Richmond H. Thomason

Symbolic Logic

An Introduction

Department of Philosophy
University of Maryland
College Park, Maryland 20742

The Macmillan Company / Collier-Macmillan Limited, London

PRINTING 56789 YEAR 456789

Library of Congress catalog card number : 72-83069

The Macmillan Company

Collier-Macmillan Canada, Ltd., Toronto, Ontario

Printed in the United States of America

Preface

THIS TEXT is an introduction to modern symbolic logic. It is meant to provide a working understanding of both the mathematical theories used by logicians and their relationship to everyday reasoning. Since the study of logic is sometimes painful for nonmathematicians I have tried to accompany all mathematical material with supplementary discussion and explanation. To make the book self-contained I have included chapters on set theory and mathematical induction. Whatever mathematical techniques are used are explained, so that the book presupposes no specialized training in mathematics.

The book covers what is commonly thought to be the standard material for a beginning course in symbolic logic. It may be used as a textbook for an introductory course at the undergraduate or the graduate level, and will serve for an intensive one-term course or a more leisurely course of two terms. It has been used in a graduate course at Yale and an undergraduate course at

Amherst, and was modified and rewritten subsequent to experience in the classroom.

Plan of the book

The first seven chapters deal with the logic of sentence connectives, or *sentence logic*, and the next five chapters with the logic of quantifiers and identity, or *predicate logic with identity*. These are treated as classical, or two-valued logics; I do not discuss so-called nonclassical logics in this book. The development of both sentence and predicate logic falls into five main areas: informal semantics, proof techniques, syntactic metatheory, semantic metatheory, and semantic completeness.

Informal semantics. This has to do with the relation of the formal languages of symbolic logic to natural language (in this book, to English). Most texts use the notion of *logical form* to account for this relation, although the notion that English sentences come prepackaged in a logical form is now philosophically outmoded. The formal languages of classical logic were devised to account for mathematical reasoning, and serve very well to express mathematical material, but very often it is difficult or impossible to render colloquial English in these languages. Many textbooks avoid this problem by confining their discussions of "formalization" to artificial or concocted examples of English. I think it is more rewarding to face the problem; this is especially important in arriving at an account of the philosophical significance of modern logic.

I therefore treat formalization as a kind of translation; like translation from one natural language to another it is an art to be learned by practice and experience. I try to give examples in which the logical theories do not fit English very well, as well as ones in which there is a good fit. This more open approach gives a flexible and natural view of the interaction between formal and natural language which is more exciting to the student and philosophically satisfying to the teacher.

Proof techniques. An uninterpreted formal system is presented in Chapter I to introduce students to the notion of a system of logic. This seems to me the best way of showing that the grammar and rules of proof of a formal system can be defined without reference to the "meanings" of its symbols. The ideas exemplified in Chapter I will be taken up and explained in later chapters, and it is best not to linger over this material.

All other discussions of proof techniques deal with systems of *natural deduction*. These systems, modeled after F. B. Fitch's, are designed to be as

close as possible to patterns of reasoning that are commonly used. The rules of proof are justified and made plausible by referring to examples of valid arguments in English. In order to foster efficient use of these systems I have included sections devoted to the strategy of finding derivations.

Syntactic metatheory. Beginning students often find metatheory a mysterious and difficult topic. I try to make the transition to metatheory as easy as possible by discussing at length the issues and concepts related to it. Distinctions such as the one between object-language and metalanguage are explained at this point, and the metalinguistic notation used in this book is elucidated. Proofs of early metatheorems are presented in great detail, while the metatheorems themselves are carefully motivated and made to seem as natural as possible.

Semantic metatheory. The development of syntactic metatheory prepares the way for a treatment of semantics that is up to current standards of rigor. My discussion of semantic material is influenced by modern mathematical formulations of this subject and emphasizes those concepts that these formulations have shown to be most significant. In the semantics of sentence logic, for instance, I stress the concept of satisfaction rather than truth-tables.

Semantic completeness. The chapters dealing with this topic discuss equivalences between syntactic and semantic concepts and proceed to establish these equivalences as metatheorems. The metatheorems are proved using the techniques developed by L. Henkin; I have made the proofs as streamlined and simple as possible.

Internal organization

For efficient cross-referencing, each chapter is divided into sections, which are numbered with Arabic numerals. 'IX.6' refers to Section 6 of Chapter IX; if this reference were made in Chapter IX, it would simply be referred to as Section 6. Examples are numbered with lower-case Roman numerals, starting with 'i' at the beginning of each chapter. To cite the seventh example of Chapter II, for instance, I would use 'II.vii', or just 'vii' if the citation were made in Chapter II. Metatheorems and definitions are numbered starting with 'M1' and 'D1' at the beginning of each chapter.

At the end of chapters are exercises and problems. The problems are meant to provide breadth and excitement; each of them requires either originality or familiarity with some subject not explained in the book. Some of them are open-ended in that there is no agreement among logicians as to their solution. Not all the exercises are easy or routine, but all of them can be solved on the basis of what is supplied in the text.

There is an index of symbols as well as a general index; symbols are listed in the former index according to their first occurrence in the book. The reference given in this index will enable the reader to find a definition or explanation of the symbol.

Using the Book as a Text

In a two-term course, it is natural to do sentence logic (Chapters I to VII) during the first term and predicate logic (Chapters VIII to XII) during the second. Chapter XIII is best used to provide background for metatheoretic work in earlier chapters. A good plan would be to do XIII.1 to XIII.9 between Chapters IV and V, and XIII.10 to XIII.21 between Chapters X and XI. Sometimes a class will need a lecture or two on mathematical induction before they feel at home with the technique. Chapter XIV can be used in coordination with such lectures, or can be covered separately at any time.

Chapter I is meant to provide students with an idea of what an uninterpreted formal system is like. I have found that, if one begins with natural deduction systems, this point is harder to make. There is no need to spend much time on Chapter I, and certainly no need to frustrate students by requiring them to become adept at manipulating the rules of the system. The purpose of the chapter will have been served as soon as students know what the system is, i.e., as soon as they can recognize formulas and proofs. Some instructors may wish to omit this chapter entirely.

In a one-term course it will be necessary to devote less time to some material. The chapters on translation are particularly flexible in this regard. Depending on how much attention is given to examples, they can be gone through briskly or at a slower pace. The final sections of many chapters can be omitted without sacrificing continuity. This applies particularly to V.16, VI.14, VI.17 to VI.19, VII.9 to VII.10, VIII.18 to VIII.19, X.12 to X.14, and XII.7 to XII.9.

Acknowledgments

This book of mine owes everything to my teachers, Professors Alan Anderson, Nuel Belnap, Jr., and Frederic Fitch. I hope that the excitement and life they give to logic is reflected here and there in what follows. I owe

thanks to Professor Robert Meyer for his judgment on many points concerning the manuscript itself. My wife, Sally, has gone over the text carefully and made the prose readable, as well as suggesting many other improvements. Most of what is graceful in the book is her doing.

R. H. T.

Contents

Symbolic Logic

I Uninterpreted Syntax of a Logical System

1. This section has to do with some rules for constructing certain strings of symbols called *formulas*, and for manipulating these formulas to build arrays called *proofs*.

Some of you have studied formulas and proofs of this sort before; for the present, please try to forget what you have learned about them. The idea is to approach this material without any reference to what the "formulas" and "proofs" may mean; the rules for manipulating symbols are enough like the rules of games so that they can be regarded in a gamelike way, without regard to anything external.

Incidentally, this sort of "horseshoe pushing" is one way of caricaturing what logicians do. But despite this, many logicians aren't very interested in this sort of thing; what they do is much more abstract and conceptual. As we go on to more advanced topics you will be able to see the contrast between these two sorts of activities.

2. All our formulas will be strings made up of the following eight symbols.

p q r s
( ) ⊃ ~

For instance, the following are formulas.

(p ⊃ q)	((p ⊃ q) ⊃ (r ⊃ s))
~r	s
~(~(r ⊃ s) ⊃ ~~p)	~~~(q ⊃ q)
p	((p ⊃ q) ⊃ q)

But not every string of symbols is a formula; for instance, the following are not formulas.

~(~p)	)p(
p ⊃ (q)	((p ⊃ ~q) ⊃ ~q
p ⊃ q	⊃

These examples should give you an idea of how to recognize and make formulas; the idea is that any of the four letters 'p', 'q', 'r', and 's' is a formula, and that more complicated formulas can be made by applying the symbol '~' to the left of any formula, or the symbol '⊃' between any two formulas. In the latter case, notice that parentheses have to be added in the appropriate places.

Note: Here are some things to think about. How long can formulas get? How many formulas are there? What is the difference between saying there is no bound on the length of formulas and saying that there is no formula of unbounded length? These are questions that will seem familiar to those with some mathematical training, but others may have difficulty with them at first. In the latter case, you should do some thinking about these matters, and may want to read something about elementary set theory. See Chapter XIII of this book, or one of the works listed in the bibliography.

3. *Proofs* are built up from *axioms* using certain *rules of inference*. The directions for generating proofs are quite simple: any axiom may be written down at any time in a proof, and rules may be applied to any formulas already written down in a proof. Often we are interested in finding proofs *of* certain formulas; a proof is said to be a proof of the formula that is its last item.

The system S_0 with which we will be dealing for the time being has just three axioms; these are the formulas

A1. (p ⊃ (q ⊃ p))
A2. ((p ⊃ (q ⊃ r)) ⊃ ((p ⊃ q) ⊃ (p ⊃ r)))
A3. ((~p ⊃ ~q) ⊃ (q ⊃ p)).

The following columns of formulas qualify as proofs in S_0.

(p ⊃ (q ⊃ p))
(i)

(p ⊃ (q ⊃ p))
(p ⊃ (r ⊃ p))
(r ⊃ (r ⊃ r))
((~p ⊃ ~q) ⊃ (q ⊃ p))
(ii)

(p ⊃ (q ⊃ p))
(p ⊃ (p ⊃ p))
((p ⊃ (q ⊃ r)) ⊃ ((p ⊃ q) ⊃ (p ⊃ r)))
((p ⊃ (p ⊃ r)) ⊃ ((p ⊃ p) ⊃ (p ⊃ r)))
((p ⊃ (p ⊃ p)) ⊃ ((p ⊃ p) ⊃ (p ⊃ p)))
((p ⊃ p) ⊃ (p ⊃ p))
(iii)

((~p ⊃ ~q) ⊃ (q ⊃ p))
((~~p ⊃ ~q) ⊃ (q ⊃ ~p))
((~~p ⊃ ~~~p) ⊃ (~~p ⊃ ~p))
(iv)

((p ⊃ (q ⊃ r)) ⊃ ((p ⊃ q) ⊃ (p ⊃ r)))
(((~p ⊃ ~~p) ⊃ (q ⊃ r)) ⊃ (((~p ⊃ ~~p) ⊃ q) ⊃ ((~p ⊃ ~~p) ⊃ r)))
(((~p ⊃ ~~p) ⊃ (~p ⊃ r))
⊃ (((~p ⊃ ~~p) ⊃ ~p) ⊃ ((~p ⊃ ~~p) ⊃ r)))
(((~p ⊃ ~~p) ⊃ (~p ⊃ p))
⊃ (((~p ⊃ ~~p) ⊃ ~p) ⊃ ((~p ⊃ ~~p) ⊃ p)))
((~p ⊃ ~q) ⊃ (q ⊃ p))
((~p ⊃ ~~p) ⊃ (~p ⊃ p))
(((~p ⊃ ~~p) ⊃ ~p) ⊃ ((~p ⊃ ~~p) ⊃ p))
(v)

Before reading the explanation given below of the rules of proof construction, you may wish to work them out for yourself by inspection of the above examples.

4. There are just two rules of inference in the system S_0. The first, *modus ponens*, may be used in case you have already gotten two steps in a proof, one of which is the result of putting the symbol '⊃' after the other, following the '⊃' by a formula, and then enclosing the whole in parentheses. *Modus ponens* then allows you to infer the second of these formulas.

Notice how obscure and complicated the above explanation has become. It is much easier to say things of this sort by making a picture. We can diagram the rule *modus ponens* as follows.

--- (--- ⊃ ...)
―――――――――――
...

Any result of replacing the dashes in this scheme by a formula, and the dots by a (not necessarily distinct) formula will be a diagram of an instance of *modus ponens*. For example,

$$\frac{(p \supset (q \supset p)) \qquad ((p \supset (q \supset p)) \supset ((p \supset q) \supset (p \supset p)))}{((p \supset q) \supset (p \supset p))}$$

is such an instance: here, we have put in '(p ⊃ (q ⊃ p))' for the dashes, and '((p ⊃ q) ⊃ (p ⊃ p))' for the dots. The formulas '(p ⊃ (q ⊃ p))' and '((p ⊃ (q ⊃ p)) ⊃ ((p ⊃ q) ⊃ (p ⊃ p)))' are the *premisses* of the inference, and '((p ⊃ q) ⊃ (p ⊃ p))' is the *conclusion.*

Notice that this particular instance of *modus ponens* is one that might turn up in a proof, since both of its premisses are provable in S_0 (in fact, they are axioms). But even though the inference

$$\frac{p \qquad (p \supset q)}{q}$$

is not one that could occur in a proof (as we will see much later, 'p' is not a provable formula of S_0), it is still an instance of *modus ponens*.

5. You may have observed that a minute ago we ran into trouble when we tried to use plain English to state something general about the system S_0: the rule *modus ponens*. Obviously we cannot state this rule by listing all of its possible instances, since there are infinitely many of these. We have to use our language, to the best of our ability, to express general things of this sort. And in doing this we resorted above to the device of drawing a picture involving dashes and dots.

A plan that proves to be much more flexible and accurate is to use letters in place of the dashes and dots. To avoid confusion, we ought to select for this purpose symbols that don't resemble the 'p', 'q', 'r', and 's' of S_0—let's choose, say, the italic capitals '*A*', '*B*', '*C*', and '*D*'. We now can state the rule of *modus ponens* a bit more explicitly.

> If in a proof steps *A* and (*A* ⊃ *B*) have occurred, then the result of writing down *B* as another step is still a proof.

Later on, we will become more self-conscious about the *A*'s and *B*'s and will want to discuss in a systematic way methodological matters such as syntactic notation. But there is no need for this yet—all we have to remember is that *these letters occur nowhere in the system* S_0. They are letters that we use to say general things about S_0.

6. The second rule of S_0 is a rule of substitution; the rule allows you to replace 'p', 'q', 'r', or 's' *in* any formula *by* any formula. One must replace *all* occurrences of the letter substituted for in the formula on which one is working; the following is not an instance of the substitution rule.

$$\frac{(p \supset (q \supset p))}{(p \supset (q \supset (p \supset p)))}$$

Although '(p ⊃ p)' is substituted for the second occurrence of 'p' in '(p ⊃ (q ⊃ p))', it is not substituted for the first occurrence of 'p' in that formula.

We might express this aspect of the rule by saying that it is a rule of *simultaneous* substitution. But the substitutions must be done one at a time; the following also is not an instance of the substitution rule.

$$\frac{(p \supset (q \supset p))}{((p \supset p) \supset (r \supset (p \supset p)))}$$

We *can* get from '(p ⊃ (q ⊃ p))' to '((p ⊃ p) ⊃ (r ⊃ (p ⊃ p)))', but it takes two steps, as in the following proof.

(p ⊃ (q ⊃ p))
(p ⊃ (r ⊃ p))
((p ⊃ p) ⊃ (r ⊃ (p ⊃ p)))

Notice that the substitution rule does not allow substitutions for formulas other than 'p', 'q', 'r', and 's', so that inferences like the following one also are not sanctioned by this rule.

$$\frac{(p \supset ((q \supset r) \supset p))}{(p \supset (q \supset p))}$$

The rule of substitution is quite a different thing from *modus ponens*, and cannot be diagrammed in the same way. We could invent special notation for displaying substitutions, but at present would gain nothing by doing so; the rule should now be clear enough so that you can recognize instances of it, and that is enough.

