(\sim\theta)^*$ from $\sim\theta$ are sanctioned by exchange rules; therefore those steps can be turned upside down to produce a derivation in QD^+ of $\sim\theta$ from $(\sim\theta)^*$. So by theorem 6.14 (on page 171) there's a derivation in QD of $\sim\theta$ from $(\sim\theta)^*$. Since QD is sound (Thm. 7.11, on page 185), it follows that $(\sim\theta)^* \models \sim\theta$. Since we know that $(\sim\theta)^*$ is true in the matrix model \mathbf{m}_M , it follows that $\sim\theta$ is true in \mathbf{m}_M too. So it follows that θ is false in \mathbf{m}_M . ■

Theorem 7.29. Weak QD Completeness Theorem: *For all sentences θ of QL, if $\models \theta$, then $\vdash \theta$ in QD.*

Proof: Assume that $\models \theta$. Then there are no models \mathbf{m} which makes θ false. Thus, if the method is applied to $\sim\theta$ it can't be that some model \mathbf{m} makes θ false. By the Main QD Weak Completeness Lemma (Thm. 7.28), it follows that when the method is applied to $\sim\theta$ it produces a derivation of θ in QD^+ . Hence there is a derivation of θ in QD. ■

Theorem 7.30. QD Completeness Theorem: *For all finite sets Δ of QL sentences and QL sentence ϕ , if $\Delta \models \phi$, then $\Delta \vdash \phi$.*

Proof: The theorem follows immediately from the Weak QD Completeness Theorem and theorem 7.14 (on page 188). ■

A consequence of our Strong Method is that if ϕ is entailed by an infinite set of sentences Δ , it is entailed by and derivable from a finite subset of Δ . If the Strong Method does not go on forever, then we get a contradiction at a finite stage and we have only assumed a finite subset of Δ .

7.4.5 Shortcut Rules for the Method

The method discussed in section 7.4.3 becomes practically unwieldy. For example, if the matrix has three disjuncts with two sentences each, then combining two instances gives 9 disjuncts with 4 elements each, and combining three gives 27 disjuncts with 8 elements each. Thus we will use some additional short cut rules to speed the process of detecting contradictions. (But note that *two* of the shortcut Rules we add here are not *exchange* shortcut rules. Greg's rule is the exception.)

Our first shortcut rule is *Greg's Rule*. We know that if a conjunction contains an atomic formula and the negation of that atomic formula then we can derive the negation of the conjunction. E.g., we can derive the negation of $(Ka \wedge Gb \wedge Kc \wedge \sim Gb)$. So if we have a disjunction, one disjunct of which contains a contradiction of this kind, we can derive the negation of that disjunct and use disjunctive syllogism to prune that disjunct. For example, assume the Method ends at the sentence $(Ka \wedge Gb \wedge Kc \wedge \sim Gb) \vee (Ka \wedge Gb \wedge Kc \wedge \sim Ga)$. We can independently derive $\sim(Ka \wedge Gb \wedge Kc \wedge \sim Gb)$. From these two we derive $(Ka \wedge Gb \wedge Kc \wedge \sim Ga)$. Greg's Rule lets us accomplish those steps by crossing out the contradictory part and writing down the remaining ones.

It is helpful to have a short cut rule which combines \wedge -Elim steps with \vee -Elim steps to go from a disjunction of which each disjunct contains a particular sentence to that sentence itself on a later line; we will call it \vee/\wedge -Elim and it sanctions the step from $(Ka \wedge Gb \wedge Kc \wedge \sim Gc) \vee (Ka \wedge Gb \wedge Kc \wedge \sim Ga)$ to Gb . In addition to citing the justification, if the sentence is at all complex you should circle the repeated subsentence.

Finally, given the opposite of even one conjunct in a conjunction, we can derive the negation of the conjunction. E.g., from Ga we can derive $\sim(Ka \wedge Gb \wedge Kc \wedge \sim Ga)$. We will call this rule *One Bad Apple*, or *OBA*.

Name	Given	May Add
Greg's Rule	$\psi_1 \vee \dots \vee \psi_n$, where some $\psi_i = \phi_1 \wedge \dots \wedge \phi_j \wedge \dots$ $\wedge \sim \phi_j \wedge \dots \wedge \phi_m$	$\psi_1 \vee \dots \vee \psi_{i-1} \vee \psi_{i+1} \vee \dots \vee \psi_n$
\vee/\wedge -Elim	$\psi_1 \vee \dots \vee \psi_n$, where	ϕ

Table 7.6: Short-Cut Rules for the Method

Continued next Page

Continued from Previous Page

Name	Given	May Add
	each ψ_i contains ϕ	
OBA	$\phi_1 \wedge \dots \wedge \phi_i \wedge \dots \wedge \phi_n, \sim\phi_i$	$\sim(\phi_1 \wedge \dots \wedge \phi_i \wedge \dots \wedge \phi_n)$
	$\phi_1 \wedge \dots \wedge \sim\phi_i \wedge \dots \wedge \phi_n, \phi_i$	$\sim(\phi_1 \wedge \dots \wedge \sim\phi_i \wedge \dots \wedge \phi_n)$

Table 7.6: Short-Cut Rules for the Method

7.5 Strong Completeness and Other Results

In this section we want to extend our results and show that QD is strongly complete. Note that the method we used to extend the Weak Completeness Theorem to the Completeness Theorem will not work here. To do that, we used theorem 7.14 (on page 188), the proof of which depending on Δ being finite. To show that QD is strongly complete, we have modify the method we used to show that it's weakly complete.

Theorem 7.31. Strong QD Completeness Theorem: *For any set Δ of SL sentences and any SL sentence ϕ , if $\Delta \models \phi$, then $\Delta \vdash \phi$.*

To show that QD is weakly complete, we gave a method that, given a sentence θ , either produces a derivation of θ or produces a model \mathfrak{m} which makes θ false. To show that QD is strongly complete, what we want is a method that, given a (possibly infinite) set Δ of sentences and another sentence ϕ , either produces a derivation of a contradiction $A \wedge \sim A$ from $\sim\phi$ and some finite subset of Δ or produces a model \mathfrak{m} that makes $\sim\phi$ and every sentence in Δ true.

The method we'll give is a modification of the original method given in section 7.4.3. Since it is just a modification of the the original method, we'll only sketch the changes needed. We'll call the modified method the *strong method*. Given some possibly countably infinite set Δ and sentence ϕ , the strong method is:

Step 1: Let $\Delta^* = \Delta \cup \{\sim\phi\}$. Then pair each sentence of Δ^* with a natural number and use that to determine the order in which they are assumed. The only constraint on this ordering is that $\sim\phi$ should be first.

Step 2: Put the first sentence of Δ^* on line 1 and put it in PDNF, just as was done in Step 0 of the method.

Step 3: Apply Step 1 of the method.

Step 4: Apply Steps 2 and 3 of the method to the whole derivation thus far.

Step 5: Check for contradictions, just as in Step 4 of the method.

Step 6:

- (1) If there's a contradiction, stop.
- (2) If there's no contradiction, write the next sentence of Δ^* on the next line of the derivation, put that sentence in PDNF, and go back into Step 4.

The strong method will either halt in a contradiction, or not.

Theorem 7.32. Strong Derivational Lemma: *If the strong method halts in a contradiction, then $\Delta \vdash \phi$.*

Proof: If the strong method halts in a contradiction, then it will have produced a derivation of a contradiction $A \wedge \sim A$ from $\sim\phi$ and some subset Δ' of Δ . We want to show that $\Delta \vdash \phi$; to do this, we first want to show that $\Delta' \vdash \phi$.

We might think we already know that $\sim\phi, \Delta' \vdash A \wedge \sim A$, but in fact, there are additional open assumptions that we made; the Strong Method tells us to make an additional assumption for each line containing an existentially quantified sentence. For these lines, we assume an instance of the existentially quantified sentence. We want to discharge these assumptions by using \exists -Elim, but these assumptions may come prior to some of the assumptions we made from Δ' . We want to keep the sentences of Δ' as assumptions, because the contradiction we reached depends on them.

We can discharge the assumed instances of existentially quantified sentences only by additionally discharging all the assumptions that come later in the derivation. Accordingly, we want to extend our derivation so that we discharge *all* our assumptions and then repeat all the assumptions *except* for the instances of existentially quantified sentences.

Let $\theta_1, \theta_2, \theta_3, \dots, \theta_n$ be the sentences of Δ' assumed in our derivation. The last open assumption is either (a) the last sentence of Δ' , i.e., θ_n ; (b) an instance of an existentially quantified sentence on an earlier line of the derivation which we'll call ψ ; or (c) $\sim\phi$. If (a) is the case, then discharge the assumption by applying the rule \rightarrow -Intro to get $\theta_n \rightarrow (A \wedge \sim A)$. If (b) is the case, then first apply \rightarrow -Intro to get $\psi \rightarrow (A \wedge \sim A)$, and then apply \exists -Elim to get $A \wedge \sim A$. When (b) is the case, we know that the assumption introduced a *new* constant, and therefore we know that that constant did not appear in any earlier assumption or in the existential of which we are taking an instance. It also (obviously) does not occur in $A \wedge \sim A$. Thus the \exists -Elim step is legitimate. If (c) is the case then we don't have to worry about instances of existentially quantified sentences, and we can skip this part of the process. Let us disregard case (c) for now.

Now we must continue to discharge the rest of the assumptions. For any assumption of an instance of an existentially quantified sentence, we may do the same thing we did in (b) above—first use \rightarrow -Intro to derive a conditional, and then use \exists -Elim to derive the RHS of that conditional. For any assumption that is a sentence of Δ' (i.e., θ_i), we will use \rightarrow -Intro to derive a conditional, as in (a) above. So, after discharging the assumption with the second to last sentence from Δ' we get $\theta_{n-1} \rightarrow (\theta_n \rightarrow (A \wedge \sim A))$.

After the third instance, we derive $\theta_{n-2} \rightarrow (\theta_{n-1} \rightarrow (\theta_n \rightarrow (A \wedge \sim A)))$. And so on, so that we eventually get a sentence of the form $\theta_1 \rightarrow (\theta_2 \rightarrow (\theta_3 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A))))$.

After discharging all such assumptions, the only open assumption left is $\sim\phi$. Apply \rightarrow -Intro once more to get something like the following: $\sim\phi \rightarrow (\theta_1 \rightarrow (\theta_2 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A))))$. Now we have no open assumptions remaining. (Note that we have effectively covered case (c) above, since applying \rightarrow -Intro in this case would give us $\sim\phi \rightarrow (A \wedge \sim A)$ and no open assumptions. In this case, we didn't have to assume any of the sentences of Δ to get a contradiction, so Δ' is the empty set.)

At this point we have a conditional, possibly a very long one. The RHS of the last conditional (possibly embedded in several conditionals) is our contradiction, $(A \wedge \sim A)$. We want to show that $\Delta' \vdash \phi$, so let us make a series of assumptions from the sentences of Δ' . That is, let us assume each of $\theta_1, \theta_2, \theta_3, \dots, \theta_n$. Now that we've assumed all the sentences of Δ' , let us assume $\sim\phi$.

Given our earlier conditional of the form $\sim\phi \rightarrow (\theta_1 \rightarrow (\theta_2 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A))))$ and the assumption $\sim\phi$, we may now apply \rightarrow -Elim to get $\theta_1 \rightarrow (\theta_2 \rightarrow (\theta_3 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A))))$. Because we also have all the sentences of Δ' as assumptions (i.e., all of $\theta_1, \theta_2, \theta_3, \dots, \theta_n$), we may apply a series of \rightarrow -Elim steps until we eventually derive $A \wedge \sim A$. That is, given our earlier assumption θ_1 and the conditional $\theta_1 \rightarrow (\theta_2 \rightarrow (\theta_3 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A))))$, we may apply \rightarrow -Elim to derive $\theta_2 \rightarrow (\theta_3 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A)))$. And then because we have θ_2 as an assumption we may again apply \rightarrow -Elim to get $\theta_3 \rightarrow \dots (\theta_n \rightarrow (A \wedge \sim A))$. And so on, until we derive $A \wedge \sim A$.

Remember that our last open assumption is $\sim\phi$. We may now discharge that assumption and apply \rightarrow -Intro to get $\sim\phi \rightarrow (A \wedge \sim A)$. Then we apply \sim -Elim to derive ϕ .

Now we have as our open assumptions only the sentences of Δ' and we have derived ϕ . We have thus shown that $\Delta' \vdash \phi$. And because Δ' is a subset of Δ , it follows that $\Delta \vdash \phi$. ■

Next, note that if the strong method doesn't halt in a contradiction, then we will have a list of matrix instances from which we can construct a matrix model \mathbf{m}_M in just the same way we did for the method (Def. 7.24, on page 201). Similar to the method, we have the following three theorems.

Theorem 7.33. The Strong Method Lemma 1: *The matrix model \mathbf{m}_M makes true all sentences on the master matrix list M .*

Proof: The same proof used for the method (Thm. 7.25, on page 201) applies here too. ■

Theorem 7.34. The Strong Method Lemma 2: *All matrix instances in the derivation are true in the matrix model \mathbf{m}_M .*

Proof: The same proof used for the method (Thm. 7.26, on page 201) applies here too. ■

Theorem 7.35. The Strong Method Lemma 3: *All quantified sentences in the derivation are true in the matrix model \mathbf{m}_M .*

Proof: The same proof used for the method (Thm. 7.27, on page 202) applies here too, so long as we can show that if an existential $\exists\alpha\psi$ appears on a line, at least one instance $\psi t/\alpha$ does, and if a universal $\forall\alpha\psi$ appears on a line, then every instance $\psi t/\alpha$ of it with a constant t appearing somewhere in the derivation appears somewhere in the derivation. So, all we need to show is that the strong method will derive the appropriate instances of all quantified sentences that appear in our derivation. Let Δ' be those sentences of Δ^* that appear in our derivation as a result of the application of the strong method.

Existential Quantifier: Say θ is some sentence in the derivation of the form $\exists\alpha\phi$. Step 4 of the strong method uses step 3 of the method on θ , which guarantees that the sentence $\phi t/\alpha$, for some constant t , appears somewhere in the derivation. By hypothesis, step 4 of the strong method is applied to all sentences in Δ' , so we know that it will derive the appropriate instances of all existentially quantified statements in Δ' .

Universal Quantifier: Say θ is a sentence appearing in the derivation of the form $\forall\alpha\phi$. Step 4 of the strong method uses step 2 of the method on θ , which, for every constant t in the derivation, guarantees that the sentence $\phi t/\alpha$ is derived. By hypothesis, step 4 of the strong method is applied to all sentences in Δ' , so we know that it will derive the appropriate instances of all universally quantified statements in Δ' .

So, the strong method derives the appropriate instances of all quantified sentences in Δ' . ■

Finally, we have one last lemma:

Theorem 7.36. Main Strong QD Completeness Lemma: *For all sets of QL sentences Δ and QL sentences ϕ , if the strong method is applied to $\Delta^* = \Delta \cup \{\sim\phi\}$ then either: (a) the strong method produces a derivation of ϕ from Δ in QD^+ , or (b) there is a model \mathbf{m} which makes every sentence in Δ true and ϕ false.*

Proof: If the method is applied to $\Delta^* = \Delta \cup \{\sim\phi\}$, then either it halts in a contradiction or not. By the Strong Derivational Lemma (7.32, on page 206), if the strong method halts in a contradiction, then $\Delta \vdash \phi$ in QD^+ .

If the method does not halt in a contradiction, then by the Strong Method Lemma 3 (Thm. 7.35, on the current page) the matrix model \mathbf{m}_M makes all the sentences in the derivation true. But since the strong method did not halt in a contradiction, for every sentence ψ in Δ and $\sim\phi$, there's some sentence in PDNF that's quantificationally

equivalent to ψ and appears in the derivation. So \mathbf{m}_M makes all the sentences in Δ and the sentence $\sim\phi$ true; hence \mathbf{m}_M makes all the sentences in Δ true and ϕ false. ■

Proof of Thm. 7.31, The Strong Completeness Theorem for QD: Assume that $\Delta \models \phi$. Then there can be no model \mathbf{m} which makes all of the sentences in Δ true and ϕ false; so, application of the method can't produce such a model. Thus by the Main QD Strong Completeness Lemma (Thm. 7.36), $\Delta \vdash \phi$ in QD⁺. It follows by theorem 6.14 (on page 171) that $\Delta \vdash \phi$ in QD. ■

Our Method for proving completeness for QD is short of being a decision procedure. If ϕ is a logical truth, then The Method will produce a derivation of it. And if ϕ is not a logical truth, then in many cases it produces a model that makes ϕ false. But sometimes it just doesn't stop. We know that if it doesn't stop there is a model that makes the original sentence false, but at each stage we may not know whether it will stop (soon?) or not. And we know that it can't stop at a finite stage for some sentences because those sentences are only false in an infinite model.