7. We can now characterize explicitly the notion of a proof in the system S_0; a proof is any column of formulas (the *steps* of the proof), such that every step of the column is an axiom, or follows from two previous steps of the column by means of *modus ponens*, or follows from one previous step of the column by means of substitution.

According to this definition a column which consists of just one step, that step being an axiom, is a proof. Thus every axiom of S_0 has a proof in S_0 (or is *provable* in S_0 or a *theorem* of S_0).

To help in recognizing proofs, it is convenient to number steps and to include justifications, as in the following annotated version of example iii.

1	$(p \supset (q \supset p))$	A1
2	$(p \supset (p \supset p))$	1, subst
3	$((p \supset (q \supset r)) \supset ((p \supset q) \supset (p \supset r)))$	A2
4	$((p \supset (p \supset r)) \supset ((p \supset p) \supset (p \supset r)))$	3, subst
5	$((p \supset (p \supset p)) \supset ((p \supset p) \supset (p \supset p)))$	4, subst
6	$((p \supset p) \supset (p \supset p))$	2, 5, m p

(iii′)

Example iii′ is more than a proof; it is a proof together with auxiliary notations that make it easier to perceive as a proof.

Exercises

1. Which of the following are formulas of S_0?

   (a) r
   (b) $\sim\sim(p)$
   (c) $(p \supset (q \supset t))$
   (d) $(q \supset (r \supset)s)$
   (e) $\sim\sim p \supset (q \supset (r \supset p))$
   (f) $(((p \supset \sim\sim q) \supset \sim\sim\sim s) \supset \sim(p \supset p))$
   (g) $((((p \supset p) \supset p) \supset p) \supset p)$
   (h) $(p \supset (p \supset (p \supset p)))$

2. Write down some formulas of S_0.
3. Annotate the proofs given in examples i, ii, iii, iv, and v.
4. Write down some proofs in S_0.
5. Find proofs in S_0 for the following formulas, and annotate these proofs.

   (a) $(p \supset (r \supset (r \supset r)))$
   (b) $(((\sim\sim p \supset \sim q) \supset q) \supset ((\sim\sim p \supset \sim q) \supset \sim p))$
   (c) $(p \supset p)$
   (d) $(\sim q \supset (q \supset p))$
   (e) $((p \supset q) \supset ((r \supset p) \supset (r \supset q)))$

Problems

1. Find a definition of 'formula of S_0'. Make it as rigorous as possible. (*Hint:* Consult the definition given in V.3, if necessary.)
2. Use your definition to show that if a string S of symbols follows from another string T of symbols by the rule of substitution, then S is a formula of S_0 if T is.
3. Try to construct a definition of 'theorem of S_0' like your definition of 'formula of S_0'. This definition should not involve the notion of a proof explicitly.

II Implication and Negation: Informal Semantics

1. Let's begin to make some sense of S_0. At this stage our interpretation of the system will be pretty crude and will not help us much in working with the system; but at least it will begin to give us an inkling of what the system has to do with the language and reasoning we use.

The letters 'p', 'q', 'r', and 's' are to be thought of as playing the grammatical role of declarative sentences, e.g., 'People who live in glass houses ought not to throw stones' or 'Queen Victoria died in 1066'. And when we translate natural languages into formulas of S_0, these letters *stand for* such sentences; for this reason, symbols such as 'p', 'q', 'r', and 's' are often called *sentence variables* or *sentence parameters*.

The symbols '$\sim$' and '$\supset$', on the other hand, play the grammatical role of conjunctions; they do not stand for declarative sentences, but serve to form compound sentences out of simpler sentences. We will reserve the word 'conjunction' for another purpose, and call these symbols *sentence connectives* or just *connectives*.

The connective '$\sim$' is interpreted as negation; so if 'q' stands for 'Queen Victoria died in 1066', then '$\sim$q' stands for 'Queen Victoria did not die in 1066'. The symbol '$\supset$' is a conditional or implicative connective; if 'r' stands for 'I'll eat my hat' and 'q' is read as before, then '$q \supset r$' is translated 'If Queen Victoria died in 1066 then I'll eat my hat'. Finally, the parentheses make for unambiguous groupings of symbols in formulas; we wouldn't be able to distinguish '$\sim(p \supset q)$' from '$\sim p \supset q$' without parentheses.

2. Armed with this interpretation of S_0, we can do several things. First, we can make some sense of formulas, and can begin to see how the axioms and rules of inference of S_0 can be justified as axioms and rules about some subject matter. Maybe we will decide that these are good axioms and rules for negation and implication and maybe that they are bad; but in any case we are no longer looking at S_0 as a game. The system has acquired a purpose and a meaning. It may even be that this new sense of meaningfulness will serve to illuminate the exercises of Chapter I. For instance, our interpretation of the connectives may suggest new formulas that ought to be theorems, and help to devise strategies for finding proofs of them.

Second, we can translate at least some sentences of natural language into S_0. For instance,

(*i*) If Sam is not sick, then if he is not at school he is playing hooky

might be translated

(*ii*) $(\sim p \supset (\sim q \supset r))$.

As with more familiar cases of translation, there is no unique way of rendering such English sentences in S_0; but we can say in certain cases that one translation is better than another. There isn't much difference between 'Can you speak German?' and 'Do you speak German?' as a translation of 'Sprechen Sie Deutsch?' But there is no doubt that both of these are better than 'Can she speak German?' It is the same way with S_0. The formula

(*ii′*) $(\sim q \supset (\sim p \supset s))$

is as good a translation as ii of i, and both translations are better than

(*iii*) $(q \supset (\sim p \supset s))$

because they capture more of the structure of the English sentence. Although iii is a *sound* translation into S_0 of i, it isn't a *complete* translation, since it neglects the fact that 'Sam is not sick' is expressed negatively in i. On the other hand, iii would be a good translation of 'If Sam is well, then if he is not at school he is playing hooky'.

Let's look at some more examples. Other sound translations of i—none of them as good as ii—are

(*iv*) $(q \supset (p \supset s))$

(*v*) $(\sim q \supset r)$

and

(*vi*) $(p \supset q)$.

Even the formula

(*vii*) p

is a sound translation of i into S_0, although a very uninformative one. It tells us absolutely nothing about the structure of the English sentence i, since formulas such as 'p' can stand for *any* indicative sentence.

One mark, then, of a good translation into S_0 is that only those English sentences that cannot be analyzed as negations and implications are translated into the atomic formulas 'p', 'q', 'r', and 's'. Notice that this is true of translation ii, where 'p' stands for 'Sam is sick', 'q' for 'Sam is at school', and 'r' for 'Sam is playing hooky'.

(We have put 'Sam' for 'he' here, ignoring the difference between a proper name and a relative pronoun. But for the time being, let's not worry about this. We'll be able to say something about this difference later, when we get to predicate logic.)

Finally, there are many unsound or incorrect ways of translating i into S_0. For example,

(*viii*) $\sim p$

and

(*ix*) $((q \supset q) \supset (q \supset q))$

would simply be mistaken as renderings of i; they differ from translations like iv in the way a falsehood differs from an understatement. Rather than failing to represent structure that *is* in i, they represent structure that *is not* in that sentence.

In short, one can err in translating into S_0 by saying too little, or by saying too much. The idea is to get it just right.

3. Up to this point we have managed to keep a major complication at arm's length: in practice it isn't as easy to spot negations and implications in

English as we have made out. Let's start out with some simple examples, which show that implication is not always expressed in English by the locution 'if . . . then ---'.

(*x*) White will win if Black isn't careful.

(*xi*) White will win only if Black isn't careful.

If we let 'p' stand for 'White will win' and 'q' for 'Black is careful', x would be translated '$(\sim q \supset p)$' and xi '$(p \supset \sim q)$'; yet neither x nor xi uses the 'if . . . then ---' construction. And there are many other forms of English that ought also to be translated by means of the connective '$\supset$'; subjunctive constructions such as

(*xii*) We will stop by to see you, should we be in New York this summer

are sometimes used in this way. You may find it interesting to try to find other English phrases that serve this purpose.

Much the same thing happens with negation. Here we have forms such as

(*xiii*) John doesn't drink.

(*xiv*) Alice is no slouch.

(*xv*) It is by no means the case that familiarity breeds contempt.

It also seems reasonable to translate

(*xvi*) Sally is atypical

as '$\sim$p', letting 'p' stand for 'Sally is typical'. But in many such cases we must watch our step: 'Bill is unhappy' doesn't seem to mean quite the same thing as 'Bill is not happy'. (Bill could fail to be happy, and yet not be unhappy, either.)

Now, let's take stock. In Section 2 we talked about the soundness and completeness of translations of English sentences into S_0. In this discussion, translation seemed a pretty cut-and-dried affair; it was as if the English sentence came neatly packaged in some logical form or other, and good translation was just a matter of fitting this form. Now it appears that it isn't always so simple; at least, there are no unique English constructions that correspond to the connectives '$\supset$' and '$\sim$' of S_0.

But things are more complicated even than this; not only are there many ways of expressing implication in English, but often English constructions which seem to express implication do not in fact do so. A simple example is

(*xvii*) You can have some cake if you like

or, even better,

(*xviii*) You will fasten your safety belts, if you please.

In xviii the 'if' functions as part of a polite formula which has little or no connection with the implication of S_0. One way of seeing this is to consider the sheer inanity of the following argument.

(*xix*) You will fasten your safety belts, if you please.
You please.
Therefore you will fasten your safety belts.

Examples of another sort are

(*xx*) If butter is warmed, it melts

and

(*xxi*) If anyone is sick, he should report to the doctor.

You should feel that something is a bit odd about both of the following arguments—particularly, the second.

(*xxii*) If butter is warmed, it melts.
Butter is warmed.
Therefore butter melts.

(*xxiii*) If anyone is sick, he should report to the doctor.
Anyone is sick.
Therefore he should report to the doctor.

(We will not really be able to sort these two examples out until we get to Chapter VIII. But one way of preparing xx for translation into logical notation is to render it as 'Whenever any piece of butter is warmed, it melts'. This can then be treated as a statement of universality—a universally quantified sentence—rather than as one of implication. Example xxi would similarly be treated as a statement of universality.)

Finally, there are a number of difficulties concerning the logical translation of subjunctive and so-called "contrary-to-fact" conditionals. Again, we can use the *modus ponens* test to indicate that something is fishy about examples such as

(*xxiv*) If Cromwell had not been born, Charles I would not have been executed.

We don't use xxiv to infer 'Charles I would not have been executed' from 'Cromwell had not been born'; this isn't even grammatical English. Even tinkering with the verbs doesn't help much; 'If Cromwell was not born then Charles I was not executed' seems very different from xxiv. Still another difficulty is that A3 of S_0 does not seem to apply to this example. The sentence 'If Charles I had been executed, then Cromwell would have been born' is not exactly a welcome consequence of xxiv, as it should be if this example involved the sort of implication at stake in S_0.

Probably this subject has now been belabored enough. The point is, there are no simple recipes for translating natural language into logical systems. As in more familiar cases of translation one has gradually to become bilingual—to get one's ear in—and to acquire the knack of good translating through diligent practice. (Incidentally, this makes the exercises on translation in this and later chapters rather important.)

4. Systems such as S_0 are often called *formal* systems and the process of translating natural languages into them is often called *formalization*. The idea behind this is that there are certain forms which sort out what is relevant to valid or logically correct inference. These logical types are captured by the grammar of the formal system.

For instance, take the following two inferences.

If Jones is an honest man, he will not be arrested by the police.
Jones will be arrested by the police.
Therefore Jones is not an honest man.

If space is four-dimensional, it is not three-dimensional.
Space is three-dimensional.
Therefore space is not four-dimensional.

Inspection of these inferences *with regard only to* correctness of the reasoning involved gives a very forcible impression that the one inference is logically correct if and only if the other is. And this can be confirmed and explained by using the formal system S_0; we can say that these arguments share the logical form

$$\frac{\begin{array}{l}(p \supset \sim q)\\ q\end{array}}{\sim p}.$$

Notice that in placing these arguments in the same category we are ignoring many features such as the truth or falsity of their component sentences or their moral and emotional overtones that people usually find more interesting than logical form. Sometimes people find it difficult to carry out the feat of abstraction needed to view texts from the standpoint of logic; it is for this

reason that E. Beth has said that "undue moral rectitude" is a handicap in the study of logic. One of the things a course in logic can do for people is to develop an ability to look at specimens of reasoning in this disinterested way; this can be a handy thing if your line of work involves the generation or evaluation of good arguments. Fortunately, this ability does not seem to detract from the appreciation of other things in language and life.

Exercises

1. Translate the following into S_0.

(a) We'll go to the beach today, provided it doesn't rain.
(b) If you do what you're told you won't get along badly here.
(c) The dog was not treated unkindly.
(d) If he castles if I move my pawn, then if I don't lose my queen I should be able to beat him.
(e) All mice are mortal.
(f) If I miss my train I can arrive only five minutes late, assuming the next train is on time.
(g) Sam isn't over five feet tall, unless he has grown.

2. Translate the following inferences into S_0. Which of them strike you as logically valid?

(a) If today is Monday then tomorrow isn't Wednesday.
Therefore if tomorrow is Wednesday then today is not Monday.
(b) It is not the case that man does not live by bread alone.
Therefore man lives by bread alone.
(c) If murder is wrong, then attempted murder is wrong.
If attempted murder is wrong, then the very thought of murder is wrong.
Therefore if murder is wrong, then the very thought of murder is wrong.
(d) If God is dead then everything is permitted.
God is not dead.
Therefore it is not the case that everything is permitted.
(e) It is not the case that if God is dead then everything is permitted.
Therefore God is dead.
(f) If roses are red if violets are blue, then violets are blue.
Therefore violets are blue.

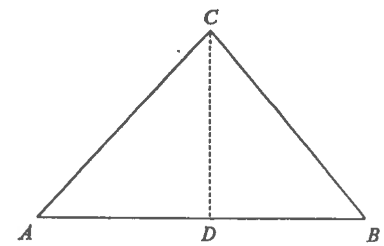
III Implication Logic: Natural Deduction Techniques

1. You couldn't have helped noticing that it was not an easy matter to find proofs in S_0. The source of this trouble with S_0 is that the system does not reproduce very accurately the way in which people in fact argue.

The aim of this chapter is to remedy this by formulating a system of logic which is more easy to work with. This system, $S_{\supset}$, is one of a family of systems of so-called *natural deduction* which have been designed with an eye to the "nature" of ordinary reasoning.

2. Let's begin by considering some specimen arguments which people use. Since we want these arguments to be as good and rigorous as possible, our examples are drawn from mathematics.

Problem: to show that if triangle ABC has side AC equal to side BC, then angles CBA and CAB are equal.



PROOF
1. Suppose that triangle ABC has side AC equal to side BC.
2. Let D be the midpoint of AB, and draw the line CD.
3. Since AC equals BC, DA equals DB, and CD equals CD, triangles ACD and BCD are congruent.
4. Corresponding angles of congruent triangles are equal.
5. Angles CBA and CAB are equal. Q.E.D.

(*i*)

If we consider only the features of this argument that have to do with implication, our attention is directed to the statement of the problem and to steps 1 and 5 of the proof. Translating these into S_0, we find that in argument i a sentence (namely, 'If triangle ABC has side AC equal to side BC, then angles CBA and CAB are equal') having the form '$(p \supset q)$' is proved by *supposing* 'p' and then *deducing* 'q'.

There is no doubt that we have discovered here a very important characteristic of the way we argue. A glance at any systematic body of reasoning in disciplines such as mathematics, the natural sciences, or philosophy will reveal a high density of words such as 'suppose', 'let', and 'assume' which signal that a hypothesis has been made.

We also reason hypothetically in more everyday situations. For instance, if we are considering which of two jobs to accept, we imagine that we have taken the one job and try to infer what will happen, and do likewise with the other job. Then we try to compare the consequences.

3. Our first task, then, is to devise a way in which supposition can be handled within some formal system. Clearly we want some notation that will distinguish *supposition* from *assertion* since, for instance, to *assert* that there will be war with China and to *suppose* that there will be war with China are entirely different things. Also we will want to be able to keep track of which sentences are inferred from which suppositions, since, on the one hand, to say that there will be prosperity under the hypothesis that a Democrat will

be elected and, on the other, to say that there will be prosperity under the hypothesis that a Republican will be elected are different things.

We will use for this purpose a format originated by Frederic B. Fitch, which uses arrays of vertical lines and horizontal strokes. The sentences above the horizontal strokes are *hypotheses* (or suppositions), and those below the horizontal stroke and to the right of the vertical line to which the stroke is attached are inferred under these hypotheses.

Example i, when written in this format, looks like this.

| Triangle ABC has side AC equal to side BC.
|—
| Let D be the midpoint of AB, and draw the line CD.
| Since AC equals BC, DA equals DB, and CD equals CD, triangles ACD and BCD are congruent.
| Corresponding angles of congruent triangles are equal.
| Angles CBA and CAB are equal.

(ii)

The first step, being above the stroke, is a hypothesis; the succeeding steps are made under this hypothesis.

Since example i was a proof of the implication 'If triangle ABC has side AC equal to side BC, then angles CBA and CAB are equal', we want this implication to follow from the argument ii as a whole. Note, however, that this implicative conclusion is *not* subject to the hypothesis 'Triangle ABC has side AC equal to side BC'. It is *categorical*; i.e., it is subject to *no* hypotheses. We can represent this by placing it outside the vertical line, as follows.

| | Triangle ABC has side AC equal to side BC.
| |—
| | Angles CBA and CAB are equal.
| If triangle ABC has side AC equal to side BC, then angles CBA and CAB are equal.

(iii)

4. We can't at present do much more with example i, since the inner steps of the argument have to do with principles of geometry rather than with principles valid in logic. So instead of pursuing this matter further, let's discuss the notion of a hypothetical derivation using other examples.

You might think that all of the entries in such a derivation would be sentences, as in example i, but a little reflection shows that this is not general enough. We may want to have derivations *within* derivations.

Suppose, for instance, that we want to prove something like 'If n is a prime number, then if $n \neq 2$ then n is odd'. We would begin by assuming that n is a prime number (i.e., a number like 7, which is evenly divisible only by 1 and itself). Our next step, however, would not be an inference from this hypothesis, but would be a new hypothesis. We will then have to separate these hypotheses by drawing another vertical line, thus.

| n is a prime number.
|—
| | $n \neq 2$.
| |—

(iv)

It will be instructive to complete this argument; the reasoning is as follows.

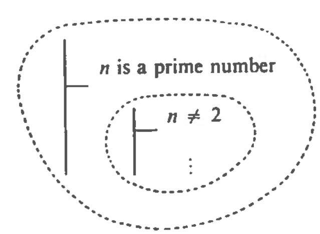
1 | n is a prime number
|—
2 | | $n \neq 2$
| |—
3 | | All prime numbers are evenly divisible only by 1 and themselves.
4 | | n is a prime number.
5 | | So n is evenly divisible only by 1 and itself.
6 | | Therefore n is evenly divisible by 2 only if $n = 2$.
7 | | But $n \neq 2$.
8 | | So n is not evenly divisible by 2.
9 | | All numbers are odd if and only if they are not evenly divisible by 2.
10 | | n is odd if and only if n is not evenly divisible by 2.
11 | | n is odd.

(v)

The steps in this example are numbered for the sake of convenience in referring to them.

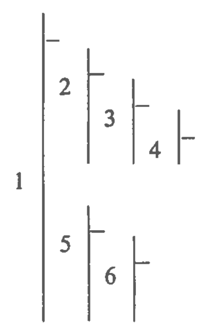
No mathematician would dream of writing this argument out in such tedious detail, but example v, by virtue of its very tediousness, illustrates some points that will prove useful. It is very nearly a logically complete argument, although it will be a while until we have enough formal horsepower available to deal with all of the inferences used in it.

Example v divides naturally into two units, as follows.



These units will be called *derivations*. (Fitch calls them "subproofs", but I will try to reserve the term 'proof' for proofs such as those in S_0.) As the diagram shows, one of these derivations is a subunit of the other (in fact, the outermost derivation consists of two items, its hypothesis and the second derivation). We will express this by saying that the second derivation is *subordinate* to the first.

Having allowed this much subordination, we have no reason to stop anywhere, and no amount of complexity of subordination can be excluded. For instance, we must be prepared to accept derivations organized like the following one.



(*vi*)

Here (for the first and only time), the numbers serve to label derivations. We will understand 'subordination' in a cumulative sense, so that in example vi derivations 2, 3, 4, 5, and 6 are subordinate to derivation 1, derivations 3 and 4 are subordinate to derivation 2, derivation 4 is subordinate to derivation 3, and derivation 6 is subordinate to derivation 5. Derivation 5, however, is *not* subordinate to derivation 2.

We have still another thing to learn from example v. Notice that at step 4, we repeated the hypothesis of the first derivation. We did this because it was required at this point for the argument; we would not have been able to get the desired conclusion, that n is odd, without using somewhere the assumption that n is prime.

Our restating at step 4 of the hypothesis that n is prime may have seemed gratuitous. But it was more than just a matter of stuttering since, after all, at this stage we are working in a different derivation which does *not* have this hypothesis. We will say that moves of this sort are justified by a rule of *reiteration*, which enables us to make use of hypotheses that are in force through subordinations.

In general, the rule even allows the reiteration of steps that are not hypotheses: stated accurately, the rule is as follows.

If one derivation D_1 is subordinate to another, D_2, then any step in D_2 which appears above the hypothesis of D_1 can be reiterated into D_1.

In other words, anything above and to the left can be reiterated. Many uses of this rule will be found below in this chapter.

Having dragged so much information out of example v, it is only fair to complete this argument. Where we left it, we had gotten to 'n is odd' from the hypothesis '$n \neq 2$'; this entitles us to conclude that if $n \neq 2$, then n is odd.

| n is a prime number.
| | $n \neq 2$.
| | ⋮
| | n is odd.
| If $n \neq 2$, then n is odd.

(*vii*)

Again, our conclusion does not depend on the second hypothesis, that $n \neq 2$. We have *discharged* this hypothesis in inferring the implication, and represent this by placing this step to the left of the derivation which proceeds from that hypothesis. At this point, the derivation has terminated.

But our conclusion is still subject to a hypothesis: the one of the outermost derivation. We can go on to discharge this hypothesis as well, if we wish.

| | n is a prime number.
| | | $n \neq 2$
| | | ⋮
| | | n is odd.
| | If $n \neq 2$, then n is odd.
| If n is a prime number, then if $n \neq 2$, then n is odd.

(*viii*)

This last conclusion is dependent on no hypotheses at all; it is *categorical*. (In such cases try to be careful to draw a line to the left of the whole derivation, with no hypotheses-strokes. This makes the whole thing look a bit neater.)

5. Up to this point our discussion of things like hypotheses, derivations, subordination, reiteration, and categoricity has provided a general structural framework on which arguments can be hung. We have also suggested a rule for deriving implications.

It is high time now to put these ideas to work within the context of some formal system. For this purpose we'll work at first with a class of formulas that in a way is more general, and in a way more restricted than the class of formulas of S_0. We will allow more (infinitely more!) sentence parameters than the four of S_0 but will not, for the time being, allow the use of negation.

The formulas of $S_\supset$, then, are defined as follows.

1. Any of the sentence parameters

$$\begin{matrix} p & q & r & s & \ldots \\ p_1 & q_1 & r_1 & s_1 & \ldots \\ p_2 & q_2 & r_2 & s_2 & \ldots \\ & \vdots & & & \end{matrix}$$

is a formula of $S_\supset$;

2. If A and B are formulas of $S_\supset$, then so is $(A \supset B)$.

A string of symbols qualifies as a formula of $S_\supset$ only if it can be shown to be a formula by repeated applications of 1 and 2.

Derivations in $S_\supset$ will use the format we developed in the previous section, the steps in these derivations being formulas rather than English sentences. Knowing only this much, we can form the following derivations in $S_\supset$.

```
|_p                |_(p ⊃ q)
|  |_(q ⊃ r)       |  |_p
|  | p             |  |  |_(q ⊃ r)
   (ix)            |  |  | p
                   |  |  | (p ⊃ q)
                         (x)
```

But we aren't going to get very far this way, just making hypotheses and using the reiteration rule. We need additional rules in order to really do something with the system. We don't have to look very far for such rules; we already used one likely candidate in examples iii and viii. This rule, you remember, allowed us to infer an implication (like '(p ⊃ q)') from a derivation of its consequent ('q', in this case) from its antecedent ('p'). Schematically, we can picture the rule as follows.

```
|  |_A
|  | ⋮
|  | B
| (A ⊃ B)
```


This diagram means that whenever in a derivation we have completed a pattern looking like

```
|  |_A
|  | ⋮
|  | B
```

(so that '*A*' and '*B*' will be replaced by particular formulas, and the dots by an argument using rules of $S_\supset$), we can go on to infer $(A \supset B)$ as shown.

We will call this rule the rule of *implication introduction*, since it allows us to infer *to* an implication; it shows us how to introduce implications in an argument by means of reasoning.

The rule of implication introduction can be combined with the reiteration rule to produce more complicated and interesting derivations, such as these.

```
|_(p ⊃ q)             |  |_p
|  |_p                |  |  |_q
|  | (p ⊃ q)          |  |  | p
| (p ⊃ (p ⊃ q))       |  | (q ⊃ p)
    (xi)              | (p ⊃ (q ⊃ p))
                          (xii)
```

We can also use the rule of implication introduction to derive formulas such as '(p ⊃ p)' categorically; this can be done in the following way.

```
|  |_p
|  |
| (p ⊃ p)
   (xiii)
```

In a way, this is a lazy application of implication introduction, since nothing at all is derived from the hypothesis 'p' before it is discharged; but nevertheless, we will regard examples such as this as correct uses of implication introduction.

6. There is just one more thing that needs to be added in order to complete the system $S_\supset$. There are various ways to see what is needed; one way is to try to see what would be required to derive '((p ⊃ (q ⊃ r)) ⊃ ((p ⊃ q) ⊃ (p ⊃ r)))' categorically.

But general considerations of symmetry also suggest that $S_\supset$ should have a rule of *implication elimination* to complement its rule of implication introduction. After all, we need to argue *from* implications as well as *to* them.

Again there is little doubt about what this rule ought to be; the rule of *modus ponens* is the most plausible candidate. Putting this suggestion into practice, let's agree that applications of the rule of implication elimination (or *modus ponens*) are permitted in the system $S_{\supset}$ as follows.

```
| A              | A ⊃ B
| ⋮              | ⋮
| A ⊃ B          | A
| ⋮              | ⋮
| B              | B
```

That is, any time when we have both formulas A and $A \supset B$ in a derivation, we may infer B by the rule of implication elimination.

In the following examples all the rules of $S_{\supset}$ are employed.

```
| |_(p ⊃ (q ⊃ r))
| |  |_(p ⊃ q)
| |  |  |_p
| |  |  |  (p ⊃ q)
| |  |  |  q
| |  |  |  (p ⊃ (q ⊃ r))
| |  |  |  (q ⊃ r)
| |  |  |  r
| |  |  (p ⊃ r)
| |  ((p ⊃ q) ⊃ (p ⊃ r))
| ((p ⊃ (q ⊃ r)) ⊃ ((p ⊃ q) ⊃ (p ⊃ r)))
```

(*xiv*)

```
|_((p ⊃ q) ⊃ r)
|  |_q
|  |  |_p
|  |  |  q
|  |  (p ⊃ q)
|  |  ((p ⊃ q) ⊃ r)
|  |  r
|  (q ⊃ r)
```

(*xv*)

As in the case of S_0, it often is helpful to annotate derivations, showing the reasons for steps. This will become particularly useful later on when we will

be working with more rules. Example xv, if annotated in this way, looks like this.

```
1  |_((p ⊃ q) ⊃ r)         hyp
2  |  |_q                  hyp
3  |  |  |_p               hyp
4  |  |  |  q              2, reit
5  |  |  (p ⊃ q)           3–4, imp int
6  |  |  ((p ⊃ q) ⊃ r)     1, reit
7  |  |  r                 5, 6, m p
8  |  (q ⊃ r)              2–7, imp int
```

(*xv′*)

7. For some purposes it is convenient to have derivations with multiple hypotheses. (One reason for this is that in everyday cases we frequently want to argue from many hypotheses—in no particular order of subordination—down to a conclusion.) It is easy to allow such derivations in $S_{\supset}$. Here is an example, closely related to example xiv above.

```
1  | (p ⊃ (q ⊃ r))         hyp
2  |_(p ⊃ q)               hyp
3  |  |_p                  hyp
4  |  |  (p ⊃ q)           2, reit
5  |  |  q                 3, 4, m p
6  |  |  (p ⊃ (q ⊃ r))     1, reit
7  |  |  (q ⊃ r)           3, 6, m p
8  |  |  r                 5, 7, m p
9  |  (p ⊃ r)              3–8, imp int
```

(*xvi*)

8. There are several mistakes that beginners often make in trying to find derivations in systems such as $S_{\supset}$. One of the most common misunderstandings leading to mistakes of this kind has to do with subordination.

For instance, suppose we are set the task of deriving 'r' from 'q' and '((p ⊃ q) ⊃ r)'. In the course of solving this problem we may be forced to make hypotheses; and since we can make any hypothesis we like, why not assume '(p ⊃ q)'? This will enable us to get to 'r' as follows.

```
1  | q                     hyp
2  |_((p ⊃ q) ⊃ r)         hyp
3  |  |_(p ⊃ q)            hyp
4  |  |  ((p ⊃ q) ⊃ r)     2, reit
5  |  |  r                 3, 4, m p
```

(*xvii*)

The trouble with xvii is that it isn't a solution to the problem. Its last step, 'r', is indeed the conclusion we wanted; but in this step 'r' has not been obtained as an item of the main derivation; it is an item of a subordinate derivation in which a further hypothesis is made. In step 5, 'r' is subject not only to the hypotheses 'q' and '((p ⊃ q) ⊃ r)', but also to the hypothesis '(p ⊃ q)'.

To see just how mistaken this technique is, it's only necessary to consider an extreme example of it like the following "derivation" of 'q' from 'p'.

```
|_p
|  |_q
|  |
```

(xviii)

This example is the most blatant possible case of the fallacy known as *petitio principii*: assuming what was to be proved. Example xvii is a more disguised version of the same fallacy. Using this method, it would be easy to "derive" anything from anything.

What is asked, then, by a request to derive *B* from hypotheses $A_1, \ldots, A_n$ is that *B* should be obtained as an item in a derivation headed by the hypotheses $A_1, \ldots, A_n$—not as an item in a derivation in which other hypotheses are in force as well. To derive *B* from $A_1, \ldots, A_n$ means to derive *B* from $A_1, \ldots, A_n$ *only*.

On the other hand, this doesn't mean that other hypotheses can't be made in carrying out a derivation. It means only that these hypotheses must be *discharged* in the course of the argument.

Consider, for instance, our original problem: to derive 'r' from 'q' and '((p ⊃ q) ⊃ r)'. The insight needed here is that we could obtain 'r' by *modus ponens* if we could get '(p ⊃ q)'. We are then led to attempt a derivation along the following lines.

```
| q
|_((p ⊃ q) ⊃ r)
|  ?
| (p ⊃ q)
| r
```

(xix)

In a situation like this, when you're trying to argue to an implication, it is never a bad idea to try to get it by implication introduction. So we set up a

subordinate derivation with 'p' as hypothesis, and try to obtain 'q' in this new derivation.

```
| q
|_((p ⊃ q) ⊃ r)
|  |_p
|  |  ?
|  | q
| (p ⊃ q)
| r
```

(xx)

The last step of this subordinate derivation isn't an implication, so we can't get it by implication introduction. Therefore we look up at the formulas that can be reiterated into the derivation, to see whether we can use them to get 'q'. In this case, it's easy; 'q' *is* one of these formulas, and we can end the derivation as it stands. The end product looks like this.

```
1  | q                  hyp
2  |_((p ⊃ q) ⊃ r)      hyp
3  |  |_p               hyp
4  |  | q               1, reit
5  | (p ⊃ q)            3–4, imp int
6  | r                  2, 5, m p
```

(xxi)

Notice that although another hypothesis has been made in xxi, it has been discharged by step 6. In this step 'r' is subject to only the hypotheses 'q' and '((p ⊃ q) ⊃ r)', as is shown by its not being separated by any lines from the outermost line of the derivation.