All we know from our work so far is that The Method works as described above. We don't know that there isn't a better method that provides a decision procedure. Church's Theorem, proved by more advanced methods that involve clarifying what counts as a "method" or "algorithm" tells us that our result is as good as we can do for all of QL.

However, we can do better for the language QL1 and a little more.

7.6 Decidability and Church's Theorem

Next we turn to refinements of the method to obtain what are called *decision procedures* for logical truth in a language L. We first introduced the idea of a decision procedure in section 6.2.5 (on page 137); here we shall fill things out a bit further.

Definition 7.37. A *decision procedure* for logical truth in a language (or sublanguage) L is a completely specified method which produces, for any sentence ϕ of L and in a finite number of steps, the answer YES if ϕ is a logical truth and the answer NO otherwise.

Example 7.38. We have already seen two decision procedures for TFT in SL: truth tables and the method discussed in the completeness proof of SD. (Others include truth trees and Quine's "fell swoop", Quine 1950, Hodges 2001b: 23.) The truth-table decision procedure is simple: take a sentence ϕ of SL and construct a truth table for it. If you get all T in the column under ϕ ; answer YES. If you don't, answer NO. Likewise for the method discussed in the completeness proof of SD: take a sentence ϕ , negate it to get $\sim\phi$, and apply the method. If it results in a contradiction $A \wedge \sim A$, answer YES. If no contradiction is reached, answer NO.

Our method of proving completeness for QD is a little short of being a decision procedure for quantificational truth in QL because it does not always produce an answer

in a finite amount of time. It can be shown that there is no decision procedure for the whole language QL (Hodges 2001b: 83–86, 2001a: 31, Bergmann et al. 2003: 486).

Theorem 7.39. Church's Theorem: *If L is a sublanguage of QL with (1) the same logical connectives as QL, and (2) at least one 2-place predicate symbol, then there is no decision procedure for the set of logical truths of L .*⁴

Church's Theorem at once tells us that there is no decision procedure for quantificational truth in QL, but we can show that with some modifications our method from section 7.4.3 can be turned into a decision procedure for certain sublanguages of QL. As the thesis suggests, one such sublanguage L of QL is the language that consists of just 1-place predicate symbols. We've been calling this language QL1.

The basic insight for modifying the method is that infinite loops are created by having an existential quantifier inside a universal quantifier; the existential requires a new constant to be instantiated, and that creates a new potential instance for the universal, which then creates a new existential, and so on.

Put more positively, if the method produces a sentence in standardized form which has only existential, or only universal quantifiers, then the method will stop. Moreover, if in the standardized form the existentials all precede the universals the method will also stop, for we will first take instances of all the existentials using n constants if there are n existentials, and afterwards we will instantiate the n new constants in the m universals giving n^m instances. But, we will be done then since no more new constants will be added.

The general principle is that if one quantifier occurs within the scope of another then it cannot be moved in front of the other quantifier. You may remember that the scope of a quantifier, say \forall in $\forall x\phi$, is the subformula ϕ of which it is the main connective. We can say two quantifiers are *independent* if neither is in the scope of the other. Let's say we have a sentence all of whose universal quantifiers are independent of each other. Notice that these quantifiers can be brought forward in any order, and thus that we have a decision procedure for such sentences.

The critical result to prove is that every sentence of QL1 is equivalent to a sentence whose quantifiers are independent. We can show this by appeal to the reverse of our procedure for putting sentences into prenex form, i.e. by moving quantifiers inward, but we use most of the same rules as for standard prenex (remember they are exchange rules).

Theorem 7.40. QL1 Equivalence Theorem: *Every sentence of QL1 is quantificationally equivalent to a sentence whose quantifiers are independent.*

Proof: Proof here. ■

⁴Actually, Church's Theorem also says that if we also consider languages with function symbols, then if L has at least two 1-place function symbols there is no decision procedure for the set of logical truths of L .

Theorem 7.41. The QL1 Decision Theorem: *The modified method just described provides a decision procedure for quantificational truth in QL1.*

7.7 Löwenheim-Skolem and Compactness

A number of results follow directly from the completeness of QD or the method we used to prove completeness.

Theorem 7.42. The Downward Löwenheim-Skolem Theorem: *If a sentence of QL is true in any model, then it is true in one whose domain consists of all or some of the natural numbers.*

Proof: If ϕ is true in some model, then $\sim\phi$ is not a quantificational truth. Thus, applying the method to $\sim\phi$ will not produce a contradiction, but will produce a model of the natural numbers which falsifies $\sim\phi$ and hence makes ϕ true. ■

It's important to note that there's really nothing special about the natural numbers. When we devised the procedure for constructing a model of the ions in the master matrix list that results from the method (when no contradiction arises), we choose to use natural numbers for the universe. But it should be clear that we did this out of convenience (it's easy, after all, to associate constants with the natural numbers). We could have used any set of objects for the universe. What's important is that, whatever we used, the domain of the constructed model will at most be countably infinite (that is, it will at most be the size of the natural numbers and no larger). Hence a more abstract version of the downward Löwenheim-Skolem Theorem simply says: If a sentence of QL is true in any model, then either it's true in only models with finite domains, or, if it's true at all in models with infinite domains, then there's an model with a *countably* infinite domain in which it's true.

This theorem was proved in a weaker form originally by Leopold Löwenheim (1915), and the proof was improved by Thoralf Skolem (1920; 1922). Notice that the theorem talks only about models, and we have proved it via a detour through derivations. As you might imagine, there are more direct proofs, including Skolem's (Tarski and Vaught 1956, Vaught 1974, Hodges 1997: ch. 3.1, 2001b: 63).

Notice also that after our work on QL1, we know that for monadic sentences the method will stop after a finite number of steps (if we arrange the prenex carefully) and so we can conclude that if a monadic sentence is true in any model then it is true in a finite one.

Theorem 7.43. *If ϕ is a sentence of QL1 and has a model, then it has a finite model.*

Our next corollary of completeness is the Compactness Theorem. Although historically the completeness theorem was proved first and compactness followed as a corollary, today the compactness theorem takes center stage in many areas of logic (especially model theory). Like the Löwenheim-Skolem Theorem, there are many different proofs of compactness that do not go through completeness or use any facts

about derivations (see Kleene 2002 [1967]: 321, Ebbinghaus 1985, Hodges 1997: ch. 5.1, Hodges 2001b: 63, 2001a: 29).

Theorem 7.44. The Compactness Theorem for QL: *For all sets of sentences Δ of QL, if for every finite subset Δ' of Δ there exists a model \mathbf{m}' that makes all the sentences in Δ' true, then there's some model \mathbf{m} that makes all the sentences in Δ true.*

Proof: By the strong completeness theorem, for all sets Δ of QL sentences and QL sentence ϕ , if $\Delta \models \phi$, then $\Delta \vdash \phi$. Now assume that there's no model \mathbf{m} that makes all the sentences in Δ true. Hence $\Delta \models A \wedge \sim A$. So by strong completeness, $\Delta \vdash A \wedge \sim A$. By definition, this implies that there's some finite subset Δ' of Δ such that $A \wedge \sim A$ can be derived from Δ' . Hence there is no model \mathbf{m} that makes all the sentences in Δ' true. Hence it's not the case that for every finite subset Δ' of Δ there exists a model \mathbf{m}' that makes all the sentences in Δ' true. ■

7.8 Exercises

7.8.1 Misc. Problems

- Let's say that any derivation rule R that has the following property is *sound*: if we add a line to a derivation D with sentence ϕ sanctioned by rule R , then $\Delta \models \phi$, where Δ is the set of unboxed assumptions for the new line. (Compare this with what it is for a rule to be truth-preserving, def. 7.1 on page 181, which is different.) Then the proof of theorem 7.7 (on page 182) basically shows that SD is sound by showing that all the basic rules of SD are sound. We know that SD^+ is sound because SD is sound and (by theorem 6.5, on page 152) anything you can derive in SD^+ can be derived in SD. But we could also show that SD^+ is sound directly (without appealing to theorem 6.5) by showing that the shortcut rules used in SD^+ themselves are sound. Of course, this follows from theorem 6.4, on page 152 and the fact that the basic rules are sound, but again we can show it directly. But again we can show it without going through the basic rules. Show directly (without appealing to theorem 6.4) that the following rules are sound (see tables 6.36, on page 149 and 6.37, on page 150):

- | | |
|-----------------|---------------------------------------|
| (a) <i>M.T.</i> | (c) \rightarrow/\vee - <i>Exch.</i> |
| (b) <i>A.C.</i> | (d) <i>Contraposition</i> |

- Recall that \rightarrow elimination can only be used on a conditional that is the main connective of a sentence. Show that if we do not make this restriction, then the rule is unsound. In other words, give a derivation which violates only that restriction (a derivation where you use \rightarrow elimination on a horseshoe that's not the main connective) and which ends with a proof of a sentence that is *not* a logical truth (not truth-functionally true) from the empty set of assumptions.

Chapter 8

Further Directions

8.1 Many-Valued Logic

In previous chapters we assumed that there are only two truth values. We now set aside that assumption. This allows us to explore many-valued logics. We explore the possibility that there are varying degrees of truth and falsity, such that there are sentences that are neither wholly true nor wholly false. (Actually it turns out that there are many possibilities because there are many many-valued logics; in fact, there are infinitely many!) Among the reasons that have been given historically for rejecting the two-valuedness assumption are the beliefs that statements about the future, statements involving vague predicates, or statements about quantum-mechanical properties are not always either true or false. Most many-valued logics begin by rejecting the law of excluded middle $A \vee \sim A$, though there are exceptions.¹ The number of values ranges from three to various infinite sets. The interpretations of the further values vary widely from author to author, as do the motivations for introducing the additional values.

Emil Post was one of the first to study many-valued logics, but his motivation seems to have been entirely formal. The other major founder of many-valued logic was Łukasiewicz. He sketched the idea of a many-valued logic in 1920 and published a systematic account in 1930. (Both are reprinted in [Borkowski 1970](#).) Unlike Post, Łukasiewicz introduced three-valued logic for a philosophical reason: to provide a more appropriate representation for the indeterminacy of the future. He apparently was led to this both by a historical concern—studying Aristotle’s discussion of necessity, particularly his sea battle example—and by a quite contemporary concern about how to accommodate the indeterminism of modern physics within logic. Aristotle’s sea battle argument is as follows:

- (1) If there will be a sea battle tomorrow, then necessarily there will be a sea battle tomorrow.
- (2) If there will not be a sea battle tomorrow, then necessarily there will not be a sea battle tomorrow.

¹For example, the most generally accepted logic for quantum mechanics keeps $A \vee \sim A$ but rejects one of the distribution principles.

- (3) Either there will or there will not be a sea battle tomorrow.
- (4) Therefore, either there will necessarily be a sea battle tomorrow or there will necessarily not be a sea battle tomorrow.

Aristotle suggested that the third premise—the law of excluded middle, $A \vee \sim A$ —should be rejected when A is a statement about a future contingency. Thus the motivation, if not the details, of many-valued logic is as ancient as the study of logic itself. Łukasiewicz developed this idea into a systematic logic.

In all of his later work, Łukasiewicz used 1 for truth, 0 for falsity, and intermediate values for other truth values. Most but not all writers use this convention. Of course it is one thing to decide that $1/2$ is your third truth value and another thing to give a philosophical explanation of it. For Łukasiewicz the intermediate value is “indeterminate”. Given this understanding, the most natural three-valued generalization of the two-valued truth tables is the following, in which negation reverses the truth value.

A	$\sim A$
1	0
$1/2$	$1/2$
0	1

Conjunction takes the minimum value of the conjuncts, while disjunction takes the maximum value of the disjuncts.

$A \wedge B$	1	$1/2$	0
1	1	$1/2$	0
$1/2$	$1/2$	$1/2$	0
0	0	0	0

$A \vee B$	1	$1/2$	0
1	1	1	1
$1/2$	1	$1/2$	$1/2$
0	1	$1/2$	0

For example, the conjunction of a true sentence and an indeterminate sentence would seem to be indeterminate. It could become true if the indeterminacy were resolved in favor of truth or false if the indeterminacy were resolved in favor of falsity.

Note that when all the components of a sentence formed from these connectives are assigned value $1/2$ the entire sentence has value $1/2$. If we introduce the conditional as

$\sim A \vee B$, as is often done in two-valued logic, then the conditionals would also have this property and there would be no sentences which are logical truths. More specifically, since the identification of the conditional with $\sim A \vee B$ makes $A \rightarrow A$ equivalent to the law of excluded middle, $A \rightarrow A$ would not be a logical truth.

Instead of using that traditional, oft questioned equivalence, Łukasiewicz defined the conditional thus:

$A \rightarrow B$	1	$\frac{1}{2}$	0
1	1	$\frac{1}{2}$	0
$\frac{1}{2}$	1	1	$\frac{1}{2}$
0	1	1	1

One way of describing this table is that the conditional is false only in the case of $T \rightarrow F$ and is indeterminate only in two cases: $T \rightarrow I$ and $I \rightarrow F$. A rationale for these choices is that if A were true and B indeterminate, then the conditional $A \rightarrow B$ could be true if B were to be true and false if B were to be false. The choice of the value 1 when both constituents have value $\frac{1}{2}$ is required if $A \rightarrow A$ is to be logically true.

Equivalence can be defined as usual: $A \leftrightarrow B$ iff $A \rightarrow B$ and $B \rightarrow A$. The truth table for the biconditional is specified by this definition above. We leave it to the reader to give a simpler, more direct explanation of the truth table. In Łukasiewicz' presentation of his system, he used only negation and the conditional, having noted that $A \vee B$ can be defined as $(A \rightarrow B) \rightarrow B$ and then $A \wedge B$ can be defined by using the usual DeMorgan's principle.

In two-valued logic we define a sentence to be logically true iff it is true in all models. When we have more than two truth values, we must indicate which subset of the values are the designated values, i.e., those that are truth-like. Our definition now becomes: A is a *logical truth* iff it has a designated value in all models.

Since Łukasiewicz' motivation was to deny excluded middle, he chose only 1 as a designated value. This achieves the purpose of rendering excluded middle not a logical truth. It has one somewhat counterintuitive consequence though, which is that under a model in which both constituents are assigned value $\frac{1}{2}$, $A \wedge \sim A$ has the same truth value as $A \vee \sim A$. Issues of the indeterminacy of the future are now generally studied within the framework of tense logic. Aristotle's argument is generally regarded as fallacious, but Łukasiewicz' innovations have opened the possibilities for a variety of other systems and ideas.

8.1.1 Finite-valued systems with more than three values

The Łukasiewicz three-valued generalization can be systematically carried further. The n -valued generalization consists of taking the values $\frac{i}{n-1}$ for $0 \leq i \leq n-1$. For example, four-valued logic would have the values 0, $\frac{1}{3}$, $\frac{2}{3}$, and 1. Conjunction will take the minimum value of the conjuncts, and disjunction the maximum value of the

disjuncts. The value of a negation is 1 minus the value of the sentence being negated. For the conditional $A \rightarrow B$ we have two cases, using $V(A)$ for the value of A :

$$V(A \rightarrow B) = \begin{cases} 1 & \text{if } V(A) \leq V(B) \\ [1 - V(A)] + V(B) & \text{otherwise} \end{cases}$$

In all of the Łukasiewicz systems the only designated value is 1. Excluded middle will not be logically true in any of these systems, though in the even-valued systems excluded middle is always truer than the contradiction $A \wedge \sim A$. Systems with more than one designated value were mentioned by Post, and this variation on Łukasiewicz systems was studied by Slupecki and others.

Four-valued logic was proposed for modal logic, the values being “necessarily true”, “contingently true”, “contingently false”, and “necessarily false”. The Łukasiewicz definitions of the usual connectives can be used and a modal operator added. While these truth tables have some uses, they have been superseded by the possible worlds approach to modal logic. We discuss possible world semantics later in this chapter.

8.1.2 Infinite-valued systems

The Łukasiewicz n -valued generalization can be systematically carried further still. Łukasiewicz also studied the cases where the set of truth values consists of all rational numbers (fractions) in the interval $[0, 1]$ and also where the values consist of all real numbers (infinite decimal expansions) in the same interval. Conjunction, disjunction, negation, and the conditional (with the two cases) are the same as for finite-valued systems with more than three values. The set of logical truths in the three-valued logic is a subset of those in our traditional logic; if a sentence can be shown to be false in a two-valued model, then that model also works in the three-valued system.

One question to ask is whether all these values make a difference. We already know part of the answer. $A \vee \sim A$ is a two-valued logical truth but not a three-valued one. But this does not give us an answer for the other systems. Before addressing this question, let's generalize our observation about excluded middle. If we think about the truth tables for \sim , \wedge , and \vee , we can see that when the input value(s) are $1/2$, the output value is always $1/2$. Thus a simple recursive proof (that we won't bother giving) shows that, in a model in which all atoms receive value $1/2$, any sentence without conditionals and biconditionals is not a logical truth, because it receives value $1/2$ on that model.

Theorem 8.1. *All logical truths of Łukasiewicz systems contain conditionals or biconditionals.*

This means that if we are looking for sentences that discriminate between the n -valued systems then we need to look at conditionals or biconditionals. One candidate for a form of sentence is $(A \rightarrow B) \vee (B \rightarrow C)$. This is a two-valued logical truth but not a three-valued one, because we can assign the values 1, $1/2$, and 0 to the respective atoms and give the sentence a value of $1/2$. Generalizing, sentences of the form

$$(A_1 \rightarrow A_2) \vee (A_2 \rightarrow A_3) \vee (A_3 \rightarrow A_4) \vee \dots \vee (A_n \rightarrow A_{n+1})$$

will be logical truths in an n -valued system but not in a system with at least $n + 1$ values.

Example 8.2. Is the sentence $(A \wedge \sim B) \rightarrow \sim(A \rightarrow B)$ a three-valued truth? If so, are they n -valued truths for all finite n ?

Solution. This is not a logical truth in any system with more than two values. If we assign both A and B the same intermediate value, then $\sim B$ will have an intermediate value and so $A \wedge \sim B$ will have an intermediate value. But since A and B have the same value, $A \rightarrow B$ will have the value 1 and $\sim(A \rightarrow B)$ will have the value 0. So, the whole conditional will have an intermediate value, not 1.

Example 8.3. Is the sentence $((A \rightarrow C) \wedge (B \rightarrow C)) \rightarrow ((A \vee B) \rightarrow C)$ a three-valued truth? If so, are they n -valued truths for all finite n ?

Solution. We will give an argument that this sentence is true for all finite-valued logics. If the values of A and B are both less than or equal to that of C , then the right conditional has value 1 and consequently the whole sentence does too. Thus it can only be less than 1 if A or B has value greater than C . Suppose A is greater than C and greater than or equal to B . Then $\text{Max}(A, B) = A$ and the right conditional has the same value as $A \rightarrow C$. But since $A > B$, $1 - A < 1 - B$, and so the value of the left conditional will also be the value of $A \rightarrow C$ because that is the minimum value on that side.

8.2 Modal Sentential Logic

We can think of sentential logic as the logic of truth functional operators and quantificational logic (at least, that part which extends sentential logic) as the logic of quantifiers and predicates. But there is a lot that's left out by this: possibility and necessity, deontic concepts like permissibility and obligation, doxastic concepts like belief and knowledge, and temporal concepts like past and future, just to name a few. Logics which attempt to capture these concepts are called modal logics. Sometimes the term 'modal logic' is just meant to pick out the logic of possibility and necessity, other times it's meant to pick out the broader class of modal logics just mentioned. In this section we give a very brief introduction to the logic of possibility and necessity.

8.2.1 The Language MSL

The language of modal sentential logic, MSL, is an extension of SL. We add to SL two 1-place logical connectives: \Box and \Diamond . Intuitively, $\Box\phi$ says that ϕ is necessarily true, while $\Diamond\phi$ says that ϕ is possibly true.

Definition 8.4. The *sentences of MSL* are given by the following recursive definition:

Base Clause: Every sentence letter (def. 2.1) is a sentence.

Generating Clauses:

- (1) If ϕ is a sentence, then so are $\sim\phi$, $\Box\phi$ and $\Diamond\phi$.
- (2) If ϕ and θ are sentences, then so are $(\phi \rightarrow \theta)$ and $(\phi \leftrightarrow \theta)$.
- (3) If all of $\phi_1, \phi_2, \phi_3, \phi_4, \dots, \phi_n$ are sentences (the list must include at least two sentences and be finite), then so are $(\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4 \wedge \dots \wedge \phi_n)$ and $(\phi_1 \vee \phi_2 \vee \phi_3 \vee \phi_4 \vee \dots \vee \phi_n)$.

Closure Clause: A sequence of symbols is a MSL sentence if and only if its being a sentence follows from the previous two clauses.

Of course, as with SL, sentences of MSL only count as either true or false relative to a model.

8.2.2 Truth, Logical Truth, and Entailment in MSL

Like the models for QL (and unlike the many-valued models for SL), models for MSL can be thought of as extensions of the (2-valued) models of SL (recall definition 2.18, on page 22).

Definition 8.5. A model \mathbf{m} for MSL is an ordered triple $\langle w_0, W, V \rangle$ where:

- (1) W is a set, the elements of which are thought of as “possible worlds”
- (2) $w_0 \in W$, where w_0 plays the role of “the actual world”
- (3) V is a function that assigns a subset of W to each sentence letter of MSL.

Elements of W , “possible worlds”, will be denoted by w, w_0, w_1, w_2 and so on. The subset $V(\phi)$ of W assigned to a sentence letter ϕ can be the empty set \emptyset , a non-empty proper subset of W , or all of W itself. Intuitively, the set $V(\phi)$ is the set of possible worlds in which ϕ is true. And $W - V(\phi)$, those worlds not in $V(\phi)$, are worlds in which ϕ is false. (We use the ‘ $-$ ’ symbol to indicate the ‘complement’ of a set.)² Thus the function V in a MSL model is like a SL model, except that it assigns truth values to sentence letters in a possible world. Just as before, we can extend the notion of truth in a possible world w on a model \mathbf{m} .

Definition 8.6. The following clauses define a function V^* which extends V to all sentences of MSL.

- (1) If ϕ is a sentence letter, $V^*(\phi) = V(\phi)$.
- (2) If ϕ is $\sim\psi$, then $V^*(\phi) = W - V(\psi)$, i.e. $V^*(\phi)$ is the set of worlds in W not in $V(\psi)$.
- (3) If ϕ is $(\psi_1 \wedge \dots \wedge \psi_n)$, then $V^*(\phi) = V^*(\psi_1) \cap \dots \cap V^*(\psi_n)$.

²For example, relative to some domain, $\{1, 2, 3, 4\}$, the complement of the set $\{1, 2\}$ is $\{3, 4\}$. We can indicate the latter with the following notation: $\{1, 2, 3, 4\} - \{1, 2\}$.

- (4) If ϕ is $(\psi_1 \vee \dots \vee \psi_n)$, then $V^*(\phi) = V^*(\psi_1) \cup \dots \cup V^*(\psi_n)$.
- (5) If ϕ is $(\psi \rightarrow \theta)$, then $V^*(\phi) = V^*(\sim\psi) \cup V^*(\theta)$.
- (6) If ϕ is $(\psi \leftrightarrow \theta)$, then $V^*(\phi) = (V^*(\psi) \cap V^*(\theta)) \cup (V^*(\sim\psi) \cap V^*(\sim\theta))$.
- (7) If ϕ is $\Box\psi$, then $V^*(\phi) = W$ iff $V^*(\psi) = W$, otherwise $V^*(\phi) = \emptyset$.
- (8) If ϕ is $\Diamond\psi$, then $V^*(\phi) = W$ iff $V^*(\psi) \neq \emptyset$, otherwise $V^*(\phi) = \emptyset$.

For a given model \mathbf{m} , we call this function V^* the *valuation function* of \mathbf{m} . From here there is a straightforward way to define truth in a world in a model:

Definition 8.7. A sentence ϕ of MSL is *true in world w* in model $\mathbf{m} = \langle w_0, W, V \rangle$ (where $w \in W$) iff $w \in V^*(\phi)$.

Although there isn't as much intuitive significance to it, we can define truth in a model:

Definition 8.8. A sentence ϕ of MSL is *true in model $\mathbf{m} = \langle w_0, W, V \rangle$* iff $V^*(\phi) = W$.

Perhaps it's best not to use "truth" here, but instead something like "truth everywhere"; so we might say that ϕ is true everywhere in \mathbf{m} iff $V^*(\phi) = W$, for V^* the valuation function of \mathbf{m} . The significant concepts are (1) truth in a world in a model (def. 8.7), and (2) truth at all worlds in a model (def. 8.8).

The concepts of logical truth, logical falsity, and logical indeterminacy can be adapted to MSL as well:

Definition 8.9. ϕ is a *modal truth* iff for all models $\mathbf{m} = \langle w_0, W, V \rangle$, $V^*(\phi) = W$.

Definition 8.10. ϕ is a *modal falsehood* iff for all models $\mathbf{m} = \langle w_0, W, V \rangle$, $V^*(\phi) = \emptyset$.

Definition 8.11. ϕ is *modally contingent* iff there's a model $\mathbf{m} = \langle w_0, W, V \rangle$ such that $V^*(\phi) \neq \emptyset$ and there's a model $\mathbf{m} = \langle w_0, W, V \rangle$ such that $V^*(\phi) \neq W$. (Note: It can be the same model for both.)

Alternatively (but equivalently) put, a sentence ϕ is a modal truth iff it's true everywhere on all models \mathbf{m} ; i.e. iff for every world w in every model \mathbf{m} , ϕ is true in w in \mathbf{m} . Similarly, we can alternatively say that a sentence ϕ is a modal falsity iff for every world w in every model \mathbf{m} , ϕ is false in w in \mathbf{m} .

Finally, we also can define when a set Δ of MSL sentences entails a sentence ϕ .

Definition 8.12. A sentence ϕ is *entailed* by a set of sentence Δ iff, for every model $\mathbf{m} = \langle w_0, W, V \rangle$ and world $w \in W$, whenever every sentence in Δ is true in w in \mathbf{m} , then ϕ is also true in w in \mathbf{m} ; i.e. iff, for every model $\mathbf{m} = \langle w_0, W, V \rangle$ and world $w \in W$, $\bigcap_{\psi \in \Delta} V^*(\psi) \subseteq V^*(\phi)$.

Example 8.13. $\Diamond A \rightarrow \Box \Diamond A$ is a modal truth.

Proof: Note that $w \in V^*(\sim\psi) \cup V^*(\theta)$, for V^* the valuation function of some model \mathbf{m} and w a world in the set W of \mathbf{m} , iff if $w \in V^*(\psi)$, then $w \in V^*(\theta)$. So, suppose

$\Diamond A$ is true at some w in some model \mathbf{m} , i.e. suppose that $w \in V^*(\Diamond A)$ for V^* the valuation function of \mathbf{m} . By (8) of definition 8.6, either $V^*(\Diamond A) = W$ or $V^*(\Diamond A) = \emptyset$. Since $w \in V^*(\Diamond A)$, $V^*(\Diamond A) \neq \emptyset$. So, $V^*(\Diamond A) = W$. Now Consider $\Box \Diamond A$. Since $V^*(\Diamond A) = W$, by (7) of definition 8.6 it follows that $V^*(\Box \Diamond A) = W$. So, $\Box \Diamond A$ is true at every element of W , including w . Therefore, $w \in V^*(\Box \Diamond A)$. Since w was arbitrary, this means that for \mathbf{m} , $V^*(\sim\psi) \cup V^*(\theta) = W$. Since \mathbf{m} was arbitrary, this means that $V^*(\sim\psi) \cup V^*(\theta) = W$ for all \mathbf{m} . ■

Example 8.14. There are sentences ϕ and ψ for which $\Box(\phi \vee \psi) \rightarrow (\Box\phi \vee \Box\psi)$ is not a modal truth.

Proof: Consider a model $\mathbf{m} = \langle w_0, W, V \rangle$ with the set $W = \{w_0, w_1\}$. Let $\phi = B$ and let $\psi = \sim B$. Let $V(B) = \{w_0\}$. By (2) in definition 8.6, $V^*(\sim B) = \{w_1\}$. By (4) in definition 8.6, $V^*(B \vee \sim B) = \{w_0, w_1\}$; hence by (7) of definition 8.6 $V^*(\Box(B \vee \sim B)) = \{w_0, w_1\}$. Thus, $\Box(B \vee \sim B)$ is true at each w in W , including world w_0 . Consider then $(\Box B \vee \Box \sim B)$. By (7) of definition 8.6, $V^*(\Box B) = \emptyset$ since $V^*(B) = \{w_0\} \neq W$. Again, $V^*(\Box \sim B) = \emptyset$ since $V^*(\sim B) = \{w_1\} \neq W$. So, by (4) in definition 8.6, $V^*(\Box B \vee \Box \sim B) = \emptyset$. Thus, $(\Box B \vee \Box \sim B)$ is false at each w in W , including world w_0 . Therefore, $\Box(B \vee \sim B) \rightarrow (\Box B \vee \Box \sim B)$ is not true in w_1 in \mathbf{m} ; so it is not a modal truth. ■

You should see a strong similarity here with the fact that $\forall x(Bx \vee \sim Bx) \rightarrow (\forall x Bx \vee \forall x \sim Bx)$ is not a logical truth.

Example 8.15. All of the following are modal truths, for any substitution of sentences ϕ and ψ . We leave the proofs as exercises to the reader.

- | | |
|---|--|
| (1) $\Box\Box\phi \rightarrow \Box\phi$ | (9) $(\Diamond\phi \vee \Diamond\psi) \leftrightarrow \Diamond(\phi \vee \psi)$ |
| (2) $\Box\Diamond\phi \rightarrow \Diamond\phi$ | (10) $\Box(\phi \wedge \psi) \leftrightarrow (\Box\phi \wedge \Box\psi)$ |
| (3) $\Diamond\phi \rightarrow \Diamond\Diamond\phi$ | (11) $\Diamond(\phi \wedge \psi) \rightarrow (\Diamond\phi \wedge \Diamond\psi)$ |
| (4) $\Box\phi \rightarrow \Diamond\Box\phi$ | (12) $\sim\Diamond\sim\phi \leftrightarrow \Box\phi$ |
| (5) $\sim\Diamond\phi \leftrightarrow \Box\sim\phi$ | (13) $\Box(\phi \rightarrow \psi) \rightarrow (\Diamond\phi \rightarrow \Diamond\psi)$ |
| (6) $\Diamond\sim\phi \leftrightarrow \sim\Box\phi$ | (14) $\Diamond(\phi \rightarrow \psi) \rightarrow (\Box\phi \rightarrow \Diamond\psi)$ |
| (7) $\Box(\phi \rightarrow \psi) \rightarrow (\Box\phi \rightarrow \Box\psi)$ | (15) $\sim\Diamond\phi \rightarrow \Box(\phi \rightarrow \psi)$ |
| (8) $(\Box\phi \vee \Box\psi) \rightarrow \Box(\phi \vee \psi)$ | (16) $\Box(\phi \leftrightarrow \psi) \rightarrow (\Box\phi \leftrightarrow \Box\psi)$ |

One of the most distinctive features of the models we're using here (def. 8.5) and the definitions of truth (def. 8.6) and modal truth (def. 8.9) that come with them is that only the innermost modality of a string of modalities matters. This follows from the iteration of the following basic facts.

Theorem 8.16. The following are modal truths, for every sentence ϕ .