Another misapprehension that sometimes causes trouble is that hypotheses made in a derivation must be *used* in getting to the conclusion. So far is this from being true that very often we have to make hypotheses that are quite irrelevant to the conclusion. This occurs, for instance, in trying to find a categorical derivation of '(p ⊃ (q ⊃ q))'.

In solving this problem, we should follow the policy of getting implications by implication introduction and begin as follows.

```
|  |_p
|  |  ?
|  | (q ⊃ q)
| (p ⊃ (q ⊃ q))
```

(xxii)

The danger here is to assume that '(q ⊃ q)' must come by somehow using the hypothesis 'p'. If you try to do this you'll get stuck.

The right way to proceed in cases such as this is to rely mechanically on the advice that implications should be obtained by implication introduction; we then form a subordinate derivation with 'q' as hypothesis.

```
| |_p
| |  |_q
| |  |  ?
| | (q ⊃ q)
| (p ⊃ (q ⊃ q))
```

(xxiii)

Since, as we explained in Section 5, we regard steps such as

```
| |_q
| |
| (q ⊃ q)
```

as instances of implication introduction, we need proceed no further; to finish the problem it's only necessary to fill in reasons.

```
1  | |_p                 hyp
2  | |  |_q              hyp
   | |  |
3  | | (q ⊃ q)           2, imp int
4  | (p ⊃ (q ⊃ q))       1–3, imp int
```

(xxiv)

This is a perfectly correct derivation, even though no use whatsoever was made of the hypothesis 'p' in deriving '(q ⊃ q)'.

Exercises

1. Find derivations of the following in $\mathbf{S}_\supset$.
   (a) (p ⊃ q) from (r ⊃ q) and (p ⊃ r)
   (b) (r ⊃ (p ⊃ (q ⊃ r))) from (p ⊃ (r ⊃ (q ⊃ r)))
   (c) (p ⊃ (p ⊃ p)) from q
   (d) (p ⊃ p) from (q ⊃ r)
   (e) r from ((p ⊃ (q ⊃ p)) ⊃ r)
   (f) ((r ⊃ q) ⊃ q) from ((p ⊃ q) ⊃ q) and (p ⊃ r)
   (g) ((p ⊃ r) ⊃ r) from ((p ⊃ q) ⊃ q), (q ⊃ r), and (r ⊃ q)
   (h) r from p and (((q ⊃ q) ⊃ p) ⊃ r)

   (i) ((p ⊃ q) ⊃ (r ⊃ r)) from ((p ⊃ q) ⊃ (r ⊃ q))
   (j) p from ((p ⊃ q) ⊃ p), (p ⊃ (r ⊃ q)), and r
   (k) (r ⊃ p) from ((p ⊃ r) ⊃ ((q ⊃ r) ⊃ (r ⊃ p)))
   (l) (p ⊃ q) from (((r ⊃ q) ⊃ (q ⊃ p)) ⊃ (p ⊃ q))

2. Find categorical derivations of the following in $\mathbf{S}_\supset$.
   (a) ((p ⊃ q) ⊃ ((q ⊃ r) ⊃ (p ⊃ r)))
   (b) (r ⊃ ((p ⊃ q) ⊃ (s ⊃ (p ⊃ q))))
   (c) (((p ⊃ p) ⊃ p) ⊃ p)
   (d) ((p ⊃ (p ⊃ q)) ⊃ (p ⊃ q))
   (e) (((p ⊃ (q ⊃ r)) ⊃ s) ⊃ (r ⊃ s))
   (f) (((p ⊃ p) ⊃ q) ⊃ q)
   (g) ((p ⊃ q) ⊃ (((r ⊃ q) ⊃ s) ⊃ ((r ⊃ p) ⊃ s)))

Problems

1. Show that if there is a categorical derivation of a formula *A* in $\mathbf{S}_\supset$ then there is a categorical derivation of (*B* ⊃ *A*) in $\mathbf{S}_\supset$, where *B* is any formula of $\mathbf{S}_\supset$.
2. Show that, if there is a categorical derivation of a formula *A* in $\mathbf{S}_\supset$, then there is a categorical derivation of any substitution instance of *A*.
3. Is there a categorical derivation in $\mathbf{S}_\supset$ of '(((p ⊃ q) ⊃ p) ⊃ p)'? If you claim there is none, can you devise any clearcut way of demonstrating that your claim is true?
4. In the next chapter, we are going to add rules for negation ('~'), conjunction ('∧'), disjunction ('∨'), and equivalence ('≡'). Figure out what these rules must be.

IV Sentence Logic: Informal Semantics and Natural Deduction Techniques

1. In the last chapter we constructed a framework, consisting of the apparatus of subordination and reiteration, in which hypothetical reasoning could be displayed. Then we devised a formal system accounting for implication by adding rules to this underlying framework. Just two rules were added, one for introducing the connective '$\supset$' and one for eliminating it.

One nice thing about natural deduction is that these features recur in very tidy and systematic fashion as we expand our logical horizons. If we wish to add more connectives we only have to find out the right pair of rules for the connective; this is almost always an easy thing to do, requiring only a little sensitivity to the way we reason. Once we have got the right pair of rules, we know at once how to incorporate the connective into the logical system.

In this chapter we will be able to dispose quickly of negation and a number of other connectives as well: conjunction, disjunction, and equivalence. By the time we are through, we will have a system of logic that can handle many new sorts of reasoning.

2. Of course, besides adding new rules, we will have to add new formulas as well. A rule for '$\sim$' wouldn't be of much use in a system unless the system allowed formulas involving the connective '$\sim$'. So in all, we will be adding four connectives to the system $S_{\supset}$ of the last chapter: '$\wedge$', '$\vee$', '$\sim$', and '$\equiv$'. Each time we add one of these connectives we will extend the class of formulas with which we are working, and in fact, by the time we finish this chapter, we will have discussed a number of logical systems: $S_{\supset\wedge}$, $S_{\supset\wedge\vee}$, $S_{\supset\wedge\vee\sim}$, and $S_{\supset\wedge\vee\sim\equiv}$.

But it would just be too fussy to have to keep track of things in this way. Rather than discussing at each stage the additions we are making to the set of formulas and giving each of these a name, we will simply regard them as part of a big system. Then we can describe the formulas of that system and be done with it.

The formulas of this inclusive system, S_s, are built up according to the following rules.

1. Any of the sentence parameters of $S_{\supset}$ (see III.5) is a formula of S_s;
2. If A and B are formulas of S_s, then so is $(A \supset B)$;
3. If A is a formula of S_s, then so is $\sim A$;
4. If A and B are formulas of S_s, then so is $(A \wedge B)$;
5. If A and B are formulas of S_s, then so is $(A \vee B)$;
6. If A and B are formulas of S_s, then so is $(A \equiv B)$.

Again, a string of symbols qualifies as a formula of S_s only if it can be shown to be a formula by repeated applications of rules 1 to 6.

Mixing up all these rules at once can lead to the construction of some pretty complicated formulas. The following are examples of formulas of S_s.

$$((p \supset ((q \wedge r) \equiv \sim(s \vee (p \wedge q)))) \vee (r_5 \supset (s \wedge p)))$$
$$((((q \wedge q) \wedge q) \wedge q) \vee p)$$
$$((\sim\sim p_2 \vee \sim\sim q_1) \equiv (p \wedge q_5))$$
$$(\sim p \supset ((q \wedge q) \vee \sim(r \wedge s)))$$

In the sections below we will deal in turn with each of these new connectives. The plan will be (1) to discuss briefly the translation of sentences of natural language into formulas involving the connective, (2) to discuss specimen arguments using English equivalents of the connective, and (3) to use these arguments in figuring out introduction and elimination rules for the connective.

3. In this section we will consider conjunction, which means that on the formal side of things we will be dealing with the connective '∧', and on the informal side with the word 'and'. The sentence 'Brasidas was a great general, and Thucydides was a great historian' would be translated into S_s as 'p ∧ q', letting 'p' and 'q' stand for the conjuncts of the English sentence. Similarly, 'Chicago and New York are large cities' would appear as a formula like 'p ∧ q'; but here one must be sensitive to nuances. 'Jack and Jill went up the hill', or better, 'Jack and Jill got married' conveys something more than the conjunction in which we are interested for logical purposes; there is the suggestion that the acts are done together in the first case, and reciprocally in the second. 'Jack counted three and pulled the ripcord', on the other hand, suggests that the actions were performed in a certain order; this also is a feature not taken into consideration in logical conjunction.

In working out logical rules for conjunction, our problem is to decide how we reason to conjunctions in arguments, and how we reason from them. As it turns out, this is a very simple matter; if you think about it a minute you should be able to work out the answer.

Suppose, for instance, that we want to show that the square root of 5 is less than 3 and greater than 2. Well, we would have to show two things: that $\sqrt{5}$ is less than 3 and that $\sqrt{5}$ is greater than 2. (It doesn't matter in which order we do this, as long as we get them both.)

So, we might argue as follows. (To keep from digressing too far, we'll suppose that for all real numbers x and y, $x < y$ if and only if $x^2 < y^2$—a real number is less than another just in case its square is less than the other's square.)

1. $3^2 = 9$.
2. $(\sqrt{5})^2 = 5$.
3. $5 < 9$.
4. So $(\sqrt{5})^2 < 3^2$, in view of 1, 2, and 3.
5. But then $\sqrt{5} < 3$.
6. $2^2 = 4$.
7. $4 < 5$.
8. So $2^2 < (\sqrt{5})^2$, in view of 2, 6, and 7.
9. But then $2 < \sqrt{5}$.
10. Therefore, in view of 5 and 9, $\sqrt{5} < 3$ and $2 < \sqrt{5}$.

(i)

The step in which we are interested here is the last, number 10. If we trans-

late the inference used in passing from 6 and 9 to 10 into the notation of S_s, we get something like this.

$$\begin{array}{|l} \vdots \\ p \\ \vdots \\ q \\ \vdots \\ (p \wedge q) \end{array}$$

(ii)

And now it's easy to state in general the introduction rule for '∧' in S_s; we have to be careful to remember, though, that the order in which we got the premisses in i didn't matter; we could as well have proved that $2 < \sqrt{5}$ before proving that $\sqrt{5} < 3$. Now for the rule; applications of the rule of *conjunction introduction* (*conj int*) are permitted in the system S_s, as follows.

$$\begin{array}{|l} \vdots \\ A \\ \vdots \\ B \\ \vdots \\ (A \wedge B) \end{array} \qquad \begin{array}{|l} \vdots \\ B \\ \vdots \\ A \\ \vdots \\ (A \wedge B) \end{array}$$

That is, whenever we have written down two formulas A and B (in any order) in a derivation in S_s, we may then write down in that derivation the result $(A \wedge B)$ of conjoining A with B.

4. If anything, it's easier to grasp the rule of conjunction elimination, and to avoid being long-winded we may as well state it without any fanfare. This rule is applied as follows.

$$\begin{array}{|l} \vdots \\ (A \wedge B) \\ \vdots \\ A \end{array} \qquad \begin{array}{|l} \vdots \\ (A \wedge B) \\ \vdots \\ B \end{array}$$

Whenever we have written a conjunction $(A \wedge B)$ in a derivation, we may thereafter write down either of its conjuncts, A and B.

So far then, we have discussed two pairs of rules for logical connectives of the system S_s: the connectives '⊃' and '∧'. Here are some specimen derivations making use of these rules.

1	⎸⎸p	hyp
2	⎸⎸⎸q	hyp
3	⎸⎸⎸p	1, reit
4	⎸⎸⎸(p ∧ q)	2, 3, conj int
5	⎸⎸(q ⊃ (p ∧ q))	2–4, imp int
6	⎸(p ⊃ (q ⊃ (p ∧ q)))	1–5, imp int

(iii)

1	⎸(p ∧ (q ∧ r))	hyp
2	⎸p	1, conj elim
3	⎸(q ∧ r)	1, conj elim
4	⎸q	3, conj elim
5	⎸(p ∧ q)	2, 4, conj int
6	⎸r	3, conj elim
7	⎸((p ∧ q) ∧ r)	5, 6, conj int

(iv)

5. Now for disjunction, expressed often by the English 'or'. The connective '∨' of S_s corresponds to the inclusive sense of 'or', in which the alternative in which both disjuncts hold true is not meant to be excluded.

For instance, if Smith maintains that Jones is a miser or a vindictive troublemaker, we would not say that Smith was wrong in case both turn out to be true. Smith may not have supposed that both in fact were true, or he would not have made his point so guardedly; he would have said that Jones was a miser *and* a vindictive troublemaker. But still, the disjunctive claim is true under these circumstances. This inclusive sense of 'or' is sometimes expressed by 'and/or' in circumstances (for instance those in which one is formulating rules of some sort) where the speaker feels it is important to make his meaning very precise.

The English 'unless' also has an inclusive sense which would be properly translated in S_s by uses of '∨'. For instance 'Unless my memory is bad the sun is 93 million miles from the earth' could be translated by '(p ∨ q)', where 'p' stands for 'My memory is bad' and 'q' for 'The sun is 93 million miles from the earth'. In this example, the first claim would still hold good in case 'My memory is bad' and 'The sun is 93 million miles from the earth' are both true; I may have made a lucky guess.

There are difficulties with 'unless' which make it risky to translate English sentences involving this word into S_s using '∨'. Briefly, the trouble is that

S_s will sanction the inference of '(q ∨ p)' from '(p ∨ q)'. But this inference is not valid for 'unless'; for instance, it would be nonsense to claim that 'Unless you will see the stars tonight, it is cloudy' is true if 'Unless it will be cloudy, you will see the stars tonight' is. I would say that this problem indicates a limitation of S_s, that cannot be cleared up without constructing a formal theory of subjunctive conditionals (see II.3, example xxiv). But that's another story.

Bearing in mind the point that disjunction in S_s is inclusive, let's consider a few more examples. Sentences such as 'Alaska or Texas is the largest state in the union' and 'It will rain tonight or I'm a monkey's uncle' would be translated in S_s by sentences of the form '(p ∨ q)'. But we ought to be a bit hesitant about translating something like 'My son will mow the lawn or I'll take away his teddy bear' in this way, since it connotes that it will not be the case that both disjuncts hold true.

There are many examples in which the joint case simply does not come up: for instance, 'He is in Boston or Zagreb', or 'Either he beats his wife or he doesn't'. In both of these, the situation in which both disjuncts are true is excluded out of hand, and it doesn't seem a matter of great importance whether or not we say these sentences are true in case both of their disjuncts are. For this reason, there seems to be no harm in translating these into S_s by formulas of the sort 'p ∨ q'.

6. Let's turn now to the question of logical rules for disjunction. In this case, it's more convenient to take up the elimination rule first. The rule of disjunction elimination is more complicated than the rules we have discussed up to now; but it corresponds closely to the way we actually reason from disjunctions, and again is made to seem natural by considering examples of this sort of reasoning. Let's consider a case in which we must argue from a disjunction to some conclusion; for instance, suppose we want to show that if a number n is greater than 20 or less than 10, then $(n - 15)^2$ is greater than 25. The thing to do is to reason as follows.