- | | |
|--|--|
| (1) $\Box\Box\phi \leftrightarrow \Box\phi$
(2) $\Box\Diamond\phi \leftrightarrow \Diamond\phi$ | (3) $\Diamond\phi \leftrightarrow \Diamond\Diamond\phi$
(4) $\Box\phi \leftrightarrow \Diamond\Box\phi$ |
|--|--|

Again we leave the rigorous proofs to the reader. But, from the intuitive semantic point of view this is because the modalities \Box and \Diamond behave like quantifiers over a fixed domain with only one variable that they can quantify. Thus any quantification after the first is vacuous. There are other ways of defining modal models of in which (roughly) the domain of quantification over worlds isn't always the same. The schemas in theorem 8.16 don't represent modal truths in these modal logics. We do not discuss alternative modal model definitions in this text.

8.2.3 Derivations in $S5$

We would like a derivation system which is sound and complete for the semantics just defined in the last section, i.e. so that a sentence of MSL is a modal truth (def. 8.9) iff it's derivable in that system. We get such a system, usually called $S5$, by adding introduction and elimination rules for \Box and \Diamond to SD.

Name	Given	May Add
\Box -Elim	$\Box\phi$	ϕ
\Box -Intro	ϕ (*)	$\Box\phi$
\Diamond -Elim	$\Diamond\phi, \phi \rightarrow \psi$ (*), (**)	ψ
\Diamond -Intro	ϕ	$\Diamond\phi$

Table 8.1: Basic Rules of $S5$

There are two restrictions to the rules: (*) in \Box -Intro and \Diamond -Elim, the rule can only be applied if all the open assumptions have modal prefixes. (**) In \Diamond -Elim, the rule can only be applied if ψ has a modal prefix.

Definition 8.17. A sentence ϕ has a *modal prefix* iff

- (1) it begins with \Box ;
- (2) it begins with \Diamond ; or
- (3) it is of the form $\sim\phi, \sim\sim\phi, \dots$ where ϕ is type (1) or (2).

Example 8.18. As they are all modal truths, every instance of the schemas in example 8.15 (on the previous page) and theorem 8.16 (on the previous page) can be derived in $S5$.

Just as before we can add shortcut rules to $S5$. These we call *modal negation* rules, similar to the quantification negation rules (table 6.62, on page 170). It can be shown that anything we can prove using $S5$ and the following shortcut rules can be proved in $S5$ alone.

Name	Given	May Add
MN	$\sim\Box\phi$	$\Diamond\sim\phi$
	$\sim\Diamond\phi$	$\Box\sim\phi$
	$\sim\Diamond\sim\phi$	$\Box\phi$
	$\sim\Box\sim\phi$	$\Diamond\phi$

Table 8.2: Modal Negation Shortcut Rules

Proving soundness for $S5$ is very similar to proving it for SD . In fact, we start with the proof for SD and add four more parts: one establishing the soundness of each new modal rule. Proving completeness for $S5$ requires a more complicated modification of the methods for SD . For logically valid sentences we still need only provide a derivation, but for invalid sentences we must construct a model with various worlds and a model falsifying the sentence.

8.2.4 History of Modal Logic

It's worth discussing some historical background. (Part of this overview is drawn from Roberta Ballarín's SEP entry (2010); the reader should also consult Goldblatt 2006.) Just as with sentential and quantificational logic, work on modal logic started as work on deduction systems without any semantics, and started with sentential logic without quantification. C.I. Lewis is well known for his early work in the 1910's on derivation systems for sentential modal logic. In his (1932), coauthored with Cooper Langford, Lewis lays out his famous systems $S1$ – $S5$. ($S5$, still of much interest today, is the system we will focus on here.) It wasn't until Ruth Barcan Marcus (1946b; 1946a; 1947) and to a lesser extent Rudolf Carnap (1946; 1947) that systems were developed for quantificational modal logic (see also Marcus's (1993)).

While Marcus and Carnap did work on semantics for quantified modal logic (with Carnap drawing on Leibniz's conception of a "possible world"), it wasn't until the work of Stig Kanger (1957), Richard Montague (1960), Jaakko Hintikka (1961), A.N. Prior (1957), and (especially) Saul Kripke (1959; 1963a; 1963b; 1965) that modern looking semantics (for both sentential and quantificational modal logic) were devised.³ These are often called relational, or Kripke, semantics and are what we'll look at here (at

³This claim isn't quite true: matrix, or algebraic, semantics for modal logics have a long history, and they surely count as "modern looking".

first in simplified form), although only for sentential modal logic. Like quantificational logic, there are alternative semantics. Richard Montague (1970) and Dana Scott (1970) independently developed what's typically called neighborhood semantics (Arló-Costa and Pacuit 2006), while matrix (or algebraic) semantics have a long history that goes back before the development of relational semantics (see Cocchiarella and Freund 2008: ch. 3 for a textbook treatment). We will not bring these up at all, instead focusing on the basics of relational semantics.

For those readers looking to pursue modal logic further, we recommend the following three textbooks (listed from most basic to most advanced): (Beall and van Fraassen 2003), (Hughes and Cresswell 1996), and (Cocchiarella and Freund 2008). For handbook-style treatments, see (Cresswell 2001), (Bull and Segerberg 2001), (Zakharyashev et al. 2001), and (Garson 2001).

8.3 Quantifier Logic with Identity

8.3.1 Introduction

We aren't able to say in QL that one object is identical to another, which is often an important fact. We address this limitation of SL by extending the language.

Before we give this extension, it's worth mentioning a few motivations. First, say we have two constants t and s . It turns out that there's no sentence ϕ of QL (or even set of sentences of QL) which is true on all and only the models \mathbf{m} where $\mathbf{m}(t) = \mathbf{m}(s)$. If we translate English names as constants in QL, then translations of the following sentences

- 53. Cicero is Tully.
- 54. Clark Kent is Superman.
- 55. Mark Twain is Samuel Clemens.

will allow for models where (say) Cicero is not Tully, or Clark Kent is not Superman. Perhaps more importantly, this means that there's no such translation of 55, along with a translation of

- 56. Mark Twain wrote *The Adventures of Tom Sawyer*.

which entails a translation of

- 57. Samuel Clemens wrote *The Adventures of Tom Sawyer*.

(We can get around this difficulty if we instead translate names in English as predicates of QL that have only one member. This is a somewhat unintuitive work-around, however.)

Next, say we want a sentence ϕ (or set of sentences Δ) which contains a predicate P and which is satisfied by all and only models \mathbf{m} such that $\mathbf{m}(P)$ contains only a single element. We leave it to readers to convince themselves that there are no sentences

like this in QL. The point generalizes to any size set: for any number n , there is no sentence (or set of sentences) of QL which is satisfied by all and only models \mathbf{m} such that $\mathbf{m}(P)$ contains exactly n elements. (This holds when n is infinite too.) But, if we extend QL to capture identity claims, then we will be able to find such sentences.

Thinking about translations of English sentences again, this inability to capture cardinality claims suggests that QL can't provide satisfactory translations for English sentences like:

- 58. There exists exactly one bad apple.
- 59. At least five different philosophers tried to change that light bulb.
- 60. There exists an infinite number of things.

So by extending QL to capture identity claims we'll also be able to capture cardinality claims, and translate sentences like these.

8.3.2 The Language QLI

To get the language QLI, we add to the basic symbols of QL (def. 4.1, on page 86) the identity symbol '=' and add a new base clause to the recursive definition of a QL formula (def. 4.2, on page 86).

Definition 8.19. The *formulas of QLI* are given by the following recursive definition:

Base Clauses:

- (1) A sentence letter (atomic sentence of SL) is an atomic formula.
- (2) An n -place predicate followed by n occurrences (tokens) of individual constants or variables is an atomic formula.
- (3) Given two individual terms t and s (see Sec. 4.2.2), $t = s$ is an atomic formula.

So $b = d$, $b = x$, $y = d$, and $x = x$ are all examples of atomic formulas.

Generating Clauses:

- (1) If ϕ is a formula, then so is $\sim\phi$.
- (2) If ϕ and θ are formulas, then so are $(\phi \rightarrow \theta)$ and $(\phi \leftrightarrow \theta)$.
- (3) If all of $\phi_1, \phi_2, \phi_3, \phi_4, \dots, \phi_n$ are formulas (the list must include at least two formulas and be finite), then so are $(\phi_1 \wedge \phi_2 \wedge \phi_3 \wedge \phi_4 \wedge \dots \wedge \phi_n)$ and $(\phi_1 \vee \phi_2 \vee \phi_3 \vee \phi_4 \vee \dots \vee \phi_n)$.
- (4) If ϕ is a formula and it does not contain an expression of the form $\forall\alpha$ or $\exists\alpha$ for some QL variable α , then $\forall\alpha\phi$ and $\exists\alpha\phi$ are formulas.

Closure Clause: A string of symbols is a formula iff it can be generated by the clauses above.

Note that this definition of formulas of QLI is exactly the same as that for formulas of QL (def. 4.2, on page 86), except for the new third base clause.

8.3.3 Truth, Logical Truth, and Entailment in QLI

The models for QLI are just the models for QL (recall def. 4.5, on page 88), but obviously we have to extend the definition of truth to account for sentences with the identity symbol, ‘=’.

Definition 8.20. The following clauses fix when a sentence of QLI is *true* (or *false*) on a model \mathbf{m} :

- (1) A sentence letter ϕ is true on \mathbf{m} iff \mathbf{m} assigns true to it, i.e. iff $\mathbf{m}(\phi) = \top$.
- (2) An atomic sentence Pt with a 1-place predicate P and an individual term t is true on \mathbf{m} iff what \mathbf{m} assigns to the individual term t is in the set \mathbf{m} assigns to the predicate, i.e. iff $\mathbf{m}(t) \in \mathbf{m}(P)$.
- (3) An atomic sentence $Pt_1 \dots t_n$ with an n -place predicate P is true on \mathbf{m} iff $\langle \mathbf{m}(t_1), \mathbf{m}(t_2), \dots, \mathbf{m}(t_n) \rangle \in \mathbf{m}(P)$.
- (4) An atomic sentence $t = s$ is true on \mathbf{m} iff $\mathbf{m}(t) = \mathbf{m}(s)$.
- (5) A negation $\sim\phi$ is true on \mathbf{m} iff the unnegated formula ϕ is false on \mathbf{m} .
- (6) A conjunction $(\phi_1 \wedge \dots \wedge \phi_n)$ is true on \mathbf{m} iff all conjuncts ϕ_1, \dots, ϕ_n are true on \mathbf{m} .
- (7) A disjunction $(\phi_1 \vee \dots \vee \phi_n)$ is true on \mathbf{m} iff at least one disjunct ϕ_1, \dots, ϕ_n is true on \mathbf{m} .
- (8) A conditional $(\psi \rightarrow \phi)$ is true on \mathbf{m} iff the LHS ψ is false or the RHS ϕ is true on \mathbf{m} .
- (9) A biconditional $(\psi \leftrightarrow \phi)$ is true on \mathbf{m} iff both sides, ψ and ϕ , have the same truth value on \mathbf{m} .
- (10) A universal quantification $\forall\alpha\phi$ is true on \mathbf{m} iff $\phi t/\alpha$ is true on *all* t -variants of \mathbf{m} (where t is the first *constant* not contained in ϕ).
- (11) An existential quantification $\exists\alpha\phi$ is true on \mathbf{m} iff $\phi t/\alpha$ is true on *some* t -variant of \mathbf{m} (where t is the first *constant* not contained in ϕ).
- (12) A sentence ϕ is false on \mathbf{m} iff ϕ is not true on \mathbf{m} .

This definition is exactly the same as the one given for QL, except for the added clause (4) which covers the new atomic sentence added in definition 8.19.

The definitions of quantificational truth (def. 3.35), falsehood (def. 3.36), and contingency (def. 3.37) are the same as they were for QL. The definition of entailment is also the same, and likewise for the other relations defined in section 4.2.4.

Example 8.21. Consider any model \mathbf{m} which includes Cicero and Tully in its universe and assigns c to Cicero and d to Tully. (Cicero was a great Roman orator from the first century BC, and ‘Tully’ is an anglicized version of his name.) Then there exists a sentence ϕ of QLI such that all and only those models \mathbf{m} in which Cicero is Tully

make ϕ true. One such sentence is $c = d$. Hence, $c = d$ is a reasonable translation of the English sentence ‘Cicero is Tully.’

Example 8.22. Consider the sentence $(c = d \wedge Rc) \rightarrow Rd$ and consider just those models \mathbf{m} which include Cicero and Tully in the universe, assign c to Cicero and d to Tully, and assign to R the set of all Romans. Then any of those models which make the sentence true are such that either $\mathbf{m}(d) \in \mathbf{m}(R)$, or either $\mathbf{m}(c) \neq \mathbf{m}(d)$ or $\mathbf{m}(c) \notin \mathbf{m}(R)$. Hence the sentence is a reasonable translation of the English sentence ‘If Cicero is Tully and Cicero is Roman, then Tully is Roman.’

Example 8.23. Consider the sentence $\exists x(Ax \wedge \forall y(Ay \rightarrow y = x))$. A model \mathbf{m} makes the sentence true iff $\mathbf{m}(A)$ contains exactly one element. Hence the sentence is a reasonable translation of the English sentence ‘There exists exactly one apple’, or any other English sentence that differs from this one only in a change of the predicate ‘apple’.

Example 8.24. Consider the sentence

$$\exists w \exists z (Aw \wedge Az \wedge w \neq z \wedge \forall v (Av \rightarrow (v = w \vee v = z))).$$

The two existential quantifiers with the negative identity guarantee that there are at least two instances of A , and the universal quantifier guarantees that these are the only ones. So a model \mathbf{m} makes the sentence true iff $\mathbf{m}(A)$ contains exactly two elements. Hence the sentence is a reasonable translation of the English sentence ‘There exists (exactly) two apples.’

Example 8.25. Consider the sentence

$$\begin{aligned} &\exists x (Ax \wedge \forall y (Ay \rightarrow y = x)) \rightarrow \\ &\sim \exists w \exists z (Aw \wedge Az \wedge w \neq z \wedge \forall v (Av \rightarrow (v = w \vee v = z))). \end{aligned}$$

This sentence is a reasonable translation of the English sentence ‘If there exists exactly one apple, then there doesn’t exist exactly two apples.’

Example 8.26. Consider the sentence $Cba \wedge Nb \wedge \forall x ((Cxa \wedge Nx) \rightarrow x = b)$. This sentence is true in those models \mathbf{m} where only one element of the domain is both in the set $\mathbf{m}(N)$ and stands in relation C to $\mathbf{m}(a)$. So, if Cba means “ b is a child of a ”, Nb means “ b is male”, a is Arnold, and b is Bob, this sentence would make a reasonable translation of the English sentence ‘Bob is Arnold’s only son.’

Example 8.27. Consider the sentence $\exists y_1 \exists y_2 \forall x (x = y_1 \vee x = y_2)$. It is true in all and only those models \mathbf{m} that have two or less elements in their universe. So, the sentence would make a reasonable translation of the English sentence ‘At most only two things exist.’

Example 8.28. Consider the sentence $\exists x \exists y x \neq y$. It is true in all and only those models \mathbf{m} with two or more elements in their domain. So, it would make a reasonable translation of the English sentence ‘At least two things exist.’

Example 8.29. Consider the sentence $\exists x \exists y \exists z (x \neq y \wedge x \neq z \wedge y \neq z)$. It’s true in all and only those models \mathbf{m} with three or more elements in their domain. So, it would make a reasonable translation of the English sentence ‘At least three things exist.’

Example 8.30. Consider the sentence

$$\forall x \{ Gx \rightarrow \forall y [(Py \wedge Ryx) \rightarrow \exists z_1 \exists z_2 \exists z_3 (Iz_1 \wedge Iz_2 \wedge Iz_3 \wedge Gxz_1y \wedge Gxz_2y \wedge Gxz_3y \wedge z_1 \neq z_2 \wedge z_1 \neq z_3 \wedge z_2 \neq z_3)] \}.$$

It would make a reasonable translation of the English sentence ‘Every genie grants anyone who releases them at least three wishes.’

Example 8.31. Even with identity, there’s no one sentence of GQLI which is true in all and only those models with infinite domains.⁴ But, there’s an infinite set of sentences Δ such that a model makes every sentence of Δ true iff it has an infinite domain. Define:

$$\begin{aligned} \bigwedge_{i \neq j}^n x_i \neq x_j \quad &\Leftrightarrow \quad (x_1 \neq x_2 \wedge x_1 \neq x_3 \wedge x_1 \neq x_4 \wedge \dots \wedge x_1 \neq x_n \wedge \\ &x_2 \neq x_3 \wedge x_2 \neq x_4 \wedge x_2 \neq x_5 \wedge \dots \wedge x_2 \neq x_n \wedge \\ &x_3 \neq x_4 \wedge x_3 \neq x_5 \wedge x_3 \neq x_6 \wedge \dots \wedge x_3 \neq x_n \wedge \\ &\vdots \\ &x_{n-2} \neq x_{n-1} \wedge x_{n-2} \neq x_n \wedge \\ &x_{n-1} \neq x_n) \end{aligned}$$

Then a model makes every sentence in the following list true iff its universe is infinite:

$$\exists x_1 \exists x_2 \bigwedge_{i \neq j}^2 x_i \neq x_j$$

$$\exists x_1 \exists x_2 \exists x_3 \bigwedge_{i \neq j}^3 x_i \neq x_j$$

$$\exists x_1 \exists x_2 \exists x_3 \exists x_4 \bigwedge_{i \neq j}^4 x_i \neq x_j$$

⁴There are some single sentences of GQLI which are such that the only models that make them true have infinite domains, but not every model with an infinite domain makes these true. So, these sentences can’t really be thought of as saying that there exists an infinite number of things. They say more than that.

$$\begin{array}{c} \exists x_1 \exists x_2 \exists x_3 \exists x_4 \exists x_5 \bigwedge_{i \neq j}^5 x_i \neq x_j \\ \vdots \end{array}$$

We leave it to the reader to show that this is true.

8.3.4 Derivations in QDI

We get a sound and complete derivation system for QDI by adding an introduction and elimination rule for identity.

Name	Given	May Add
$=\text{-Intro}$		$t = t$
$=\text{-Elim}$	$\phi, t = s$	$\phi t/s$

Table 8.3: Basic Rules of QDI

Note that since only sentences can appear on lines of derivations, t and s must be constants. Also, note that $=\text{-Intro}$ is like *Assumption* insofar as neither is “applied” to previous lines; both allow you to write a sentence that fits the may-add schema at any point in a derivation.

Like QD, QDI is both sound and strongly complete.

Theorem 8.32. QDI Soundness Theorem: *For all sentences ϕ in QLI and sets of sentences Δ , if $\Delta \vdash \phi$ in QDI, then $\Delta \models \phi$.*

Theorem 8.33. QDI Strong Completeness Theorem: *For all sentences ϕ in QLI and sets of sentences Δ , if $\Delta \models \phi$ in QDI, then $\Delta \vdash \phi$.*

We won’t prove either the soundness or completeness of QDI, but a few remarks are in order. There is nothing essentially different about the soundness proof of QDI, compared with the proof for QD. We can still use the general approach from section 7.4 (on page 194) to prove the (strong) completeness of QDI, but there are two important differences. Here we briefly mention where changes need to be made without describing how those changes can be made.

The first difference concerns our search for contradictions (recall Step 4 from Sec. 7.4.3, on page 197). It’s not enough to take the conjunction of all the matrix instances (by $\wedge\text{-Intro}$), use *Distribution* to get the conjunction into DNF and check whether every disjunct contains a contradiction. The reason is that there might be some disjuncts that don’t contain a contradiction, but do contain an ion ϕ , an ion $\sim\psi$ where $\phi = \phi t/s$, and the ion $t = s$. For example, consider the disjunct $Ga \wedge \sim Gb \wedge a = b$. A

contradiction can be derived from these disjuncts using $=\text{-Elim}$, so we need to adjust our procedure accordingly.

The second difference concerns how we construct the model when The Method (or The Strong Method) doesn't produce a contradiction. The old procedure works by assigning distinct elements to each constant that appears in the list of sentences produced by The Method. So, if we could construct a model that made all the sentences produced by The Method (when it doesn't end in a contradiction) true by the old procedure, that model would assign distinct elements to each constant. But there are sets Δ of QLI sentences that both contain ions of the form $t = s$ and are consistent. Clearly no model that assigns distinct elements to the constants t and s could make all the sentences in such a set Δ true.

Our procedure is to take the model generated by the previous method and modify it by systematically changing the constant assignments so that if $s = t$ is in the Master Matrix list s and t are both assigned the same constant.

8.4 Exercises

8.4.1 TFT in Many-Valued Logic

Each of the following sentences are 2-valued TFT. Which are also 3-valued TFT? Which of the 3-valued TFT are n -valued TFT for all finite n ?

1. $\sim\sim A \leftrightarrow A$
2. $(A \rightarrow B) \rightarrow (\sim A \vee B)$
3. $(\sim A \vee B) \rightarrow (A \rightarrow B)$
4. $\sim(A \vee B) \leftrightarrow (\sim A \wedge \sim B)$
5. $A \rightarrow (\sim A \rightarrow B)$
6. $A \rightarrow ((A \rightarrow B) \rightarrow B)$
7. $(A \wedge \sim A) \rightarrow B$
8. $(A \rightarrow (B \vee C)) \rightarrow ((A \rightarrow B) \vee C)$
9. $(A \leftrightarrow B) \vee (A \leftrightarrow C) \vee (A \leftrightarrow D) \vee (B \leftrightarrow C) \vee (B \leftrightarrow D) \vee (C \leftrightarrow D)$
10. $((A \vee B) \wedge \sim A) \rightarrow B$

8.4.2 Modal Truths in MSL #1

For each schema below, show that it's a modal truth for all MSL sentences ϕ and ψ . These are from example 8.15 (on page 220).

1. $\Box\Box\phi \rightarrow \Box\phi$
2. $\Box\Diamond\phi \rightarrow \Diamond\phi$
3. $\Diamond\phi \rightarrow \Diamond\Diamond\phi$
4. $\Box\phi \rightarrow \Diamond\Box\phi$
5. $\sim\Diamond\phi \leftrightarrow \Box\sim\phi$
6. $\Diamond\sim\phi \leftrightarrow \sim\Box\phi$
7. $\Box(\phi \rightarrow \psi) \rightarrow (\Box\phi \rightarrow \Box\psi)$
8. $(\Box\phi \vee \Box\psi) \rightarrow \Box(\phi \vee \psi)$

9. $(\Diamond \phi \vee \Diamond \psi) \leftrightarrow \Diamond (\phi \vee \psi)$
10. $\Box (\phi \wedge \psi) \leftrightarrow (\Box \phi \wedge \Box \psi)$
11. $\Diamond (\phi \wedge \psi) \rightarrow (\Diamond \phi \wedge \Diamond \psi)$
12. $\sim \Diamond \sim \phi \leftrightarrow \Box \phi$
13. $\Box (\phi \rightarrow \psi) \rightarrow (\Diamond \phi \rightarrow \Diamond \psi)$
14. $\Diamond (\phi \rightarrow \psi) \rightarrow (\Box \phi \rightarrow \Diamond \psi)$
15. $\sim \Diamond \phi \rightarrow \Box (\phi \rightarrow \psi)$
16. $\Box (\phi \leftrightarrow \psi) \rightarrow (\Box \phi \leftrightarrow \Box \psi)$

8.4.3 Modal Truths in MSL #2

For each schema below, show that it's a modal truth for all MSL sentences ϕ . These are from theorem 8.16, on page 220.

1. $\Box \Box \phi \leftrightarrow \Box \phi$
2. $\Box \Diamond \phi \leftrightarrow \Diamond \phi$
3. $\Diamond \phi \leftrightarrow \Diamond \Diamond \phi$
4. $\Box \phi \leftrightarrow \Diamond \Box \phi$

8.4.4 Modal Truths in MSL #3

Show whether each of the following is a modal truth.

1. $\vdash \Diamond A \leftrightarrow \sim \Box \sim A$
2. $\vdash \Box A \rightarrow \Box \Box A$
3. $\vdash \Box (A \rightarrow B) \rightarrow (\Box A \rightarrow \Box B)$
4. $\vdash \Diamond (A \vee B) \leftrightarrow (\Diamond A \vee \Diamond B)$
5. $\vdash (\Diamond A \wedge \Diamond B) \rightarrow \Diamond (A \wedge B)$
6. $\vdash \Box (A \rightarrow B) \rightarrow (A \rightarrow \Box B)$
7. $\vdash A \rightarrow \Box A$
8. $\vdash \Diamond (A \rightarrow B) \rightarrow (\Box A \rightarrow \Diamond B)$
9. $\vdash \Box (A \rightarrow B) \rightarrow (\Diamond A \rightarrow \Diamond B)$
10. $\vdash \Box (A \wedge B) \leftrightarrow (\Box A \wedge \Box B)$
11. $\vdash \Box \Diamond A \rightarrow \Diamond \Box A$

8.4.5 Derivations in MSL

Write derivations for each schema below in $S5$. Try finding derivations both with and without the modal negation rules. Again, these are from example 8.15 (on page 220).

1. $\Box \Box \phi \rightarrow \Box \phi$
2. $\Box \Diamond \phi \rightarrow \Diamond \phi$
3. $\Diamond \phi \rightarrow \Diamond \Diamond \phi$
4. $\Box \phi \rightarrow \Diamond \Box \phi$
5. $\sim \Diamond \phi \leftrightarrow \Box \sim \phi$
6. $\Diamond \sim \phi \leftrightarrow \sim \Box \phi$
7. $\Box (\phi \rightarrow \psi) \rightarrow (\Box \phi \rightarrow \Box \psi)$
8. $(\Box \phi \vee \Box \psi) \rightarrow \Box (\phi \vee \psi)$
9. $(\Diamond \phi \vee \Diamond \psi) \leftrightarrow \Diamond (\phi \vee \psi)$
10. $\Box (\phi \wedge \psi) \leftrightarrow (\Box \phi \wedge \Box \psi)$
11. $\Diamond (\phi \wedge \psi) \rightarrow (\Diamond \phi \wedge \Diamond \psi)$
12. $\sim \Diamond \sim \phi \leftrightarrow \Box \phi$

13. $\Box(\phi \rightarrow \psi) \rightarrow (\Diamond \phi \rightarrow \Diamond \psi)$ 15. $\sim \Diamond \phi \rightarrow \Box(\phi \rightarrow \psi)$
 14. $\Diamond(\phi \rightarrow \psi) \rightarrow (\Box \phi \rightarrow \Diamond \psi)$ 16. $\Box(\phi \leftrightarrow \psi) \rightarrow (\Box \phi \leftrightarrow \Box \psi)$

8.4.6 Derivations in QLI

Write derivations for each of the following in QDI.

1. $\vdash \forall x x = x$
2. $\vdash \forall x \forall y (x = y \rightarrow y = x)$
3. $\vdash \forall x \forall y \forall z ((x = y \wedge y = z) \rightarrow x = z)$
4. $\vdash \forall x (Gx \leftrightarrow \exists y (x = y \wedge Gy))$
5. $\vdash \forall x (Gx \leftrightarrow \forall y (x = y \rightarrow Gy))$
6. $\vdash \forall x \forall y (x = y \rightarrow (Gx \leftrightarrow Gy))$
7. $\vdash \exists x \forall y (Gy \leftrightarrow y = x) \rightarrow (\exists x Gx \wedge \forall x \forall y ((Gx \wedge Gy) \rightarrow x = y))$
8. $\vdash (\exists x Gx \wedge \forall x \forall y ((Gx \wedge Gy) \rightarrow x = y)) \rightarrow \exists x \forall y (Gy \leftrightarrow y = x)$
9. $\vdash \forall x \exists y (y \neq x \wedge Gy) \rightarrow \exists x \exists y (x \neq y \wedge (Gx \wedge Gy))$
10. $\vdash \exists x \exists y (x \neq y \wedge (Gx \wedge Gy)) \rightarrow \forall x \exists y (y \neq x \wedge Gy)$
11. $\vdash (\forall x \exists y Gxy \wedge \forall x \sim Gxx) \rightarrow \forall x \exists y (x \neq y \wedge Gxy)$
12. $\vdash (Ga \wedge \sim Gb) \rightarrow \exists x \exists y x \neq y$
13. $\vdash (Ga \wedge \forall x (x \neq a \rightarrow Gx)) \leftrightarrow \forall x Gx$
14. $\vdash \forall x (x \neq a \rightarrow Gx) \rightarrow \forall x \forall y (x \neq y \rightarrow (Gx \vee Gy))$
15. $\vdash \exists x \forall y (y \neq x \rightarrow Gy) \rightarrow \forall x \forall y (x \neq y \rightarrow (Gx \vee Gy))$
16. $\vdash \forall x \forall y (x \neq y \rightarrow (Gx \vee Gy)) \rightarrow \exists x \forall y (y \neq x \rightarrow Gy)$
17. $\vdash \exists y \forall x x = y \rightarrow (\forall x Gx \vee \forall x \sim Gx)$
18. $\vdash \forall x \forall y \forall z (x = y \vee x = z \vee y = z) \rightarrow (\forall x Gx \vee \forall x (Gx \rightarrow Hx) \vee \forall x (Gx \rightarrow \sim Hx))$

8.4.7 Translations into QLI

Translate each of the following English sentences into QLI sentences about the model **m** given in table 8.4 (on the next page).

1. Bob is Arnold's only son.
2. Arnold has at least two children.
3. Arnold has at least two sons.
4. Arnold has at most two sons.
5. Arnold has exactly two sons.
6. Bob is Carol's only brother.
7. Bob is an only child.
8. Bob is an only son.
9. Bob has at least two sisters.
10. Bob has just one sister.
11. Everyone has a sister.
12. Carol's mother is Bob's only sister.
13. Bob has at least two grandchildren.
14. Diane only dates Bob.
15. Bob only dates daughters.
16. Bob dates only daughters.
17. Do (16) another way.
18. Bob only dates only daughters.
19. Bob dates only only daughters.
20. Only Bob only dates only daughters.

	Symbol	Assignment
Universe:		All people
Constants:	a	Arnold
	b	Bob
	c	Carol
	d	Diane
1 place predicates:	M'	Male
	E'	Female
2 place predicates:	P''	is the parent of
	D''	dates

Table 8.4: Interpretation for Translations in Section 8.4.7

Appendix A: List of Theorems

Sentences

Theorem 2.17 (on page 21). The number of SL sentences is equal to the number of natural numbers.

Theorem 2.69 (on page 44). Main Connective Theorem: For any non-atomic official SL sentence ϕ (any sentence with order 2 or greater), either ϕ has no token logical connectives other than negation, or there is one and only one token truth connective (or string of tokens) that's not a negation in ϕ that has one more token of '(' than ')' to the left of it.

Theorem ?? (on page ??). The Disjunctive Normal Form Theorem: Every sentence of SL is truth functionally equivalent to a SL sentence which is in DNF.

Theorem 2.78 (on page 52). The Truth-functional Expressive Completeness Theorem: Any truth-functional connective of any fixed number of arguments (ternary, quaternary, etc) is already expressible in SL.

Theorem 7.21 (on page 195). Prenex Normal Form Theorem: For all sentence θ of QL, there is a provably equivalent sentence θ^* in prenex normal form; that is, θ^* is in prenex normal form and $\vdash \theta \leftrightarrow \theta^*$ in QD.

Theorems about Truth, Logical Truth, and Entailment

Theorem ?? (on page ??). For every interpretation \mathbf{m} , definition 2.21 (on page 23) fixes a unique truth value on \mathbf{m} for every sentence ϕ of SL.

Theorem 2.68 (on page 42). If two interpretations \mathbf{m} and \mathbf{m}' agree on all of the sentence letters of ϕ , then ϕ is true in \mathbf{m} iff ϕ is true in \mathbf{m}' .

Theorem 2.39 (on page 31). For all SL sentences θ , $\models \theta$ iff θ is TFT.

Theorem 2.48 (on page 33). SL Exportation Theorem: For all SL sentences ϕ and θ , $\phi \models \theta$ iff $\models (\phi \rightarrow \theta)$.

Theorem 3.50 (on page 81). QL Exportation Theorem: For all QL sentences ϕ and θ , $\phi \models \theta$ iff $\models (\phi \rightarrow \theta)$.

See also theorem 2.49 (on page 34).