1. Assume that $n > 20$ or $n < 10$.
2. Suppose that $n > 20$.
3. Then $(n - 15) > 5$, so $(n - 15)^2 > 25$.
4. On the other hand, suppose that $n < 10$.
5. Then $(n - 15) < -5$.
6. So in this case also, $(n - 15)^2 > 25$.
7. Therefore, $(n - 15)^2 > 25$.
8. Hence, if $n > 20$ or $n < 10$, then $(n - 15)^2 > 25$.

(v)

Translating the key steps of this argument into S_s and putting the argument into the format of subordinated derivations, we get something like this.

```
| |_(p ∨ q)
| |  |_p
| |  |  ⋮
| |  |  r
| |
| |  |_q
| |  |  ⋮
| |  |  r
| |  r
| ((p ∨ q) ⊃ r)
```

(vi)

Here, 'p' stands for '$n > 20$', 'q' for '$n < 10$', and 'r' for '$(n - 15)^2 > 25$'. Example vi gives us a pretty good picture of how to get a conclusion from a disjunction '(p ∨ q)'. Besides '(p ∨ q)' we need two other premisses in order to extract a conclusion 'r': a derivation of 'r' from 'p' and one of 'r' from 'q'.

In general, the rule of *disjunction elimination* (*dis elim*) looks like this.

```
| ⋮              | ⋮              | ⋮
| (A ∨ B)        | (A ∨ B)        |  |_B
| ⋮              | ⋮              |  |  ⋮
|  |_A           |  |_B           |  |  C
|  |  ⋮          |  |  ⋮          | ⋮
|  |  C          |  |  C          | (A ∨ B)        etc.
| ⋮              | ⋮              | ⋮
|  |_B           |  |_A           |  |_A
|  |  ⋮          |  |  ⋮          |  |  ⋮
|  |  C          |  |  C          |  |  C
| ⋮              | ⋮              | ⋮
| C              | C              | C
```

That is, whenever we have obtained as items in a derivation a formula $(A \vee B)$ and derivations of C from A and of C from B (in any order), we may then write down C. Above, all the different possible arrangements aren't displayed; there are six in all.

Notice, by the way, that the rule of dis elim makes good sense from the standpoint of truth. If $(A \vee B)$ is true, then A is true or B is true, and so if C follows from A and follows from B, then C is true if A is, and also true if B is.

So C is true in either case. We will develop this idea in great detail below in Chapter VI, where we discuss the semantic interpretation of S_s.

The rule of disjunction elimination, by the way, corresponds to one of the favorite devices of classical orators. This is to discuss a question by beginning with two alternatives, and then giving a pair of speeches designed to show that in either case the desired conclusion follows. Socrates' speech about death in the *Apology* is an example: he asserts that death is either a form of unconsciousness or of consciousness and then argues that in either case it is a good.

7. It isn't easy to motivate the rule of disjunction introduction in the same way we have developed previous rules by referring to samples of reasoning. This is because we hardly ever use this rule in actual argumentation; after we have worked out the rule you will see why this is so.

However, we can get at the rule in a slightly different way by talking about the *justification* of a disjunctive claim. If Peterson asserts that Scrags' car has a short in the generator or a broken fanbelt, how would we go about justifying this assertion? Well, we will have determined that Peterson is right as soon as we have verified that the car has a short in the generator, or verified that it has a broken fanbelt. We know that a disjunctive claim is true as soon as we know that either of its disjuncts is true.

Surely, the rule sanctioning this sort of justification is one which permits the inference of a disjunction from either of its disjuncts: for instance, 'Scrags' car has a short in the generator or a broken fanbelt' from 'Scrags' car has a short in the generator', and 'Scrags' car has a short in the generator or a broken fanbelt' from 'Scrags' car has a broken fanbelt'.

The formal counterpart in S_s of this rule will permit us to argue to disjunctions $(A \vee B)$; this rule of *disjunction introduction* (*dis int*) takes the following form.

```
| ⋮              | ⋮
| A              | B
| ⋮              | ⋮
| (A ∨ B)        | (A ∨ B)
```

Whenever in a derivation we have written either A or B, we may thereafter write the disjunction $(A \vee B)$.

Now that we have the rule, we can go back to the point with which we began this section and explain why the rule is not often used explicitly in argumentation. The reason is that if we already *know* a thing—say, that the car has a broken fanbelt—we wouldn't bother to reason from this to a disjunction—say, that the car has a short in the generator or a broken fanbelt. We would only lose information in doing this; it is silly, unless one is dissimulating, to assert a disjunction when one knows which of the two disjuncts is true.

Nevertheless, we have shown that dis int is important in the justification of disjunctive statements, and it turns out to be just the right rule for formal purposes. As we will see later, it generates, together with the other rules, a theory of disjunction which is *semantically complete.*

8. At this point we have, besides reit, six rules with which to build derivations. (Two rules apiece for each of the connectives '⊃', '∧', and '∨'; remember that 'implication elimination' is just another name for *modus ponens.*) Here are some examples.

```
1   |_(p ∨ (p ∧ r))        hyp
2   |  |_p                 hyp
    |  |
3   |  |_(p ∧ r)           hyp
4   |  |  p                3, conj elim
5   | p                    1, 2, 3–4, dis elim
```

(vii)

```
1   |_((p ∨ q) ⊃ r)        hyp
2   |  |_p                 hyp
3   |  |  (p ∨ q)          2, dis int
4   |  |  ((p ∨ q) ⊃ r)    1, reit
5   |  |  r                3, 4, m p
6   | (p ⊃ r)              2–5, imp int
```

(viii)

```
1   |_(p ∨ (q ∧ r))              hyp
2   |  |_p                       hyp
3   |  |  (p ∨ q)                2, dis int
4   |  |  (p ∨ r)                2, dis int
5   |  |  ((p ∨ q) ∧ (p ∨ r))    3, 4, conj int
    |
6   |  |_(q ∧ r)                 hyp
7   |  |  q                      6, conj elim
8   |  |  (p ∨ q)                7, dis int
9   |  |  r                      6, conj elim
10  |  |  (p ∨ r)                9, dis int
11  |  |  ((p ∨ q) ∧ (p ∨ r))    8, 10, conj int
12  | ((p ∨ q) ∧ (p ∨ r))       1, 2–5, 6–11, dis elim
```

(ix)


```
1   |_((p ∨ q) ∧ (q ⊃ r))   hyp
2   | (q ⊃ r)               1, conj elim
3   | (p ∨ q)               1, conj elim
4   |  |_p                  hyp
5   |  |  (p ∨ r)           4, dis int
    |
6   |  |_q                  hyp
7   |  |  (q ⊃ r)           2, reit
8   |  |  r                 6, 7, m p
9   |  |  (p ∨ r)           8, dis int
10  | (p ∨ r)               3, 4–5, 6–9, dis elim
```

(x)

```
1   |_(p ∧ ((p ⊃ q) ∨ r))   hyp
2   | p                     1, conj elim
3   | ((p ⊃ q) ∨ r)         1, conj elim
4   |  |_(p ⊃ q)            hyp
5   |  |  p                 2, reit
6   |  |  q                 4, 5, m p
7   |  |  (q ∨ r)           6, dis int
    |
8   |  |_r                  hyp
9   |  |  (q ∨ r)           8, dis int
10  | (q ∨ r)               3, 4–7, 8–9, dis elim
```

(xi)

9. Now we come to negation. We have, of course, already dealt with proofs involving '∼' in the system S_0 of Chapter I, and have discussed in Chapter II the translation of negative English sentences into formal notation. Let's turn at once, then, to the problem of reproducing in S_s the way in which negation figures in correct reasoning.

The rule of negation introduction is fairly straightforward. Suppose that we want to prove something negative: say, that the square root of 2 is irrational. This means that there are no whole numbers n and m such that $\sqrt{2} = n/m$. This is to say, $\sqrt{2}$ is not a fraction, or *ratio.*

The following proof of the irrationality of $\sqrt{2}$ was known in antiquity.

1. Suppose that $\sqrt{2}$ is rational.
2. Then for some whole numbers n and m, $\sqrt{2} = n/m$.

3. By factoring out all the common divisors of n and m, we obtain whole numbers j and k having no common divisors but 1, such that $^j/_k = {^n/_m}$.
4. In view of step 3, $^j/_k$ is in lowest terms.
5. Since $\sqrt{2} = {^j/_k}$, $2 = {^{j^2}/_{k^2}}$.
6. Therefore $2 \cdot k^2 = j^2$, so that j^2 is divisible by 2.
7. But if 2 divides j^2 (i.e., divides $j \cdot j$), then 2 is a factor of j, and so j is even.
8. Since j is even, 4 is a factor of j^2.
9. But then, since $2 \cdot k^2 = j^2$, 2 is a factor of k.
10. Therefore k and j are both divisible by 2.
11. So $^j/_k$ is not in lowest terms, which contradicts step 4.
12. Therefore $\sqrt{2}$ is not rational.

(xii)

In the last step of xii, a negative conclusion has been inferred from steps 1 to 11. For our purposes, the crucial features of the reasoning are steps 1, 4, and 11; the remaining steps are mathematical stages in the extraction of a contradiction from the assumption that $\sqrt{2}$ is rational.

Translating these main steps into $\mathbf{S_s}$ and casting the argument into subordinated form we obtain the following pattern.

```
| |_p
| | ⋮
| | q
| | ⋮
| | ~q
| ~p
```

(xiii)

This rendering of xii displays the rule of neg int perspicuously enough so that it is a simple matter to extract the general rule. Again, however, we have to remember that in xiii we could as well have gotten '~q' before 'q'.

```
| ⋮              | ⋮
| |_A            | |_A
| | ⋮            | | ⋮
| | B            | | ~B
| | ⋮            | | ⋮
| | ~B           | | B
| ⋮              | ⋮
| ~A             | ~A
```


Whenever we obtain as an item in a derivation a subordinated derivation of two contradictory formulas B and $\sim B$ (in either order) from a hypothesis A, we may conclude $\sim A$ in the original derivation.

You may already know the traditional name of this rule: *reductio ad absurdum*. Certainly, you have used arguments which are instances of this rule. Neg int is one of the most frequently used argument patterns in everyday reasoning; in any situation in which the object is to show that something is not the case, the natural thing to do is to assume it and attempt to derive an absurdity. In ordinary cases, this absurdity is not a flat contradiction, as we have required in $\mathbf{S_s}$; it often is a conclusion contrary to popular opinion, or perhaps to the accepted tenets of some profession. We can handle arguments of this sort in $\mathbf{S_s}$ by treating these opinions, tenets, or whatever as hypotheses of the whole argument. They would then be reiterated to produce an out-and-out contradiction: that is, a formula together with its negation.

10. The rule of negation elimination raises questions which are more ticklish than any we have encountered so far in this section. It is possible to disagree radically concerning this rule, and the decision made at this juncture can lead to various different logics. Nor are all of these options—in particular, the so-called intuitionistic and classical logics—mere formal games. Both have their appropriate philosophical justification, and both are founded in patterns of reasoning that actually are employed.

An examination of these issues thorough enough to give satisfaction would carry us far astray from the task of presenting the fundamental techniques of modern logic. So we will ride roughshod over the subtleties, and just present the classical approach. Once the techniques are out in the open, they can be applied as readily to intuitionistic as to classical systems of logic, and it will be more easy to appreciate subtleties. For the time being, though, keep in mind that there are legitimate alternatives to the course we will take.

11. To isolate a rule of negation elimination that will generate the classical logic, let's go back to *reductio ad absurdum* arguments. There is a frequently used type of argument which uses the device of reduction to absurdity and yet does not fit the pattern of negation elimination.

I am thinking here of cases in which the conclusion of the argument is not a negative statement. Consider, for instance, the following proof by *reductio* that every real number is equal to, greater than, or less than zero.

1. Suppose that it is not the case that for every real number x, $x = 0$ or $x > 0$ or $x < 0$.

2. Then for some real number, say r, it is not the case that r = 0 or r > 0 or r < 0.
3. Therefore not r = 0 and not r > 0 and not r < 0.
4. But since not r = 0 and not r > 0, then r < 0.
5. And in view of step 3, not r < 0; which contradicts step 4.
6. Therefore for every real number x, x = 0 or x > 0 or x < 0.

(xiv)

From the mathematical standpoint this argument is superficial, but it illustrates our point. Rendering the most important parts of xiv (namely, steps 1, 4, 5, and 6) in S_s, we get something looking like this.

```
| |_~p
| | :
| | q
| | ~q
| p
```

(xv)

Example xv is certainly reminiscent of the rule of negation introduction, but if you look back at this rule you will see that xv is *not* an instance of neg int. Now, can we *derive* the inference xv using neg int and the other rules of S_s? That is, can we use the rules we have discussed so far to get to 'p' from a derivation of 'q' and of '~q' from '~p'? You may want to try before reading on.

If you did, I hope you didn't succeed! Example xv cannot be derived from our present stock of rules. (I say this, though it isn't as easy as you might think to find a convincing proof that this is so.) The closest we can come is this.

```
| |_~p
| | :
| | q
| | ~q
| ~~p
```

(xvi)

In xvi, '~~p' is justified by the rule of neg int. But there is no way to get from '~~p' to 'p', our desired conclusion.

By now, you must have guessed what the rule of neg elim will be: it is precisely what we need in this situation, and can be pictured as follows.

```
| :
| ~~A
| :
| A
```

The rule of neg elim permits one to write down A in any derivation in which a step $\sim\sim A$ has appeared.

12. Using the rules of neg int and neg elim in connection with the other rules discussed above, we can build derivations of considerable complexity. Adding rules for negation, in fact, makes things much less straightforward than they were before; it often is necessary to be devious and indirect in finding derivations, as in the categorical derivation (example xviii) of '(p ∨ ~p)' below.

For this reason we will provide in this section a number of examples of derivations of various kinds. In all but two of these examples the reasons are omitted; filling the rest in is left as an exercise.

```
1  | p          hyp
2  |_~p         hyp
3  | |_~q       hyp
4  | | p        1, reit
5  | | ~p       2, reit
6  | ~~q        3–5, neg int
7  | q          6, neg elim
```

(xvii)

The next derivation shows that our system conforms to the law of excluded middle: '(p ∨ ~p)' can be derived categorically in S_s. The laws of excluded middle and noncontradiction are among the first principles of logic to have been formulated. According to the principle of excluded middle any sentence such as 'It is raining this afternoon or it isn't' which corresponds to the formula '(p ∨ ~p)' must be true. According to the principle of non-contradiction any sentence such as 'It is raining this afternoon and it isn't' which corresponds to the formula '(p ∧ ~p)' must be false.

Like many derivations in S_s with disjunctive conclusions, the argument is indirect. It would be impossible to obtain '(p ∨ ~p)' in a categorical derivation by the rule dis int, since neither 'p' or '~p' can be derived categorically in S_s. (We can't prove this yet, but it's so anyway.) The only way to carry out the derivation is to obtain '~~(p ∨ ~p)' using **neg int** and the rules for disjunction; then '(p ∨ ~p)' can be obtained by **neg elim**.

```
1  | |_~(p ∨ ~p)          hyp
2  | | |_p                hyp
3  | | | (p ∨ ~p)         2, dis int
4  | | | ~(p ∨ ~p)        1, reit
5  | | ~p                 2–4, neg int
6  | | (p ∨ ~p)           5, dis int
7  | ~~(p ∨ ~p)           1–6, neg int
8  | (p ∨ ~p)             7, neg elim
          (xviii)
```

The derivations below use the rules of S_s (except for the rules for '≡', which we've not yet discussed) in various combinations.

```
|_~(p ∨ q)
| |_p
| | (p ∨ q)
| | ~(p ∨ q)
| ~p
| |_q
| | (p ∨ q)
| | ~(p ∨ q)
| ~q
| (~p ∧ ~q)
   (xix)
```

```
|_(~p ∨ q)
| |_p
| | (~p ∨ q)
| | |_~p
| | | |_~q
| | | | p
| | | | ~p
| | | ~~q
| | | q
| | |
| | |_q
| |
| | q
| (p ⊃ q)
      (xx)
```

```
| |_~~(p ∧ ~p)
| | (p ∧ ~p)
| | p
| | ~p
| ~~~(p ∧ ~p)
     (xxi)
```

```
| p
|_~q
| |_(p ⊃ q)
| | p
| | q
| | ~q
| ~(p ⊃ q)
    (xxii)
```



```
| |_(p ⊃ ~q)
| | |_q
| | | |_p
| | | | (p ⊃ ~q)
| | | | ~q
| | | | q
| | | ~p
| | (q ⊃ ~p)
| ((p ⊃ ~q) ⊃ (q ⊃ ~p))
      (xxiii)
```

```
|_~(p ⊃ q)
| |_q
| | |_p
| | | q
| | (p ⊃ q)
| | ~(p ⊃ q)
| ~q
   (xxiv)
```

13. The only connective of S_s which we've not yet discussed is equivalence or, as it is sometimes called, the biconditional. This is often expressed in English by 'if and only if', or by locutions using phrases such as 'necessary and sufficient condition'. For instance, 'The senate will pass the bill if and only if the house of representatives does', and 'This polygon has three sides if and only if it has three angles' would both be rendered in S_s by formulas of the sort '(p ≡ q)'.