Theorem 2.70 (on page 46). Truth Functional Equivalence Replacement: If ϕ and ϕ^* are truth functionally equivalent, and θ^* is the result of replacing one occurrence of ϕ by ϕ^* in θ , then θ and θ^* are truth functionally equivalent.

Theorem ?? (on page ??). For all sentences ϕ of QL and interpretations \mathbf{m} , ϕ is true on \mathbf{m} relative to at least one assignment \mathbf{m} iff it's true on \mathbf{m} relative to all assignments \mathbf{m} .

Theorem ?? (on page ??).

- (1) For all formulas ϕ of QL with at most variable α free and for all interpretations \mathbf{m} and assignments \mathbf{m} , ϕ is true on \mathbf{m} relative to the assignment \mathbf{m} iff it's true on \mathbf{m} relative to every interpretation \mathbf{m}' that matches \mathbf{m} on α .
- (2) For all formulas ϕ of QL that have at most variables $\alpha_1, \dots, \alpha_n$ free and for all interpretations \mathbf{m} and assignments \mathbf{m} , ϕ is true on \mathbf{m} relative to the assignment \mathbf{m} iff it's true on \mathbf{m} relative to all assignments \mathbf{m}' that match \mathbf{m} on all of $\alpha_1, \dots, \alpha_n$.

Theorem ?? (on page ??). If α is the only free variable of a formula ϕ , then a partial α -assignment on an interpretation \mathbf{m} is sufficient, given definition ?? (on page ??), to fix a truth value for ϕ on \mathbf{m} .

Theorem ?? (on page ??). For all QL sentences $\phi, \phi_1, \dots, \phi_n, \psi$ and for all interpretations \mathbf{m} ,

- (1) $(\phi \rightarrow \psi)$ is true in \mathbf{m} iff: if ϕ is true in \mathbf{m} , then ψ is true in \mathbf{m} .
- (2) $(\phi \leftrightarrow \psi)$ is true in \mathbf{m} iff: ϕ is true in \mathbf{m} iff ψ is true in \mathbf{m} .
- (3) $(\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_n)$ is true in \mathbf{m} iff all of the conjuncts $\phi_1, \phi_2, \dots, \phi_n$ are true in \mathbf{m} .
- (4) $(\phi_1 \vee \phi_2 \vee \dots \vee \phi_n)$ is true in \mathbf{m} iff at least one disjunct $\phi_1, \phi_2, \dots, \phi_n$ is true in \mathbf{m} .
- (5) $\sim\phi$ is true in \mathbf{m} iff ϕ is not true in \mathbf{m} .

Theorem ?? (on page ??). For all QL formulas ϕ that at most have α free and for all interpretations \mathbf{m} ,

- (1) $\forall\alpha\phi$ is true in \mathbf{m} iff one of the following two equivalent conditions holds:
 - (a) ϕ is true in \mathbf{m} on every partial α -assignment \mathbf{m} .

- (b) there is no partial α -assignment \mathbf{m} on which ϕ is false in \mathbf{m} .
- (2) $\exists \alpha \phi$ is true in \mathbf{m} iff there is at least one partial α -assignment \mathbf{m} on which ϕ is true.

Theorem ?? (on page ??). For all QL formulas ϕ that at most have $\alpha_1, \dots, \alpha_n$ free and for all interpretations \mathbf{m} ,

- (1) $\forall \alpha_1 \dots \forall \alpha_n \phi$ is true in \mathbf{m} iff one of the following two equivalent conditions holds:
 - (a) ϕ is true in \mathbf{m} on every partial $(\alpha_1, \dots, \alpha_n)$ -assignment \mathbf{m} .
 - (b) there is no partial $(\alpha_1, \dots, \alpha_n)$ -assignment \mathbf{m} on which ϕ is false in \mathbf{m} .
- (2) $\exists \alpha_1 \dots \exists \alpha_n \phi$ is true in \mathbf{m} iff there's at least one partial $(\alpha_1, \dots, \alpha_n)$ -assignment \mathbf{m} on which ϕ is true.

Theorem 4.9 (on page 95). The Dragnet Theorem: If

- (1) ϕ contains no occurrences, whether bound or unbound, of term s , and $\phi^* = \phi s/t$, and
- (2) \mathbf{m} and \mathbf{m}^* are variable assignments with respect to interpretation \mathbf{m} and \mathbf{m}^* that differ only in that what \mathbf{m} assigns to t , \mathbf{m}^* assigns to s ,

then: \mathbf{m} makes ϕ true on \mathbf{m} iff \mathbf{m}^* makes ϕ^* true on \mathbf{m}^* .

Theorem 7.8 (on page 182). Monotonicity of Entailment: For all SL sentences $\phi_1, \dots, \phi_n, \theta, \psi$:

$$\text{If } \phi_1, \phi_2, \dots, \phi_n \models \psi, \text{ then } \phi_1, \phi_2, \dots, \phi_n, \theta \models \psi$$

Theorem 7.9 (on page 183). Transitivity of Entailment: For all SL sentences $\phi_1, \dots, \phi_n, \theta$, and ψ_1, \dots, ψ_k :

$$\text{If } \phi_1, \phi_2, \dots, \phi_n \models \psi_1 \text{ and}$$

$$\phi_1, \phi_2, \dots, \phi_n \models \psi_2 \text{ and}$$

$$\vdots$$

$$\phi_1, \phi_2, \dots, \phi_n \models \psi_k \text{ and}$$

$$\psi_1, \psi_2, \dots, \psi_k \models \theta \text{ then:}$$

$$\phi_1, \phi_2, \dots, \phi_n \models \theta$$

Derivations

Theorem 7.4 (on page 181). Every application of every basic rule of GSD is truth-preserving, or sound.

Theorem 6.6 (on page 156). Every application of every shortcut rule from SD^+ , including both standard and exchange shortcut rules, is truth-preserving.

Theorem 6.8 (on page 157). Every application of every exchange shortcut rule from SD^+ is truth-preserving, even if we extend the notion of sanctioning for them with definition 6.7.

Theorem 6.3 (on page 151). For all SL sentences $\theta_1, \dots, \theta_n, \delta$ and rules R_1, \dots, R_p , if

- (1) δ can be derived from $\theta_1, \dots, \theta_n$ using rules R_1, \dots, R_p and the basic rules of SD, and
- (2) every application of a rule R_1 is derivable using the rules R_2, \dots, R_p and the basic rules of SD,

then δ can be derived from $\theta_1, \dots, \theta_n$ using only rules R_2, \dots, R_p and the basic rules of SD.

Theorem 6.15 (on page 171). For all QL sentences $\theta_1, \dots, \theta_n, \delta$ and rules R_1, \dots, R_p , if

- (1) δ can be derived from $\theta_1, \dots, \theta_n$ using rules R_1, \dots, R_p and the basic rules of SD, and
- (2) every application of a rule R_1 is derivable using the rules R_2, \dots, R_p and the basic rules of QD,

then δ can be derived from $\theta_1, \dots, \theta_n$ using only rules R_2, \dots, R_p and the basic rules of QD.

Theorem 6.4 (on page 152). For all standard and exchange shortcut rules R (see tables 6.36 and 6.37), every application of R is derivable using the basic rules of SD.

Theorem 6.16 (on page 171). For all standard and exchange shortcut rules R (see tables 6.36, 6.37, and 6.62), every application of R is derivable using the basic rules of QD (see tables 6.1 and 6.44).

Theorem 6.5 (on page 152). Shortcut Rule Elimination Theorem: For all SL sentences ϕ_1, \dots, ϕ_m and ψ , if ψ can be derived from ϕ_1, \dots, ϕ_m in SD^+ (that is, using the basic rules of SD and any of the standard and exchange shortcut rules), then ψ can be derived from ϕ_1, \dots, ϕ_m in SD (that is, using only the basic rules).

Theorem 6.14 (on page 171). Shortcut Rule Elimination Theorem for QD⁺: For all QL sentences ϕ_1, \dots, ϕ_m and ψ , if ψ can be derived from ϕ_1, \dots, ϕ_m in QD⁺, then ψ can be derived from ϕ_1, \dots, ϕ_m in QD.

Theorem 6.9 (on page 158). For all exchange shortcut rules R from SD⁺, if ϕ and ϕ^* are the sentences you get after substituting SL sentences into the given and may-add schemas of R , respectively, then ϕ and ϕ^* are truth functionally equivalent.

Theorem 6.11 (on page 160). Restricted Replacement Theorem for SD: For all sentences ψ of SL: if

- (1) ϕ and ϕ^* are SL sentences such that $\phi \vdash \phi^*$ and $\phi^* \vdash \phi$, and
- (2) if ϕ is a subsentence of ψ , then ψ^* is the SL sentence you get when you replace one instance (token) of ϕ with an instance (token) of ϕ^* , and ψ^* is ψ if not,

then ψ^* can be derived from ψ using only the basic rules of SD, i.e. $\psi \vdash \psi^*$.

Theorem 6.17 (on page 172). The Replacement Theorem for QD: If ϕ and ϕ^* are provably equivalent formulas of QL, and θ and θ^* differ only in that θ contains the subformula ϕ in one place where θ^* contains the subformula ϕ^* , then θ and θ^* are provably equivalent.

Recall that we used the One-step Replacement Lemmas (theorem 6.20) to prove this.

Theorem 7.10 (on page 183). Non-decreasing Assumption Principle (NDAP): If Δ_1 is the set of assumptions of an unboxed line and Δ_2 is the set of assumptions of a later unboxed line, then Δ_1 is a subset of Δ_2 , i.e., $\Delta_1 \subseteq \Delta_2$.

Theorem 7.17 (on page 190). Any two QL formulas got by substituting other QL formulas into the may-add and given schemas of \leftrightarrow -Exchange are provably equivalent; that is, $\vdash \forall((\theta \leftrightarrow \psi) \leftrightarrow ((\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)))$.

Theorem 7.22 (on page 196). For all Prenex Exchange Rules R , any two QL formulas got by substituting other QL formulas into the may-add and given schemas of R are provably equivalent.

Soundness and Completeness

Theorem 7.14 (on page 188). SD is weakly complete iff it's complete; and likewise for QD.

Theorem 7.6 (on page 182). SD Soundness Theorem: SD is sound; i.e., for every set Δ of sentences of SL and every sentence ϕ of SL, if $\Delta \vdash \phi$ in SD, then $\Delta \models \phi$.

Theorem 7.7 (on page 182). Soundness Lemma: For any sequence of derivation lines that is a derivation, the sentence ϕ on the last line is entailed by the set Δ of sentences that are on unboxed lines and are sanctioned by *Assumption*.

Recall that we used theorems 7.8, 7.9, and 7.10 to prove the Soundness Lemma.

Theorem 7.11 (on page 185). QD Soundness Theorem: QD is sound; i.e., for every set Δ of sentences of QL and every sentence ϕ of QL, if $\Delta \vdash \phi$ in SD, then $\Delta \models \phi$.

Theorem 7.16 (on page 189). The SD Weak Completeness Lemma: For any sentence ϕ of SD, either $\phi \vdash A \wedge \sim A$, or ϕ is true in some interpretation \mathbf{m} .

Theorem 7.18 (on page 194). Weak SD Completeness Theorem: For all SL sentences ϕ : if $\models \phi$, then $\vdash \phi$ in SD.

Theorem 7.19 (on page 194). SD Completeness Theorem: For every finite set Δ of sentences of SL and every sentence ϕ of SL, if $\Delta \models \phi$, then $\Delta \vdash \phi$ in SD.

Theorem 7.23 (on page 199). Derivational Lemma: If the Method starts with $\sim\theta$ and produces a contradiction, then there is a derivation of θ .

Theorem 7.32 (on page 206). Strong Derivational Lemma: If the strong method halts in a contradiction, then $\Delta \vdash \phi$.

Theorem 7.25 (on page 201). The Method Lemma 1: The matrix model \mathbf{m}_M makes true all sentences on the master matrix list M generated by the method when it doesn't halt in contradiction.

Theorem 7.33 (on page 207). The Strong Method Lemma 1: The matrix model \mathbf{m}_M makes true all sentences on the master matrix list M generated by the strong method when it doesn't halt in contradiction.

Theorem 7.26 (on page 201). The Method Lemma 2: All matrix instances in the derivation generated by the method are true in the matrix model \mathbf{m}_M .

Theorem 7.34 (on page 207). The Strong Method Lemma 2: All matrix instances in the derivation generated by the strong method are true in the matrix model \mathbf{m}_M .

Theorem 7.27 (on page 202). The Method Lemma 3: All quantified sentences in the derivation generated by the method are true in the matrix model \mathbf{m}_M .

Theorem 7.35 (on page 208). The Strong Method Lemma 3: All quantified sentences in the derivation generated by the strong method are true in the matrix model \mathbf{m}_M .

Theorem 7.28 (on page 203). Main Weak QD Completeness Lemma: For all sentences θ of QL, if the method is applied to $\sim\theta$ then either: (a) the method produces a derivation of θ in QD^+ , or (b) an interpretation \mathbf{m} can be read off which makes θ false.

Theorem 7.36 (on page 208). Main Strong QD Completeness Lemma: For all sets of sentences Δ of sentences of QL and QL sentences ϕ , if the strong method is applied to $\Delta^* =$

$\Delta \cup \{\sim\phi\}$ then either: (a) the strong method produces a derivation of ϕ from Δ in QD^+ , or (b) an interpretation \mathbf{m} can be read off which makes every sentence in Δ true and ϕ false.

Theorem 7.29 (on page 203). Weak QD Completeness Theorem: For all sentences θ of QL, if $\models \theta$, then $\vdash \theta$ in QD.

Theorem 7.30 (on page 203). QD Completeness Theorem: For all finite sets Δ of QL sentences and QL sentence ϕ , if $\Delta \models \phi$, then $\Delta \vdash \phi$.

Theorem 7.31 (on page 205). Strong QD Completeness Theorem: For any set Δ of SL sentences and any SL sentence ϕ , if $\Delta \models \phi$, then $\Delta \vdash \phi$.

Decidability, Compactness, and Löwenheim-Skolem

Theorem 7.39 (on page 210). Church's Theorem: If L is a sublanguage of QL with at least one 2-place predicate symbol, then there is no decision procedure for the set of logical truths of L.

Theorem 7.40 (on page 210). Monadic QL Equivalence Theorem: Every sentence of monadic QL is quantificationally equivalent to a sentence whose quantifiers are independent.

Theorem 7.41 (on page 211). The Monadic Decision Theorem: The modified method just described provides a decision procedure for quantificational truth in monadic QL.

Theorem 7.42 (on page 211). The Downward Löwenheim-Skolem Theorem: If a sentence of QL is true in any interpretation, then it is true in one whose domain consists of all of some of the natural numbers.

Theorem 7.43 (on page 211). If ϕ is a sentence of monadic QL and has an interpretation, then it has a finite interpretation.

Theorem 7.44 (on page 212). The Compactness Theorem for QL: For all sets of sentences Δ of QL, if for every finite subset Δ' of Δ there exists an interpretation \mathbf{m}' that makes all the sentences in Δ' true, then there's some interpretation \mathbf{m} that makes all the sentences in Δ true.

Many-valued, Modal, and Quantifier Logic with Identity

Theorem ?? (on page ??). There exists no (possibly countably infinite) set of formulas Δ of QL each of which at most has the variables α and β free such that:

For any two constants t and s , if Δ^* is the set of sentences got by substituting t for all occurrences of α and s for all occurrences of β in every sentence of Δ , then: for all interpretations \mathbf{m} , \mathbf{m} makes every sentence of Δ^* true iff $\mathbf{m}(t) = \mathbf{m}(s)$.

Theorem ?? (on page ??). If Δ is (at most) a countably infinite set of QL sentences that's consistent (i.e., there's at least one interpretation \mathbf{m} that makes every sentence in Δ true) and the constants t and s each appear at least once in one of the sentences of Δ , then there's an interpretation \mathbf{m} which makes every sentence in Δ true such that $\mathbf{m}(t) \neq \mathbf{m}(s)$.

Theorem 8.32 (on page 228). QDI Soundness Theorem: For all sentences ϕ in QLI and sets of sentences Δ , if $\Delta \vdash \phi$ in QDI, then $\Delta \models \phi$.

Theorem 8.33 (on page 228). QDI Strong Completeness Theorem: For all sentences ϕ in QLI and sets of sentences Δ , if $\Delta \models \phi$ in QDI, then $\Delta \vdash \phi$.

Appendix B: List of Derivation Rules

Derivation Systems:

SD	Set 1
SD ⁺	Sets 1,2,3
QD	Sets 1,4
QD ⁺	Sets 1,2,3,4,5
QD _m ⁺	Sets 1,2,3,4,5,6,7,8
S5	Sets 1,9
S5 ⁺	Sets 1,2,3,9,10
QDI	Sets 1,4,11
QDI ⁺	Sets 1,2,3,4,5,11

Name	Given	May Add
Set 1: Basic Rules of GSD, table 6.1 (on page 129)		
Ass.		ϕ
Rep.	ϕ	ϕ
\rightarrow -Elim	$\theta \rightarrow \psi, \theta$	ψ
\rightarrow -Intro	$\theta \Rightarrow \psi$	$\theta \rightarrow \psi$, Box $\theta \Rightarrow \psi$
\wedge -Elim	$\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n$	Conjunction of any proper subset of the conjuncts
\wedge -Intro	$\theta_1, \theta_2, \dots \theta_n$	$\theta_1 \wedge \theta_2 \wedge \dots \wedge \theta_n$
\vee -Elim	$\theta_1 \vee \theta_2 \vee \dots \vee \theta_n,$ $\theta_1 \rightarrow \psi,$ $\theta_2 \rightarrow \psi,$	

Name	Given	May Add
	\vdots	
	$\theta_n \rightarrow \psi$	ψ
\vee -Intro	θ	$\psi_1 \vee \psi_2 \vee \dots \vee \psi_n$, where $\theta = \psi_i$ for some i .
\sim -Intro	$\theta \rightarrow (\psi \wedge \sim\psi)$	$\sim\theta$
\sim -Elim	$\sim\theta \rightarrow (\psi \wedge \sim\psi)$	θ
\leftrightarrow -Intro	$\theta \rightarrow \psi, \psi \rightarrow \theta$	$\theta \leftrightarrow \psi$
\leftrightarrow -Elim	$\theta \leftrightarrow \psi, \psi$	θ
\leftrightarrow -Elim	$\theta \leftrightarrow \psi, \theta$	ψ
Set 2: Standard Shortcut Rules of GSD, table 6.36 (on page 149)		
M.T.	$\phi \rightarrow \theta, \sim\theta$	$\sim\phi$
D.S.	$\phi_1 \vee \dots \vee \phi_i \vee \dots \vee \phi_n, \sim\phi_i$	$\phi_1 \vee \dots \vee \phi_{i-1} \vee \phi_{i+1} \vee \dots \vee \phi_n$
	$\phi_1 \vee \dots \vee \sim\phi_i \vee \dots \vee \phi_n, \phi_i$	$\phi_1 \vee \dots \vee \phi_{i-1} \vee \phi_{i+1} \vee \dots \vee \phi_n$
A.C.	$\phi, \sim\phi$	ψ
\sim/\leftrightarrow -Intro	$\phi \leftrightarrow \psi$	$\sim\phi \leftrightarrow \sim\psi$
Set 3: Exchange Shortcut Rules of GSD, table 6.37 (on page 150)		
DeM	$\sim(\phi_1 \wedge \dots \wedge \phi_n)$	$\sim\phi_1 \vee \dots \vee \sim\phi_n$
	$\sim\phi_1 \vee \dots \vee \sim\phi_n$	$\sim(\phi_1 \wedge \dots \wedge \phi_n)$
	$\sim(\phi_1 \vee \dots \vee \phi_n)$	$\sim\phi_1 \wedge \dots \wedge \sim\phi_n$
	$\sim\phi_1 \wedge \dots \wedge \sim\phi_n$	$\sim(\phi_1 \vee \dots \vee \phi_n)$
$\sim\sim$ -Elim	$\sim\sim\phi$	ϕ
$\sim\sim$ -Intro	ϕ	$\sim\sim\phi$
\rightarrow/\vee - Exchange	$\phi \rightarrow \theta$	$\sim\phi \vee \theta$
	$\sim\phi \vee \theta$	$\phi \rightarrow \theta$
Contraposition	$\phi \rightarrow \theta$	$\sim\theta \rightarrow \sim\phi$
	$\sim\theta \rightarrow \sim\phi$	$\phi \rightarrow \theta$

Name	Given	May Add
\sim/\rightarrow - Exchange	$\sim(\phi \rightarrow \theta)$	$\phi \wedge \sim\theta$
	$\phi \wedge \sim\theta$	$\sim(\phi \rightarrow \theta)$
Distribution	$\theta \wedge (\phi_1 \vee \dots \vee \phi_n)$	$(\theta \wedge \phi_1) \vee \dots \vee (\theta \wedge \phi_n)$
	$(\theta \wedge \phi_1) \vee \dots \vee (\theta \wedge \phi_n)$	$\theta \wedge (\phi_1 \vee \dots \vee \phi_n)$
	$(\phi_1 \vee \dots \vee \phi_n) \wedge \theta$	$(\phi_1 \wedge \theta) \vee \dots \vee (\phi_n \wedge \theta)$
	$(\phi_1 \wedge \theta) \vee \dots \vee (\phi_n \wedge \theta)$	$(\phi_1 \vee \dots \vee \phi_n) \wedge \theta$
	$\theta \vee (\phi_1 \wedge \dots \wedge \phi_n)$	$(\theta \vee \phi_1) \wedge \dots \wedge (\theta \vee \phi_n)$
	$(\theta \vee \phi_1) \wedge \dots \wedge (\theta \vee \phi_n)$	$\theta \vee (\phi_1 \wedge \dots \wedge \phi_n)$
	$(\phi_1 \wedge \dots \wedge \phi_n) \vee \theta$	$(\phi_1 \vee \theta) \wedge \dots \wedge (\phi_n \vee \theta)$
	$(\phi_1 \vee \theta) \wedge \dots \wedge (\phi_n \vee \theta)$	$(\phi_1 \wedge \dots \wedge \phi_n) \vee \theta$
Set 4: Basic Rules of GQD, table 6.44 (on page 164)		
\forall -Elim	$\forall\beta\phi$	$\phi a/\beta$, for 'a' any individual constant
\forall -Intro	$\phi a/\beta$	$\forall\beta\phi$, iff 'a' does not occur in ϕ nor in any unboxed assumption
\exists -Intro	$\phi a/\beta$	$\exists\beta\phi$
\exists -Elim	$\exists\beta\phi, \phi a/\beta \rightarrow \theta$	θ , iff 'a' does not occur in ϕ or θ , nor in any unboxed assumption
Set 5: Exchange Shortcut Rules of GQD, table 6.62 (on page 170)		
QN	$\sim\forall\beta\phi$	$\exists\beta\sim\phi$
	$\exists\beta\sim\phi$	$\sim\forall\beta\phi$
	$\sim\exists\beta\phi$	$\forall\beta\sim\phi$
	$\forall\beta\sim\phi$	$\sim\exists\beta\phi$
Set 6: DNF Exchange Shortcut Rules for GSD, table 7.1 (on page 190)		
\leftrightarrow -Exchange	$\theta \leftrightarrow \psi$	$(\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)$
	$(\theta \wedge \psi) \vee (\sim\theta \wedge \sim\psi)$	$\theta \leftrightarrow \psi$

Name	Given	May Add
Set 7: Prenex Exchange Shortcut Rules for GQD, table 7.5 (on page 196)		
α/β -Exch	$(\# \alpha)\phi$	$(\# \beta)\phi\beta/\alpha$
Q Shuffling	$((\#x)\theta \wedge \psi)$	$(\#x)(\theta \wedge \psi)$
	$(\theta \wedge (\#x)\psi)$	$(\#x)(\theta \wedge \psi)$
	$((\#x)\theta \vee \psi)$	$(\#x)(\theta \vee \psi)$
	$(\theta \vee (\#x)\psi)$	$(\#x)(\theta \vee \psi)$
	$(\theta \rightarrow (\#x)\psi)$	$(\#x)(\theta \rightarrow \psi)$
	$(\exists x\theta \rightarrow \psi)$	$\forall x(\theta \rightarrow \psi)$
	$(\forall x\theta \rightarrow \psi)$	$\exists x(\theta \rightarrow \psi)$
Set 8: The Method Shortcut Rules for GQD, table 7.6 (on page 204)		
Greg's Rule	$\psi_1 \vee \dots \vee \psi_n$, where some $\psi_i = \phi_1 \wedge \dots \wedge \phi_j \wedge \dots$ $\wedge \sim \phi_j \wedge \dots \wedge \phi_m$	$\psi_1 \vee \dots \vee \psi_{i-1} \vee \psi_{i+1} \vee \dots \vee \psi_n$
\vee/\wedge -Elim	$\psi_1 \vee \dots \vee \psi_n$, where each ψ_i contains ϕ	ϕ
OBA	$\phi_1 \wedge \dots \wedge \phi_i \wedge \dots \wedge \phi_n, \sim \phi_i$	$\sim(\phi_1 \wedge \dots \wedge \phi_i \wedge \dots \wedge \phi_n)$
	$\phi_1 \wedge \dots \wedge \sim \phi_i \wedge \dots \wedge \phi_n, \phi_i$	$\sim(\phi_1 \wedge \dots \wedge \sim \phi_i \wedge \dots \wedge \phi_n)$
Set 9: Basic Rules for S5, table 8.1 (on page 221)		
\Box -Elim	$\Box \phi$	ϕ
\Box -Intro	$\phi (*)$	$\Box \phi$
\Diamond -Elim	$\Diamond \phi, \phi \rightarrow \psi (*), (**)$	ψ
\Diamond -Intro	ϕ	$\Diamond \phi$
Set 10: Modal Negation Exchange Shortcut Rules for S5, table 8.2 (on page 222)		
MN	$\sim \Box \phi$	$\Diamond \sim \phi$
	$\sim \Diamond \phi$	$\Box \sim \phi$
	$\sim \Diamond \sim \phi$	$\Box \phi$
	$\sim \Box \sim \phi$	$\Diamond \phi$

Name	Given	May Add
Set 11: Basic Rules for GQDI, 8.3 (on page 228)		
$=\text{-Intro}$		$t = t$
$=\text{-Elim}$	$\phi, t = s$	$\phi t/s$

(*) in $\Box\text{-Intro}$ and $\Diamond\text{-Elim}$, the rule can only be applied if all the open assumptions have modal prefixes.

(**) In $\Diamond\text{-Elim}$, the rule can only be applied if ψ has a modal prefix.

Works Cited

- Anderson, J. M. and Johnstone, H. W. 1962. *Natural Deduction: The Logical Basis of Axiom Systems*. Wadsworth.
- Arló-Costa, Horacio and Pacuit, Eric. 2006. “First-Order Classical Modal Logic”. *Studia Logica* 84 (Special Issue Ways of Worlds II. On Possible Worlds and Related Notions):171–210.
- Ballarin, Roberta. 2010. “Modern Origins of Modal Logic”. In Edward N. Zalta (ed.), *The Stanford Encyclopedia of Philosophy*. Winter 2010 edition. <http://plato.stanford.edu/archives/win2010/entries/logic-modal-origins/>.
- Barwise, Jon and Etchemendy, John. 1987. *The Liar: an Essay on Truth and Circularity*. New York: Oxford University Press.
- Bealer, George. 1998. “Propositions”. *Mind* 107:1–32.
- Beall, J.C. and van Fraassen, Bas. 2003. *Possibilities and Paradox: an Introduction to Modal and Many-Valued Logic*. Oxford New York: Oxford University Press.
- Bergmann, Merrie, Moor, James, and Nelson, Jack. 2003. *The logic book*. New York London: McGraw-Hill Higher Education McGraw-Hill, 4 edition.
- Blanchette, Patricia. 2001. “Logical Consequence”. In Lou Goble (ed.), *The Blackwell Guide to Philosophical Logic*. Blackwell, 115–135.
- Boole, George. 1854. *An Investigation of The Laws of Thought on Which are Founded the Mathematical Theories of Logic and Probabilities*. Originally published by Macmillan. Reprint by Dover, 1958.
- Borkowski, L. 1970. *Jan Lukasiewicz: Selected Works*. North-Holland Publishing Company.
- Bull, R. A. and Segerberg, K. 2001. “Basic Modal Logic”. In Dov M. Gabbay and Franz Guenther (eds.), *Handbook of Philosophical Logic*, volume 3. Kluwer Academic Publishers, 1–82.
- Carnap, Rudolf. 1946. “Modalities and Quantification”. *Journal of Symbolic Logic* 11:33–64.

- . 1947. *Meaning and Necessity: A Study in Semantics and Modal Logic*. The University of Chicago Press.
- Chomsky, Noam. 2002 [1957]. *Syntactic Structures*. Mouton de Gruyter (formerly Mouton, The Hague).
- Church, Alonzo. 1956. *Introduction to Mathematical Logic*, volume 1. Princeton University Press.
- Cocchiarella, Nino and Freund, Max. 2008. *Modal Logic: an Introduction to its Syntax and Semantics*. Oxford New York: Oxford University Press.
- Cresswell, M. J. 2001. “Modal Logic”. In Lou Goble (ed.), *The Blackwell Guide to Philosophical Logic*. Blackwell, 136–158.
- De Morgan, Augustus. 1847. *Formal Logic: or The Calculus of Inference*. Taylor & Walton.
- . 1860. *Syllabus of a Proposed System of Logic*. Walton & Malbery.
- Demopoulos, William and Clark, Peter. 2005. “The Logicism of Frege, Dedekind, and Russell”. In Stewart Shapiro (ed.), *The Oxford Handbook of Philosophy of Mathematics and Logic*. Oxford University Press, 129–165.
- Dipert, Randall. 1984. “Studies in Logic by Members of the Johns Hopkins University by Charles S. Peirce: Max H. Fisch; Achim Eschbach (Review)”. *Transactions of the Charles S. Peirce Society* 20:469–472.
- Dunn, J M and Belnap, N D. 1968. “The Substitution Interpretation of the Quantifiers”. *Noûs* 2:177–185.
- Ebbinghaus, H. D. 1985. “Extended Logics: The General Framework”. In J. Barwise and S. Feferman (eds.), *Model-Theoretic Logics*, Perspectives in Mathematical Logic. Springer-Verlag, 25–76.
- Enderton, Herbert B. 2010. *Computability Theory: An Introduction to Recursion Theory*. Academic Press (Elsevier Science).
- Fitch, Frederic. 1952. *Symbolic Logic*. Ronald Press.
- Frege, Gottlob. 1879. *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Halle. Trans. “Concept Script, a formal language of pure thought modelled upon that of arithmetic”.
- . 1892. “Über Sinn und Bedeutung”. *Zeitschrift für Philosophie und philosophische Kritik* 100:25–50. Trans. “On Sense and Reference”, available in Martinich, A. P. (ed.), 2001, *The Philosophy of Language*, Oxford: Oxford University Press, 4th edition and in Peter Geach and Max Black (eds.), 1966, *Translations from the Philosophical Writings of Gottlob Frege*, Basil Blackwell, 2nd edition.