At this point there probably is no need to produce examples of arguments to and from biconditional statements. All that is needed to work out the rules for '≡' is the thought that statements such as 'Harrison is the mayor if and only if he has been duly elected' amount to conjunctions of implicative statements—in this case, to the conjunction 'Harrison is the mayor if he has been duly elected, and Harrison is the mayor only if he has been duly elected'; i.e., to the conjunction 'If he has been duly elected then Harrison is the mayor, and if Harrison is the mayor then he has been duly elected'.

The rules for '≡', then, will be doubled versions of the rules for '⊃'. Thus the rule of equivalence introduction takes the following form.

```
| ⋮                | ⋮
| |_A              | |_B
| | ⋮              | | ⋮
| | B              | | A
| ⋮                | ⋮
| |_B              | |_A
| | ⋮              | | ⋮
| | A              | | B
| ⋮                | ⋮
| (A ≡ B)          | (A ≡ B)
```

Whenever we have obtained derivations of A from B and of B from A (in either order) as items in another derivation, we may thereafter write down $(A \equiv B)$.

And the rule of equivalence elimination is like *modus ponens*, although it is symmetrical.

```
| ⋮          | ⋮          | ⋮          | ⋮
| (A ≡ B)    | (A ≡ B)    | A          | B
| ⋮          | ⋮          | ⋮          | ⋮
| A          | B          | (A ≡ B)    | (A ≡ B)
| ⋮          | ⋮          | ⋮          | ⋮
| B          | A          | B          | A
```

Whenever we have obtained both $(A \equiv B)$ and A (in any order) in a derivation, we may conclude B; and whenever we have obtained both $(A \equiv B)$ and B (in any order) in a derivation, we may conclude A.

14. Below are some examples of derivations using the rules of eqv int and eqv elim, together with the other rules of S_s.

```
 1  | |_(p ⊃ q)                  hyp
 2  | | |_~q                     hyp
 3  | | | |_p                    hyp
 4  | | | | (p ⊃ q)             1, reit
 5  | | | | q                   3, 4, m p
 6  | | | | ~q                  2, reit
 7  | | | ~p                     3–6, neg int
 8  | | (~q ⊃ ~p)               2–7, imp int

 9  | |_(~q ⊃ ~p)                hyp
10  | | |_p                      hyp
11  | | | |_~q                   hyp
12  | | | | (~q ⊃ ~p)           9, reit
13  | | | | ~p                  11, 12, m p
14  | | | | p                   10, reit
15  | | | ~~q                    11–14, neg int
16  | | | q                      15, neg elim
17  | | (p ⊃ q)                  10–16, imp int
18  | ((p ⊃ q) ≡ (~q ⊃ ~p))      1–8, 9–17, eqv int
```

(*xxv*)


```
1  |_(p ≡ (q ≡ q))      hyp
2  | |_q                hyp
   | |
3  | (q ≡ q)            2, eqv int
4  | p                  1, 3, eqv elim
```

(*xxvi*)

In this last example, step 3 suffices to introduce '(q ≡ q)' by eqv int, because it is not required by this rule that the derivations of B from A and of A from B needed to introduce $(A \equiv B)$ must be distinct. In this particular case A is the same formula as B and there is just one derivation, consisting only of step 2, but this is enough to justify step 3.

```
| (p ≡ q)
|_(r ≡ s)
| |_(p ⊃ r)
| | |_q
| | | (p ≡ q)
| | | p
| | | (p ⊃ r)
| | | r
| | | (r ≡ s)
| | | s
| | (q ⊃ s)
|
| |_(q ⊃ s)
| | |_p
| | | (p ≡ q)
| | | q
| | | (q ⊃ s)
| | | s
| | | (r ≡ s)
| | | r
| | (p ⊃ r)
| ((p ⊃ r) ≡ (q ⊃ s))
```

(*xxvii*)

```
|_(p ≡ q)
| |_(p ∧ ~q)
| | p
| | ~q
| | (p ≡ q)
| | q
| ~(p ∧ ~q)
| |_(~p ∧ q)
| | ~p
| | q
| | (p ≡ q)
| | p
| ~(~p ∧ q)
| (~(p ∧ ~q) ∧ ~(~p ∧ q))
```

(*xxviii*)

15. Before turning to other matters, let's think a moment about how to find derivations in S_s. In this area nothing can substitute for practice, and in working on the exercises you may already have discovered for yourself some of the tricks discussed below. But maybe it will be helpful to make them explicit here.

First, there are some rules that can be very useful in keeping on the right track when trying to construct a derivation. They can never get you into trouble and often will keep you out of blind alleys.

1. When trying to derive a formula of the sort $(A \supset B)$, always set up a subordinate derivation with hypothesis A and try to get B.
2. Similarly, when trying to derive $\sim A$, set up a subordinate derivation with hypotheses A, and try to get a contradiction.
3. Do the same with formulas of the sort $(A \wedge B)$ and $(A \equiv B)$; that is, try to get them by the appropriate introduction rules.
4. Be careful about getting formulas of the sort $(A \vee B)$ directly, by dis int. Often you have to get it from $\sim\sim(A \vee B)$ by neg elim, or by dis elim or some other elimination rule. See, for instance, example xviii and the paragraphs preceding it.
5. Whenever you have a premiss of the sort $(A \vee B)$, use dis elim unless there is some obvious other way of getting the desired conclusion.
6. Whenever there is an obvious application of an elimination rule—for instance, if you have two formulas A and $(A \supset B)$, or a formula $(A \wedge B)$—use the elimination rule unless you see some obvious other way to get the desired conclusion.
7. In general, work from the top and bottom of an incomplete derivation in towards the middle, using rules 1 to 6 wherever applicable. This means, roughly, using elimination rules to argue from the formulas given at the top and introduction rules to get the formulas desired at the bottom. If the problem is solvable, you often will see at once how to fill in the middle to obtain the desired derivation. But sometimes ingenuity is required, even after using rules 1 to 6 as much as they can be used. Practice is the only thing that can develop real skill in such situations.

We've called the above admonitions "rules", but you must realize that they aren't *rules of* the system S_s. The latter sort of rules, if you like, are rules for *recognizing* (or *characterizing*) derivations; the former are rules of thumb for *finding* derivations. It is like the difference between rules of chess like 'Try not to lose your queen' and 'The king may move one square in any direction, unless the square is threatened by an opposing piece'. The latter is one of the rules constituting the game of chess; the former is a piece of advice intended to help a player do well.

At the risk of making this analogy tedious, let's consider one more point.

To play chess at all well, you have to think ahead—and this requires the assimilation of certain recurring patterns. You learn that if you take a piece that is defended, the attacking piece will usually be taken; for instance, if you take a defended knight with a bishop, the bishop will ordinarily be taken itself. Stringing together such patterns, one is able to extrapolate a position into the future and weigh possible alternatives with more skill. Very good chess players learn by rote large numbers of classical openings, many of them quite long and complicated.

Actually, this sort of thing is not peculiar to games—it takes place in learning any skill. Basic steps cluster into patterns, and these in turn form into larger groups. And in learning to find derivations in S_s, the sort of patterns one has to learn are *argument forms.* For instance, example xvii establishes an important such pattern: whenever you have contradictory formulas A and $\sim A$ in a derivation you can use the rules of proof of S_s to obtain any formula B whatsoever. The strategy is this.

```
| ⋮                 | ⋮
| A                 | ~A
| ⋮                 | ⋮
| ~A                | A
| ⋮                 | ⋮
|  |_~B             |  |_~B
|  |  A             |  |  A
|  |  ~A            |  |  ~A
| ~~B               | ~~B
| B                 | B
```

(*xxix*)

The above display shows in general how to go about getting *any* formula B, once one has obtained *any* contradiction. In a way, xxix establishes a rule of *contradiction elimination*: from A and $\sim A$ to infer B. But this is not a rule of S_s in the sense that, say, neg elim is a rule of that system. If, for instance, we were simply to proceed as follows,

```
|_p
|  |_~p
|  |  p
|  |  q
```

(*xxx*)

this would *not* be a derivation in S_s. But xxix shows us how to fill in steps in xxx to make a derivation in S_s out of it.

Rules such as neg elim are often called *primitive*, and rules such as contradiction elimination *derived*. The former rule is a primitive rule of S_s, in that it is a basic or unanalyzable principle of proof in the system. The latter rule is derivable in S_s, because the primitive rules of S_s may be manipulated in such a way as to justify any instance of contradiction elimination, as is shown by the scheme xxix. Derived rules, then, are justified by argument patterns which are complexes of primitive rules.

Notice, by the way, that the notions of *primitive* and *derived rule* are relative to the system under consideration. Contradiction elimination might well be a primitive rule of some other system.

Here are two more argument patterns, which often are useful in finding derivations.

```
| ⋮
| A
|   |_B
|   | A
| (B ⊃ A)
(xxxi)
```

```
| ⋮
| (A ∨ B)
| ⋮
| ~A
|   |_A
|   |   |_~B
|   |   | A
|   |   | ~A
|   | ~~B
|   | B
|
|   |_B
|   |
| B
(xxxii)
```

Notice that contradiction elimination is used in constructing example xxxii.

Below are listed some other argument patterns that are especially useful. Rather than filling in the steps of these patterns, however, we will simply list their premisses and conclusions. Their completion is left to you.

```
|_~A          |_(~A ⊃ ~B)    |_(A ⊃ B)      |_(A ⊃ ~B)
| ⋮           | ⋮            | ⋮            | ⋮
| (A ⊃ B)     | (B ⊃ A)      | (~B ⊃ ~A)    | (B ⊃ ~A)

              |_(~A ⊃ B)     | (A ∨ B)
              | ⋮            |_~B
              | (~B ⊃ A)     | ⋮
                             | A
```



```
|_~(A ∨ B)     |_(~A ∧ ~B)    |_~(A ∧ B)     |_(~A ∨ ~B)
| ⋮            | ⋮            | ⋮            | ⋮
| (~A ∧ ~B)    | ~(A ∨ B)     | (~A ∨ ~B)    | ~(A ∧ B)

|_(~A ∨ B)     |_(A ⊃ B)      |_~(A ⊃ B)     |_(A ∧ ~B)
| ⋮            | ⋮            | ⋮            | ⋮
| (A ⊃ B)      | (~A ∨ B)     | (A ∧ ~B)     | ~(A ⊃ B)
```

A final warning. The instructions above will usually result in successful discovery of a derivation, if there is one to be found. Practice in using them is all that is required. But if for some reason you set out to derive something that is underivable in S_s, they of course will not generate an answer; these are instructions for finding a derivation in case one exists, not for showing that there is no derivation in case none exists. One way students sometimes get into trouble is by setting out to derive something that is underivable; then, in order to get an answer, they misapply the rules of S_s. (Deriving '$\sim p$' from '$\sim(p \wedge q)$' by "conj elim" is a typical mistake of this sort.) So, if you find yourself unable to find a correct derivation of one of the exercises, always check to see whether you have copied or translated the exercise properly.

Exercises

1. Whenever possible, translate the following English sentences into S_s. If you feel that a successful translation is not possible, explain why. In giving a translation, be sure to specify which English sentence each of the sentence parameters in the translation stands for.
   (a) He'll come tomorrow if his car doesn't break down on the way.
   (b) Both Baltimore and Hagerstown are in Maryland.
   (c) I never learned to speak German well, but I can understand you if you speak slowly.
   (d) Where is the nearest post office?
   (e) The accused is guilty if and only if he was not at home on the night of July 15; but his wife will not testify against him and there are no other witnesses.
   (f) I've only been in town a week, and already I've been to three parties and a concert.
   (g) Everyone in this room is unable to read music.
   (h) You are allowed to walk your dog only if you have him on a leash, and the dog must have a license.
   (i) $2^{10} + 1$ is a prime number just in case it is not divisible by 17 or by 21.
   (j) If the boss is neither at home nor at work, then he is at the golf course or something extraordinary has happened and he has been detained; but if he has been detained we'll know about it soon.

(k) New York is nearer to Cleveland than to Kingston.
(l) If we're only two days behind schedule, we may be able to catch up if we get some luck.
(m) Go five miles down the road and turn left at the fire station.
(n) The best way to get there is to go five miles down the road and turn left at the fire station.
(o) He didn't know that Mozart and his father were musicians.
(p) If the train isn't late, I'll stop at a bar and get a drink; and that's for sure.

2. Translate the following into $\mathbf{S_s}$, again specifying which English sentence each of the sentence variables signifies. Then find a derivation in $\mathbf{S_s}$ of the conclusion from the hypotheses.

(a) If 2 < 3 then not 3 < 2.
Therefore if 3 < 2 then not 2 < 3.
(b) Oscar is at home, or, if not, he left a message.
Therefore if he didn't leave a message, Oscar is at home.
(c) Albert is either a fool or a liar.
If he is a liar, then what he told me about his sister is false, and I'll look like a fool.
Therefore Albert is a fool or I'll look like a fool.
(d) Fort Wayne is neither in Ohio nor in Illinois.
If Fort Wayne is in Cook County, it is in Illinois.
Therefore it is not the case that Fort Wayne is in Cook County and in Indiana.
(e) If John is arrested he will plead guilty and have to pay a large fine, or plead not guilty and go to a lot of trouble.
If he has to pay a large fine, John will go to a lot of trouble.
Therefore John will go to a lot of trouble if he is arrested.
(f) It is not the case that if God exists, there is unnecessary evil in the world.
Therefore God exists and there is not unnecessary evil in the world.

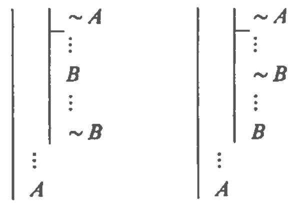
3. Give derivations in $\mathbf{S_s}$ of the following, supplying reasons for all steps.
(a) $(q \vee p)$ from $(p \vee q)$
(b) $(q \wedge p)$ from $(p \wedge q)$
(c) $((p \wedge r) \vee q)$ from $((p \vee q) \wedge r)$
(d) $((p \wedge q) \supset r)$ from $(p \supset (q \supset r))$
(e) $(p \supset (q \supset r))$ from $((p \wedge q) \supset r)$
(f) $\sim\sim r$ from r
(g) $(p \supset (q \vee r))$ from $\sim p$
(h) $((p \vee q) \wedge (p \vee \sim q))$ from p
(i) $\sim p$ from $(p \supset \sim p)$
(j) p from $(\sim p \supset p)$
(k) $((p \wedge \sim\sim q) \supset r)$ from $((p \wedge q) \supset r)$
(l) $((p \vee q) \supset r)$ from $((p \vee \sim\sim q) \supset r)$

(m) $(\sim s \supset (r \vee p))$ from $((p \supset q) \supset (r \vee s))$
(n) p from $((p \supset q) \supset q)$ and $(q \supset \sim q)$
(o) $(((q \supset p) \supset p) \supset p)$ from $((p \supset q) \supset p)$
(p) $(r \supset p)$ from $((p \supset q) \supset (r \supset q))$ and $(q \supset p)$
(q) $((p \supset r) \supset r)$ from $((p \supset q) \supset q)$ and $(q \supset r)$
(r) $(((r \supset q) \supset s) \supset ((r \supset p) \supset s))$ from $(p \supset q)$
(s) $(p \vee r)$ from $(p \vee (q \equiv r))$ and $(p \vee q)$
(t) $((p \wedge q) \vee (\sim p \wedge \sim q))$ from $(p \equiv q)$
(u) $(p \vee q)$ from $\sim(p \equiv q)$
(v) $((p \equiv q) \equiv r)$ from $(p \equiv (q \equiv r))$

4. Find categorical derivations in $\mathbf{S_s}$ of the following, supplying a reason for each step.
(a) $(p \vee (p \supset q))$
(b) $((p \supset q) \equiv (\sim p \vee q))$
(c) $((p \vee q) \equiv \sim(\sim p \wedge \sim q))$
(d) $((p \wedge q) \equiv \sim(\sim p \vee \sim q))$
(e) $((((p \wedge q) \vee (\sim p \wedge q)) \vee (p \wedge \sim q)) \vee (\sim p \wedge \sim q))$
(f) $(((p \supset q) \supset p) \supset p)$
(g) $((p \equiv q) \equiv (\sim p \equiv \sim q))$
(h) $(((p \supset q) \supset q) \equiv (p \vee q))$
(i) $((p \vee p) \supset p)$

Problems

1. Devise introduction and elimination rules for the exclusive sense of disjunction, in which the case in which both disjuncts are true is ruled out.
2. Equivalence might have been *defined* in $\mathbf{S_s}$; i.e., formulas such as '$((p \supset q) \wedge (q \supset p))$' might have been used in place of formulas such as '$(p \equiv q)$'. Are there other connectives of $\mathbf{S_s}$ that can be defined in terms of the remaining connectives? How many connectives of $\mathbf{S_s}$ can be eliminated in this way? What are the criteria that allow us to say that a connective may be defined in terms of others?
3. Let the system $\mathbf{S_s}'$ be like $\mathbf{S_s}$ except that the rules of neg int and neg elim are replaced by the following rule, discussed in Section 11.



Show that $\mathbf{S_s}$ and $\mathbf{S_s}'$ are equivalent systems. (Clearly, $\mathbf{S_s}$ and $\mathbf{S_s}'$ are not *identical* systems, since they have different rules of inference, so different arrays of formulas will count as proofs in them. Part of the problem is to figure out an appropriate sense of "equivalence" in which the two systems *are* equivalent.)

V Sentence Logic: Syntax

1. Students sometimes find it difficult to adjust to the sort of thing we will begin to do in this chapter. The purpose of this section and the next is to explain, in a rough and preliminary way, what we will be doing and why it is worthwhile at all.

The subject of our investigation will be a system $\mathbf{H}_s$ very like the one presented back in Chapter I. ('H' is for David Hilbert, a mathematician whose name is associated with axiomatic systems of this kind.) But now that we know a bit more about logic, we will be able to develop this material in more generality and depth. Not only will the new system be more powerful and interesting than S_0, but our account of it will be more sophisticated.

Let's discuss the latter, methodological question in more detail. In Chapters I, III, and IV we were concerned primarily with producing proofs and derivations according to the rules of the systems under consideration; in Chapters II and III, we also occupied ourselves with relationships between these formulas and derivations, on the one hand, and sentences and arguments of

English on the other. Now, however, that we've done enough theorem-proving and derivation-finding to get a feel for the relationship of these systems to actual reasoning, we can turn to other matters. We will now begin to concentrate on a more general and theoretical treatment of logical systems.

Some analogies may help to clarify what is going on here. Most people know that there is quite a difference between being a musician and being a musicologist. It's no accident that many people are both, but nevertheless, some very good musicians know almost nothing about harmonic theory and the like, or even about the theory of how to play their instruments. (Such persons are usually bad teachers; they can *use* musical technique, but not communicate about it.) On the other hand, there is no reason why a musicologist has to be a good musician; he may have a ready grasp of the theory of music and be apt at analyzing musical works without being able to compose or even play well.

One of the most spectacular examples of this sort of thing is language. A person may be able to speak a language fluently—to produce and recognize grammatical discourse with ease—without knowing a thing about the grammar of the language. He knows the language, in being able to use it, without knowing about the language in the way that a linguist would be expected to know about it.

To a considerable extent the situation as regards formal languages is parallel to this. Up to this point we've concentrated on learning to *use* these languages—to find proofs in them and so forth. Logicians, however, are more like linguists than native speakers of natural languages; they are interested more in obtaining general and discursive knowledge of logical systems than in being "fluent" users of them.

All of these paragraphs have been devoted to saying in various ways that we are now going to talk *about* formal systems rather than *in* them. We have emphasized this because the distinction often troubles students. One reason for this is that the objects studied in logic are themselves linguistic. There is no danger of confusing bugs and theories about bugs, and so entomologists don't have to worry about distinguishing the two. But logicians must be more careful about such matters, and so distinguish what they call the *object-language* of an investigation from the *metalanguage*. The former is the language under investigation—for instance, S_s in Chapter IV— and the latter is the language used to discuss the object-language. In Chapter IV, then, the metalanguage was English, which is not a formal language.

This distinction is not meant to set a gulf between two kinds of languages, object-languages and metalanguages. The distinction between object-language and metalanguage is not absolute, but is always made with respect to a specific logical investigation, in which a language is used to study a formalized language. This, however, doesn't mean that the metalanguage may not itself be formalized and thus become the object-language for another metalanguage. For instance, logical studies have been carried out in English of two formal languages at once, one of them a metalanguage for the other. In such a case we would have a formalized object-language and a formalized metalanguage; the English of the study would be a metametalanguage of the former.

2. We've now agreed to concentrate on talking about logical systems, but haven't said much about how we intend to do this. What tools will we use in developing an account of such systems? Here, the fact that these systems are *formalized* languages is crucial. The precision and definiteness with which such languages are formulated allows logicians to develop their treatments of them *deductively*, with postulates, definitions, and theorems; thus, the more advanced portions of a well-made logical theory will be developed from earlier portions by means of rigorous arguments. In a word, logicians use the methods of mathematics.

To a considerable extent this is like a methodological device used by theoreticians in the natural and social sciences. The technique is this: in studying a subject-matter, first idealize it in mathematical terms, and then direct theoretical attention to this idealization. Sometimes—especially in the social sciences—this sort of enterprise is called building a mathematical *model*. Though this terminology is a bit deprecatory when used of very successful theories, it would not be inaccurate to regard logical systems as mathematical models of certain areas of reasoning.

Do not be surprised later, then, when we begin to state definitions and to prove propositions, as mathematicians do; this is simply the method we'll use in developing our account of formal systems. For purposes of clarity, it is good to distinguish theorems *of* the formal system under consideration (i.e., formulas that can be proved by means of the rules of these systems) from theorems proved *about* these systems in some metalanguage. The latter are usually called *metatheorems*.

3. Proceeding now to the system $\mathbf{H_s}$, we first want to characterize its formulas. Like $\mathbf{S_0}$, $\mathbf{H_s}$ will possess only connectives for negation and implication but, like $\mathbf{S_s}$, infinitely many sentence parameters. (It may seem that $\mathbf{H_s}$ is impoverished in comparison to $\mathbf{S_s}$ as regards connectives; $\mathbf{H_s}$ has just two, whereas $\mathbf{S_s}$ has five. But this isn't really so; we will show below, in

Section 11, that it's possible to *define* these other connectives in terms of negation and implication.)

1. Any of the sentence parameters $P_1, P_2, P_3, \ldots$ is a formula of $\mathbf{H_s}$.
2. If A is a formula of $\mathbf{H_s}$, then so is $\sim A$.
3. If A and B are formulas of $\mathbf{H_s}$, then so is $(A \supset B)$.

The symbols of $\mathbf{H_s}$ are the connectives $\sim$ and $\supset$, the parentheses) and (, and the sentence parameters $P_1, P_2, \ldots$. A string of symbols of $\mathbf{H_s}$ qualifies as a formula of $\mathbf{H_s}$ only if it can be shown to be a formula by repeated applications of 1, 2, and 3.

The above definition allows formulas to be generated by procedures that resemble proofs. The strings of symbols corresponding to axioms are the sentence parameters; just as axioms are the simplest possible theorems, sentence parameters are the simplest possible formulas. Formulas other than sentence parameters are complexes built up from sentence parameters by means of rules 2 and 3. Thus, if P, Q, and R are sentence parameters of $\mathbf{H_s}$, the following column would show how the formula $\sim\sim(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$ is generated.

1	R
2	$\sim R$
3	P
4	$(\sim R \supset P)$
5	Q
6	$(Q \supset Q)$
7	$((Q \supset Q) \supset (\sim R \supset P))$
8	$\sim((Q \supset Q) \supset (\sim R \supset P))$
9	$\sim P$
10	$(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$
11	$\sim(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$
12	$\sim\sim(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$

This column not only shows that $\sim\sim(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$ is a formula of $\mathbf{H_s}$, but it also displays the syntactic structure of that formula in showing how it can be produced by the rules that characterize the formulas of $\mathbf{H_s}$.

We have not paid much attention to "proofs" of this kind, because it's easy enough to recognize formulas without their aid. Since, however, we have found as yet no way of recognizing *theorems* without actually producing proofs of them, we have spent (and will spend) a good deal of time discussing proofs and derivations. But it's interesting to observe that with formulas as well as with theorems the same generative sort of characterization may be used.

4. This section is a digression, but has now become unavoidable. Some of you may have noticed that our notation for talking about logical systems was different in the above section; for instance, we made no use at all of quotation marks. Another thing that may have caught your eye and puzzled you is our talking of P, Q, and R as sentence parameters of $\mathbf{H}_s$, although the letters 'P', 'Q', and 'R' do not appear among the symbols 'P_1', 'P_2', and so on. How then can P be a sentence parameter of $\mathbf{H}_s$?

To straighten out these matters we will have to make explicit the conventions used by logicians in talking about formal languages. And this necessitates a full-scale discussion of the use-mention distinction.

In saying that the founder of the Lycaeum was a tutor of Alexander the Great, you *use* a rather roundabout phrase to *mention* Aristotle. A more direct way to do it would be to use the name 'Aristotle', and say that Aristotle was a tutor of Alexander the Great. Whenever we mention a thing we have to use something—usually a phrase or a word, written or spoken—as a name of the other. Now, when we're talking about nonlinguistic things, there is little danger of mixing these things up; we wouldn't be likely to confuse a person John with his name 'John'. But when we do logic, we find ourselves continually talking about expressions, and the danger of confusion is greater. For example, in the sentence

(*i*) John is a word with four letters

the word 'John' is clearly used as a name of itself rather than of a person John; the word is used *autonymously* ("selfnamingly") in this sentence. But the sentence

(*ii*) The first sentence in this chapter contains exactly six words

is *ambiguous*. It's not clear whether the phrase 'The first sentence in this chapter' is used autonymously here (in which case ii would be true) or as a name of the first sentence in this chapter (in which case ii would be false). Now, unless one is careful to distinguish use and mention, object-language and metalanguage, ambiguities of this sort can easily arise in doing logic. For this reason, logicians are often very attentive to policies designed to avoid confusion of use and mention.

It happens that there exists a convention ready-made for this purpose: quotation marks. Thus, example i and the true sense of example ii would more properly be written like this.

(*i'*) 'John' is a word with four letters.

(*ii'*) 'The first sentence in this chapter' contains exactly six words.

Compare also the use of quotation marks in the following pair of sentences.

(*iii*) Chicago is a city by Lake Michigan.

(*iv*) 'Chicago' is a seven-letter word.

In iii, one speaks of a city by way of its name. Similarly, in iv one speaks of the name of that city by way of an expression formed with quotation marks. Thus, in iii, 'Chicago' is used to name the city Chicago, and in iv ' 'Chicago' ' is used to name the word 'Chicago'.

We have already used this device in speaking about S_0 and other formalized languages; e.g., when we mentioned formulas such as '$(p \supset q)$'. But in one respect—and in this we follow ordinary practice—we have allowed exceptions to this rule. When an expression is *displayed*, rather than appearing in the text, it is not quoted; thus, above, the sentence iv is mentioned, not used. We have followed this use of quotation marks carefully, and will continue to follow it below, even though there isn't much likelihood of confusion arising from omitting quotes in discussing formalized languages. Often, the formulas of these languages have no customary use, and it would be clear from context that they are being mentioned, not used. For instance, expressions such as '$(p \supset q)$', unlike 'The first sentence in this chapter', have no customary English use. For this reason, no confusion would have been likely to arise even if we had neglected to use quotes entirely in Chapters I to IV.

Besides quotation marks, there is another way of talking about expressions that's especially important for logical purposes. In Section 1, above, we said that logicians are interested in *general* characteristics of formal systems. Just as we use quotes to form names of particular expressions (e.g., 'p' and '$(p \supset q)$') so we need an efficient way of talking about these general characteristics. A long time ago, in I.4, we discovered that plain English isn't very good for saying general things about the system S_0, and introduced letters such as 'A' to improve this situation. These variables enable us to express such things in a more elegant and perspicuous way.

As before, we will use the letters 'A', 'B', 'C', and 'D' to stand for formulas; in this chapter, for formulas of $\mathbf{H}_s$. Letters such as these are often called *metavariables*; in particular, 'A', 'B', and so on are metavariables taking formulas of $\mathbf{H}_s$ as values. Now a variable, roughly speaking, is any symbol that functions grammatically like a name, but which is allowed to assume various values for the sake of expressing general propositions. Using numerical variables '*m*' and '*n*', for instance, we might say that for all whole numbers *m* there is a whole number *n* such that $n = m + 1$. This expresses something general about numbers.

Our use of metavariables should produce no difficulty; everyone who has studied high-school algebra is familiar with variables and can handle them easily. And metavariables are used just like other variables; the only difference is that they occur in some metalanguage and usually take linguistic things as values.

Still another device that we have employed to refer to expressions is the use of names not involving quotation marks. For example, in this chapter 'iii' is a name of the sentence 'Chicago is a city by Lake Michigan'. In other words, iii *is* the sentence 'Chicago is a city by Lake Michigan'. Don't be deceived by this last way of putting it; it's perfectly correct. The only thing that may make it look a bit odd at first is that people aren't very used to names of expressions. Thus, it's easy to feel that 'iii' is used autonymously here, as a name of itself. But this is a mistake; 'iii' is not iii, any more than 'Chicago' is Chicago; iii is a sentence containing seven words, whereas 'iii' is a Roman numeral.

5. Now let's apply these ideas to our metalanguage for $\mathbf{H}_s$. The letters 'P_1', 'P_2', and so on are *names* of the sentence parameters of $\mathbf{H}_s$; they are *used*, not *mentioned*, in characterizing the formulas of this system. Thus, like the numeral 'iii' in the last example, 'P_1' is a name of an expression. Although it's perfectly proper to say that P_1 is a sentence parameter of $\mathbf{H}_s$, this by no means guarantees that 'P_1' is to be found among the sentence parameters of $\mathbf{H}_s$.

Like the metavariable 'A', the name 'P_1' is metalinguistic; this is the reason it is italicized, like 'A'. Unlike 'A', however, 'P_1' is always used to refer to a fixed expression, the first sentence parameter of $\mathbf{H}_s$; 'P_1' is a metalinguistic *constant*. This explains why 'P_1', 'P_2', and 'P_3' weren't quoted in Section 3, above; they were used, not mentioned.

Besides constants such as 'P_1' which name sentence parameters, it's convenient also to have *metavariables* that take sentence parameters of $\mathbf{H}_s$ as values; we will use 'P', 'Q', 'R', and 'S' for this purpose. Thus, 'P' may stand for any of the sentence parameters P_1, P_2, and so on. It should now be clear why it is legitimate to say, for instance, 'Let P be a sentence parameter of $\mathbf{H}_s$'; this is exactly like saying 'Let A be a formula of $\mathbf{H}_s$'. Both are proper and correct, though 'P' is not a sentence parameter and 'A' not a formula of $\mathbf{H}_s$.