- . 1893/1903. *Grundgesetze der Arithmetik*. Jena: Verlag Hermann Pohle. Trans. “The Basic Laws of Arithmetic”.
- . 1966 [1891]. “Function and Concept (Funktion und Begriff)”. In *Translations from the Philosophical Writings of Gottlob Frege*. Basil Blackwell. Orig. title “Funktion und Begriff”.
- Garson, J. 2001. “Quantification in Modal Logic”. In Dov M. Gabbay and Franz Guenther (eds.), *Handbook of Philosophical Logic*, volume 3. Kluwer Academic Publishers, 267–324.
- Gentzen, Gerhard. 1934. “Untersuchungen über das Logische Schliessen”. *Math Zeitschrift* 39:176–210 and 405–431.
- Gleitman, Lila R. 1965. “Coordinating Conjunction in English”. *Language* 41:260–293.
- Gödel, Kurt. 1929. *Über die Vollständigkeit des Logikkalküls*. Ph.D. thesis, University Of Vienna.
- . 1930. “Die Vollständigkeit der Axiome des logischen Funktionenkalküls”. *Monatshefte für Mathematik und Physik* 37:349–360.
- Goldblatt, Rob. 2006. “Mathematical Modal Logic: A View of its Evolution”. In Dov M. Gabbay and John Woods (eds.), *Handbook of the History of Logic*, volume 7. Elsevier, 1–98.
- Grandy, Richard. 1993. “What do ‘Q’ and ‘R’ Stand for Anyway?” In R.I.G. Hughes (ed.), *A Philosophical Companion to First-Order Logic*. Hackett, 50–61.
- Gupta, Anil. 2001. “Truth”. In Lou Goble (ed.), *The Blackwell Guide to Philosophical Logic*. Blackwell, 90–114.
- Henkin, Leon. 1949. “The Completeness of the First-Order Functional Calculus”. *Journal of Symbolic Logic* 14:159–166.
- Hilbert, David and Ackermann, Wilhelm. 1928. *Grundzüge der theoretischen Logik*. Springer-Verlag.
- Hintikka, Jaakko. 1961. “Modalities and Quantification”. *Theoria* 27:119–28.
- . 1996. *The principles of mathematics revisited*. Cambridge New York: Cambridge University Press.
- Hodges, Wilfrid. 1997. *A shorter model theory*. Cambridge New York: Cambridge University Press.
- . 2001a. “Classical Logic I: First-Order Logic”. In Lou Goble (ed.), *The Blackwell Guide to Philosophical Logic*. Blackwell, 9–32.

- . 2001b. “Elementary Predicate Logic”. In Dov M. Gabbay and Franz Guentner (eds.), *Handbook of Philosophical Logic*, volume 1. Kluwer Academic Publishers, 1–129.
- Hughes, G. and Cresswell, M.J. 1996. *A New Introduction to Modal Logic*. London New York: Routledge.
- Jacquette, Dale (ed.). 2002. *Philosophy of Logic: an anthology*. Blackwell Philosophy Anthologies. Malden, Mass: Blackwell Publishers.
- Kalish, Donald and Montague, Richard. 1964. *Logic: Techniques of Formal Reasoning*. Harcourt, Brace and World.
- Kanger, Stig. 1957. “Provability in Logic”. In *Acta Universitatis Stockholmiensis*, volume 1 of *Stockholm Studies in Philosophy*. Almqvist and Wiksell.
- King, Jeffrey C. 2007. *The Nature and Structure of Content*. Oxford University Press.
- Kleene, Stephen. 2002 [1967]. *Mathematical logic*. Mineola, N.Y: Dover Publications.
- Kripke, Saul. 1959. “A Completeness Theorem in Modal Logic”. *Journal of Symbolic Logic* 24:1–14.
- . 1963a. “Semantical Analysis of Modal Logic I: Normal Modal Propositional Calculi”. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik* 9:67–96.
- . 1963b. “Semantical Considerations on Modal Logic”. *Acta Philosophica Fennica* 16:83–94.
- . 1965. “Semantical Analysis of Modal Logic II. Non-normal Modal Propositional Calculi”. In J. W. Addison, L. Henkin, and A. Tarski (eds.), *Symposium on the Theory of Models*. Amsterdam: North-Holland, 206–220.
- . 1975. “An Outline of a Theory of Truth”. *Journal of Philosophy* 72:690–716.
- Ladd-Franklin, Christine. 1883. “On the Algebra of Logic”. In C. S. Peirce (ed.), *Studies in Logic*. Little, Brown and Co, 17–71.
- Leblanc, Hugues. 2001. “Alternative to Standard First-order Semantics”. In Dov M. Gabbay and Franz Guentner (eds.), *Handbook of Philosophical Logic*, volume 2. Kluwer Academic Publishers, 53–132.
- Lemmon, E. J. 1965. *Beginning Logic*. Nelson.
- Lewis, C. I. and Langford, Cooper H. 1932. *Symbolic Logic*. London: Century. 2nd edition 1959, New York: Dover.

- Löwenheim, Leopold. 1915. "Über Möglichkeiten im Relativkalkül". *Math Annalen* 76:447–470.
- Marcus, Ruth Barcan. 1946a. "The Deduction Theorem in a Functional Calculus of First Order Based on Strict Implication". *Journal of Symbolic Logic* 11:115–118.
- . 1946b. "A Functional Calculus of First Order Based on Strict Implication". *Journal of Symbolic Logic* 11:1–16.
- . 1947. "The Identity of Individuals in a Strict Functional Calculus of Second Order". *Journal of Symbolic Logic* 12:12–15.
- . 1993. *Modalities: Philosophical Essays*. Oxford University Press.
- Mates, Benson. 1972. *Elementary Logic*. Oxford University Press, 2 edition.
- Mitchell, O H. 1883. "On a New Algebra of Logic". In C. S. Peirce (ed.), *Studies in Logic*. Little, Brown and Co, 72–106.
- Montague, Richard. 1960. "Logical Necessity, Physical Necessity, Ethics, and Quantifiers". *Inquiry* 3:259–269.
- . 1970. "Universal Grammar". *Theoria* 36:373–98.
- Peirce, C.S. 1883. "A Theory of Probable Inference. Note B. The Logic of Relatives". In Boston Members of the Johns Hopkins University (ed.), *Studies in Logic*. Little, Brown and Co.
- . 1902. "The Simplest Mathematics". In C Hartshorne et al (ed.), *Collected Papers of Charles Sanders Peirce*, volume IV. Harvard University Press, 189–262.
- Pospesel, Howard and Marans, David. 1978. *Arguments: Deductive Logic Exercises*. Pearson Education, Inc, 2 edition.
- Post, Emil. 1921. "Introduction to a General Theory of Elementary Propositions". *American Journal of Mathematics* 43:163–185.
- Prawitz, Dag. 1965. *Natural Deduction: a Proof-theoretical Study*. Almqvist and Wiksell.
- Prior, A.N. 1957. *Time and Modality*. Clarendon Press.
- Quine, Willard Van Orman. 1950. *Methods of Logic*. Holt.
- . 1982. *Methods of Logic*. Harvard University Press, 4 edition.
- . 1986. *Philosophy of logic*. Cambridge, Mass: Harvard University Press, 2 edition.
- Russell, Bertrand. 1996 [1903]. *The Principles of Mathematics*. New York: W.W. Norton & Company.

- Schiffer, Stephen. 1987. *Remnants of Meaning*. The MIT Press.
- Scott, Dana. 1970. "Advice in Modal Logic". In K. Lambert (ed.), *Philosophical Problems in Logic*. Dordrecht, Netherlands: Reidel, 143–73.
- Shapiro, Stewart. 2005. "Logical Consequence, Proof Theory, and Modal Theory". In Stewart Shapiro (ed.), *The Oxford Handbook of Philosophy of Mathematics and Logic*. Oxford University Press, 651–670.
- Skolem, Thoralf. 1920. "Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit oder Beweisbarkeit Mathematischer Sätze nebst einem Theoreme über dichte Mengen". *Videnskapsselskapets Skrifter, I. Matem.-Naturv. Klasse 4*.
- . 1922. "Einige Bemerkungen zur axiomatischen Begründung der Mengenlehre". *Matematikerkongressen i Helsingfors den 4–7 Juli 1922*.
- . 1967 [1928]. "On mathematical logic". In Jean van Heijenoort (ed.), *From Frege to Gödel: A Source Book in Mathematical Logic, 1879 to 1931*. Harvard University Press, 508–524.
- Smith, Nicholas J.J. 2012. *Logic: The Laws of Truth*. Princeton University Press.
- Smith, Robin. 1995. "Logic". In Jonathan Barnes (ed.), *The Cambridge Companion to Aristotle*. Cambridge University Press, 27–65.
- Smullyan, Raymond M. and Fitting, Melvin. 2010. *Set Theory and the Continuum Problem*. Dover. Originally published in 1996, Oxford University Press.
- Soames, Scott. 2010. *What is Meaning?* Soochow University Lectures in Philosophy. Princeton University Press.
- Stevenson, L. 1973. "Frege's Two Definitions of Quantification". *Philosophical Quarterly* 23:207–223.
- Suppes, Patrick. 1957. *Introduction to Logic*. Van Nostrand.
- Tarski, Alfred. 1944. "The Semantic Conception of Truth and the Foundations of Semantics". *Philosophy and Phenomenological Research* 4:341–375. Available in Martinich, A. P. (ed.), 2001, *The Philosophy of Language*, Oxford: Oxford University Press, 4th edition.
- . 1983 [1933]. "The Concept of Truth in Formalized Languages". In John Corcoran (ed.), *Logic, Semantics, Metamathematics: Papers from 1923 to 1938*. Hackett Publishing, 152–278.
- Tarski, Alfred and Vaught, R. L. 1956. "Arithmetical Extensions of Relational Systems". *Compositio Math* 13:81–102.

- Tennant, Neil. 1978. *Natural Logic*. Edinburgh University Press.
- Thomason, R. H. 1970. *Symbolic Logic, An Introduction*. Macmillan.
- van Dalen, Dirk. 1980. *Logic and Structure*. Springer.
- Vaught, R. L. 1974. "Model Theory before 1945". In L. Henkin et al (ed.), *Proceedings of the Tarski Symposium*. AMS, 153–172.
- Westerståhl, Dag. 2001. "Quantifiers". In Lou Goble (ed.), *The Blackwell Guide to Philosophical Logic*. Blackwell, 437–460.
- Zakharyashev, M., Wolter, F., and Chagrov, A. 2001. "Advanced Modal Logic". In Dov M. Gabbay and Franz Guenther (eds.), *Handbook of Philosophical Logic*, volume 3. Kluwer Academic Publishers, 83–266.

Index

- $S5$, 200
- \Leftarrow, \Rightarrow , 24
- \vee/\wedge -Elim, 189
- \models , 21
- type/token distinction, 5
- use/mention distinction, 4

- algebraic semantics, *see* matrix semantics
- antecedent, 7
- argument, 56

- base clause(s), 6
- basic rule, 116, 118
- basic symbols, 6, 73
- biconditional, 7
- boolean algebra, 17

- census, 186
- Church's Theorem, 192
- closure clause, 6
- completeness, 116, 167
 - of QDI, 207, 216
 - strong, 167
 - weak, 167
 - weak QD, 187
 - weak SD, 174, 214
- conclusion, 56
- conditional, 7
- conjunction, 7
- conjuncts, 7
- connective, 46
 - truth-functional, 46
- consequent, 7
- construction tree, 11, 76
- contradictory
 - quantificational, 101
 - truth functional, 26
- contraries
 - quantificational, 101
 - truth functional, 26

- decidable, 125, 191
- decision procedure, 125, 191
 - for QT in QL, *see* Church's Theorem
 - for QT in monadic QL, 192
 - for TFT in SL, 192
 - truth table, 192
- definability, 30
- derivation, 121
 - line, 121
 - rule, 122, 123
 - $S5$, 200
 - QDI, 206
 - basic, 116, 118
 - elimination, 116, 118, 200, 206
 - for DNF, 174
 - introduction, 116, 118, 200, 206
 - shortcut, 116, 200
 - sound, *see* truth-preserving
 - schemas, 139
- derivationally equivalent, *see* provably equivalent
- disjunction, 7
- disjunctive normal form, 41
 - prenex, 182
- disjuncts, 7
- DNF, *see* disjunctive normal form, 174
- domain, 81
- double turnstile, 21

- elimination rule, 116, 118, 200, 206
- entailment, 198

- entails, [21](#)
- equivalent sentences
 - quantificational, [101](#)
 - truth functional, [26](#)
- Exchange Rules
 - Prenex, [179](#)
- exclusive disjunction, [49](#)
- existential quantifier, [73](#)
- extended model, [81](#)
 - α -variant, [82](#)
 - partial, [82](#)
- falsehood
 - logical, [18](#)
 - modal, [197](#)
 - quantificational, [98](#)
 - truth functional, [17](#)
- finite sets, [38](#)
- formulas, [74](#)
 - existential, [75](#)
 - of QLI, [202](#)
 - of QL, [74](#)
 - universal, [75](#)
 - unofficial, [76](#)
- generating clause(s), [6](#)
- given schemas, [123](#)
- glossary, [44](#)
- GQDI, [206](#)
- GQL
 - monadic, [192](#)
- Greg's Rule, [188](#)
- inclusive disjunction, [49](#)
- independent sentences
 - quantificational, [101](#)
 - truth functional, [26](#)
- indeterminate
 - logical, [18](#)
 - modal, [198](#)
 - quantificational, [98](#)
 - truth functional, [17](#)
- individual terms, [82](#)
- inference, [56](#)
- introduction rule, [116](#), [118](#), [200](#), [206](#)
- ions, [182](#)
- Kripke semantics, *see* relational semantics
- language, [6](#)
 - artificial, [4](#)
 - formal, [4](#)
 - meta-, [5](#)
 - natural, [4](#)
 - object, [5](#)
- LHS, [7](#)
- licenses, *see* sanctions
- logical
 - falsehood, [18](#)
 - indeterminate, [18](#)
 - truth, [v](#), [18](#)
- logical adequacy, [45](#)
- logical connectives, [45](#)
- logical consequence, [v](#), [1](#), [22](#), [108](#)
- logical truth, [1](#)
- main connective, [46](#)
 - of GQL, [76](#)
 - of GSL, [10](#)
 - theorem, [35](#), [209](#)
- master matrix list, [185](#)
- mathematical logic, [v](#)
- MathEnglish, [7](#)
- matrix, [182](#)
 - instances, [182](#)
 - master list, [185](#)
 - model, [185](#)
- matrix semantics, [196](#)
- may-add schema, [123](#)
- metalanguage, [5](#)
- method, the, [124](#), [181](#), [182](#)
 - strong, [190](#)
- modal prefix, [200](#)
- model, [13](#), [80](#)
 - extension, [81](#)
- Monadic Decision Theorem, The, [193](#), [215](#)
- NAND, [44](#)

- natural deduction, 21, 117
- natural numbers, 38
- neighborhood semantics, 196
- NOR, 44
- object language, 5
- One Bad Apple, 189
- order, 9, 76
- partial models, 14
- predicate language, 73
- premises, 56
- prenex disjunctive normal form, 182
- prenex normal form, 179
- principle of mathematical induction, 38
- propositional language, *see* sentential language
- provably equivalent
 - formulas of QL, 159
 - sentences of SL, 146
- QF, *see* falsehood, quantificational
- QI, *see* indeterminate, quantificational
- QT, *see* truth, quantificational
- Quantificational Derivation System, 150
- quantificationally entails, 100
- quantifier
 - s, independent, 192
- quantifier language, 73
- recursive assumption, 31
- recursive definition, 6
- recursive proof, 31
- relational semantics, 195
- RHS, 7
- sanctions, 122, 123, 151
- scope, 46, 79
- semantic adequacy, 45
- semantic tableaux, 21
- sentence, 6
 - atomic, 7, 79
 - contradictory, 2
 - formal, 1
 - liar, 2
 - of QL, 78
 - of MSL, 196
 - of SL, 7
 - paradoxical, 2
 - ungrounded, 2
 - unofficial (of QL), 79
 - unofficial (of SL), 8
 - valid, *see* truth, logical
- sentence schema, 8, 139
- Sentential Derivation System, 116
- sentential language, 5
- Sheffer stroke, *see* NAND
- single quotes, 4, 9, 81
- soundness, 116, 167
 - of QDI, 207, 216
 - of QD, 172
 - of SD, 169
 - of derivation rule, *see* truth-preserving
- string, 6
- strong method, the, 190
- subcontraries
 - quantificational, 101
 - truth functional, 26
- subformulas, 76
- subsentence, 9
- tautology, *see* truth, logical
- TFF, *see* falsehood, truth functional
- TFI, *see* indeterminate, truth functional
- TFT, *see* truth, truth functional
- token, 5
- translations
 - criteria for, 45
 - general method for, 48
 - logical adequacy, 45
 - semantic adequacy, 45
- truth
 - logical, 18
 - modal, 197
 - quantificational, 98
 - truth functional, 17
- truth-preserving, 116, 123

type, **5**

undecidable, **125**, 191

universal closure, **159**

universal quantifier, **73**

universe, **81**

use/mention distinction, 81

valid, **57**

valuation function, **197**

variables

 bound, **78**

 free, **78**

 individual (GQL), **73**

 MathEnglish, 7, 74