There is one more point to be settled: what of symbols such as '$\supset$'?

According to the treatment of Chapters I to IV, '$\supset$' is a symbol of the object-language. This means that expressions such as '$A \supset B$' are hybrids, constituted of metalinguistic as well as object-linguistic symbols. Thus, in using '$A \supset B$' to speak of a formula $A \supset B$, 'A' and 'B' are used and '$\supset$' is mentioned. To be systematic in our policies concerning use and mention we must work out an account of these mixed expressions.

Various logicians have solved this problem differently. In his *Mathematical Logic*, W. V. Quine handles it by introducing special quotation marks '⌜' and '⌝'. These *quasi-quotes* are used to indicate that object-linguistic signs such as '$\sim$', '$\supset$', ')', and '(' are mentioned, while metalinguistic signs are used. Thus ⌜$\sim A$⌝ is an abbreviated way of referring to the formula resulting from placing '$\sim$' before the formula A. According to Quine, then, it would be correct to say that if A is a formula of $\mathbf{H}_s$, then ⌜$\sim A$⌝ is a formula of $\mathbf{H}_s$.

Alonzo Church, in his *Introduction to Mathematical Logic*, formulates a policy that makes it unnecessary to use special quotation marks. His idea is to make symbols such as '$\sim$' part of the metalanguage as well as the object-language, and to use them autonymously as names of themselves. Thus, for Church, '$\sim$' is *used* as well as mentioned when one speaks of a formula of the sort $\sim A$, and there is no need for quotes. The expression '$\sim A$' is no longer so much of a cross-breed, since both '$\sim$' and 'A' are used in it; but '$\sim$' is used as a name of itself. Church's convention is a very natural and simple one to use and our practice in previous chapters could be justified as a variant of this policy.

But in fact, we will adopt neither Quine's nor Church's approach. Instead, we will go one step further than Church, and follow the more radical usage of H. B. Curry's *Foundations of Mathematical Logic*. In practice, Curry's policy does not differ greatly from Church's, but in theory it is more abstract.

Consider as an example the metalinguistic assertion

(*v*) If A and B are formulas of $\mathbf{H}_s$, then so is $(A \supset B)$.

Quine would say that '$\supset$' is mentioned, not used in v; Church would say that '$\supset$' is used *and* mentioned. According to Curry, on the other hand, '$\supset$' is used, but need not be mentioned at all in v. Like the symbol 'P_1', '$\supset$' is part of the metalanguage for $\mathbf{H}_s$, and is a *name* of some symbol—not necessarily the symbol '$\supset$'—of the object-language. Curry's idea is to absorb the business of logic into the metalanguage: to make the treatment of logical systems entirely a matter of use. To some extent, this cuts us adrift from the object-language, although we still know everything about it that matters for logical purposes. For instance, we have no notion any more of what '$\supset$' names; its denotation may be 'b' or 'Δ', or perhaps '$\supset$'. It may even be a sound, or the moon. The question "What *are* the formulas and symbols of $\mathbf{H_s}$?" is left completely unanswered.

But really, there isn't any reason why we should expect to be presented with the symbols of $\mathbf{H_s}$. It's possible to learn a great deal about a subject without being handed the objects it studies; what we find, for instance, in a book on Roman history is not Julius Caesar or Augustus in person, but their names and a lot of talk about them. But a closer analogy is found in modern mathematical theories; in the study of whole numbers, the question "What *is* the number two?" is somehow inappropriate. The theory of whole numbers has many realizations and to fasten on only one of them as correct would detract from the generality of the theory. The same applies to logic; in an abstract treatment of logical systems, it's unnecessary to seize on a particular symbolic structure as *the* one that is intended. It is even unnecessarily restrictive to insist that only structures made up of written symbols be realizations of $\mathbf{H_s}$; for instance, $\mathbf{H_s}$ might be a sign language or spoken code of some sort.

6. This abstract approach enables us to simplify a number of matters. As an illustration, we will develop in this section a number of conventions regarding the elimination of parentheses.

The definition given in Section 3 of the formulas of $\mathbf{H_s}$ has fixed once and for all the number of parentheses which occur in any given formula of that system. But we are free to *talk about* formulas in any way that we find convenient. And since too many parentheses can be awkward and hard to read in names of formulas, it's convenient to have conventions for eliminating them. We must of course be careful that these conventions don't lead to any ambiguity or confusion in our metalanguage.

Our first convention is that outermost parentheses can always be dropped in referring to formulas. Thus, '$P_1 \supset P_1$' and '$A \supset (B \supset A)$' abbreviate '$(P_1 \supset P_1)$' and '$(A \supset (B \supset A))$' and so refer to the formulas $(P_1 \supset P_1)$ and $(A \supset (B \supset A))$, respectively.

A second convention allows us to eliminate parentheses by using dots after the symbol '$\supset$'. A metalinguistic expression containing '$\supset.$' is an abbreviation that is expanded by replacing '$\supset.$' by '$\supset($' and matching the left parenthesis with a right parenthesis placed as far as possible to the right without going through a right parenthesis mated with a left parenthesis to the left of the occurrence of '$\supset.$'. Thus, if there is no ')' to the right of the occurrence of '$\supset.$' mated with a '(' to the left of the occurrence of '$\supset.$', the left parenthesis is matched by a right parenthesis placed to the right of the entire expression. Otherwise, the left parenthesis is matched by a right parenthesis

placed next to the first appearance of ')' to the right of the occurrence of '$\supset.$' which is mated with a '(' to the left of the occurrence of '$\supset.$'.

This is something that gets fussy and complicated when stated generally, but isn't very difficult to pick up from examples. According to this convention,

$$A \supset. B \supset C$$

abbreviates

$$A \supset (B \supset C),$$

and

$$P \supset. (Q \supset (R \supset P)) \supset \sim Q$$

abbreviates

$$P \supset ((Q \supset (R \supset P)) \supset \sim Q).$$

In '$(A \supset. B \supset C) \supset D$', however, there are mated parentheses spanning the occurrence of '$\supset.$', so this abbreviates '$(A \supset (B \supset C)) \supset D$'. Likewise,

$$((P \supset Q) \supset. \sim Q \supset R) \supset S$$

abbreviates

$$((P \supset Q) \supset (\sim Q \supset R)) \supset S.$$

If there are several occurrences of '$\supset.$' in a name of a formula, the abbreviation can still be eliminated without ambiguity; it makes no difference in which order parentheses are restored. For instance,

$$B \supset. C \supset. D \supset B$$

abbreviates

$$B \supset (C \supset (D \supset B))$$

and

$$(A \supset. B \supset C) \supset. A \supset. (B \supset C) \supset \sim(A \supset B)$$

abbreviates

$$(A \supset (B \supset C)) \supset (A \supset ((B \supset C) \supset \sim(A \supset B))).$$

With a little practice, this dot notation becomes very convenient and easy to read.

A final convention that we will sometimes employ is that, where use of the above conventions does not result in an unambiguous name of a formula of

$\mathbf{H}_s$, the missing parentheses are to be restored by grouping at the left. A simple case of this is '$A \supset B \supset C$', which according to this convention abbreviates '$(A \supset B) \supset C$'. Other examples are

$$P \supset Q \supset P \supset P$$

which abbreviates

$$((P \supset Q) \supset P) \supset P$$

and

$$A \supset B \supset B \supset. C \supset A$$

which abbreviates

$$A \supset B \supset B \supset (C \supset A)$$

according to our second convention, and so in turn abbreviates

$$((A \supset B) \supset B) \supset (C \supset A).$$

Also

$$(P \supset (Q \supset R)) \supset ((P \supset Q) \supset (P \supset R))$$

can be written in abbreviated form as

$$(P \supset. Q \supset R) \supset. P \supset Q \supset. P \supset R.$$

Notice that in restoring parentheses to abbreviated names of formulas, '$\sim$' is always taken to apply to the smallest possible grouping: '$\sim A \supset B \supset B$' refers to $(\sim A \supset B) \supset B$, not to $\sim(A \supset B) \supset B$ or to $\sim((A \supset B) \supset B)$. Thus, our conventions do not allow us to abbreviate a name of a formula by dropping parentheses after a '$\sim$'.

7. As in S_0, proofs in $\mathbf{H}_s$ are generated from axioms by means of rules of inference. Our formulation of these axioms and rules, however, will be simpler than was the case with S_0 in that we will be able to get along without any rule of substitution. This gain in simplicity is offset by the fact that we must then allow *infinitely* many axioms, rather than only three. But in practice this turns out to be no drawback, since the axioms of $\mathbf{H}_s$ fall into three easily recognizable kinds. In fact, these kinds correspond to the axioms of S_0. Any formula of $\mathbf{H}_s$ having one of the following three forms qualifies as an axiom of $\mathbf{H}_s$.

AS1. $A \supset. B \supset A$
AS2. $(A \supset. B \supset C) \supset. A \supset B \supset. A \supset C$
AS3. $\sim A \supset \sim B \supset. B \supset A$

The metalinguistic expressions '$A \supset. B \supset A$', and so forth are called *axiom-schemes* (above, 'AS' stands for 'axiom-scheme'). Each of them determines infinitely many axioms of $\mathbf{H}_s$. For instance, $P_1 \supset. P_2 \supset P_1$, $P_1 \supset. P_1 \supset P_1$, and $\sim P_1 \supset. \sim(P_2 \supset P_3) \supset \sim P_1$ are all formulas of $\mathbf{H}_s$ which have the form $A \supset. B \supset A$, and so each of them is an instance of AS1 and hence an axiom of $\mathbf{H}_s$. And clearly, there is no limit to the number of axioms of this sort.

The only primitive rule of inference of $\mathbf{H}_s$ is *modus ponens*:

$$\frac{A \quad A \supset B}{B}.$$

Below we will furnish a metalinguistic proof that the rule of substitution is unnecessary. (See M31—i.e., metatheorem 31—in Section 13.) More precisely, what we will show is that the addition of substitution to $\mathbf{H}_s$ as a primitive rule of inference would not yield any new theorems.

The notion of a *proof* in $\mathbf{H}_s$ is sufficiently important to call for a full-scale definition. (Below, 'D1' stands for 'definition 1'.)

D1. A proof in $\mathbf{H}_s$ *is an array* $A_1, \ldots, A_n$ *of formulas of* $\mathbf{H}_s$ *such that every entry of the array is an axiom of* $\mathbf{H}_s$ *or follows from previous entries by modus ponens. In other words, for all numbers i such that* $1 \leq i \leq n$, A_i *is an axiom of* $\mathbf{H}_s$, *or there exist numbers j, k < i such that* A_j *is* $A_k \supset A_i$. *An array* $A_1, \ldots, A_n$ *which is a proof is said to be a proof of its last entry* A_n.

If it isn't at once clear to you that the second sentence of D1 is a reformulation of the condition given in the first sentence, you should pause to verify that this is so. Formulations of this kind are usually convenient when one wants to use a definition in proving some metatheorem, and we will make use of them below.

The following is an example of a proof in $\mathbf{H}_s$ of $P_1 \supset P_1$. Notice, by the way, that our abundance of axioms enables us to do this proof much more economically than, say, the proof of '$p \supset p$' in S_0.

$(P_1 \supset. P_1 \supset P_1 \supset P_1) \supset. P_1 \supset (P_1 \supset P_1) \supset. P_1 \supset P_1$
$P_1 \supset. P_1 \supset P_1 \supset P_1$
$(P_1 \supset. P_1 \supset P_1) \supset. P_1 \supset P_1$
$P_1 \supset. P_1 \supset P_1$
$P_1 \supset P_1$

(*vi*)

Notice that AS3 is not used in this proof.

D2. A formula A is provable in $\mathbf{H}_s$ *(or, equivalently, is a theorem of* $\mathbf{H}_s$*) in case there exists a proof of A in* $\mathbf{H}_s$*. It will be convenient to express this symbolically with the aid of a metalinguistic sign '*$\vdash$*': thus '*$\vdash_{\mathbf{H}_s} A$*' means that A is a theorem of* $\mathbf{H}_s$*. In this and the next two chapters, we need not write the subscript '*$\mathbf{H}_s$*' time after time, and will simply omit it.*

In view of vi, we know that $P_1 \supset P_1$ is a theorem of $\mathbf{H}_s$, i.e., that $\vdash P_1 \supset P_1$. We could dignify this fact with the name of *metatheorem* (or theorem of the metalanguage). But this would be foolishly specific, since any substitution instance of $P_1 \supset P_1$ can be proved by the same method as the one used in vi. This can be indicated directly, by means of a so-called *proof-scheme.*

$$
\begin{array}{l}
(A \supset . A \supset A \supset A) \supset . A \supset (A \supset A) \supset . A \supset A \\
A \supset . A \supset A \supset A \\
(A \supset . A \supset A) \supset . A \supset A \\
A \supset . A \supset A \\
A \supset A
\end{array}
$$

(*vii*)

Since the metavariable 'A' can stand for any formula of $\mathbf{H}_s$ whatsoever, the scheme vii shows that any formula of the sort $A \supset A$ is provable in $\mathbf{H}_s$, and so we have a general metatheorem.

M1. $\vdash A \supset A$.

8. In III.2, we introduced the notion of *hypothetical* reasoning, which proved to be a key idea in formulating $\mathbf{S}_\supset$ and other systems of natural deduction. This idea is also useful in developing the system $\mathbf{H}_s$. We can put it to work by speaking of *deductions* in $\mathbf{H}_s$—arrays like proofs except that hypotheses are allowed in them as well as axioms. To keep track of the hypotheses used in deductions, we will always speak of deductions *from sets of formulas*; in a deduction from a particular set of formulas, only members of that set can be used as hypotheses.

Since in discussing deductions we will be referring frequently to sets of formulas, we'll need some notation or other for talking about such sets. To a large extent, we can use the standard set-theoretic notation discussed in Chapter XIII, Sections 1 to 9. Thus, $\{P, P \supset Q\}$ is the set containing just the formulas P and $P \supset Q$, and $\{A \supset B \mid A$ and B are formulas of $\mathbf{H}_s\}$ is the set of formulas of $\mathbf{H}_s$ having the form $A \supset B$. But it's also convenient to have special metavariables taking sets of formulas as values; for this purpose, we'll use the capital Greek letters 'Γ', 'Δ', 'Θ', and 'Ξ'.

D3. Let Γ *be a set of formulas of* $\mathbf{H}_s$*. An array* $A_1, \ldots, A_n$ *of formulas of* $\mathbf{H}_s$ *is a deduction of* A_n *from hypotheses* Γ *in case for all i such that* $1 \leq i \leq n$*, (1)* A_i *is an axiom of* $\mathbf{H}_s$*, or (2)* A_i *is a member of* Γ*, or (3) for some* $j, k < i$*,* A_j *is* $A_k \supset A_i$*.*

As a simple example, consider the following deduction of P_1 from the set $\{\sim P_1 \supset \sim P_2, P_2\}$.

$$
\begin{array}{l}
\sim P_1 \supset \sim P_2 \\
\sim P_1 \supset \sim P_2 \supset . P_2 \supset P_1 \\
P_2 \supset P_1 \\
P_2 \\
P_1
\end{array}
$$

(*viii*)

Example viii shows that P_1 is *deducible* from the set $\{\sim P_1 \supset \sim P_2, P_2\}$ of formulas. Unlike provability, deducibility isn't a property of formulas; it's a relation that holds between sets of formulas and formulas. We will employ the same symbol '$\vdash$' that we used to stand for provability to express the relation of deducibility; '$\Gamma \vdash A$' will mean that A is deducible from the set Γ of formulas. It may seem awkward to give '$\vdash$' these two meanings, but we will show in M2 that provability can be regarded as a special case of deducibility.

D4. A formula A is deducible in $\mathbf{H}_s$ *from a set* Γ *of formulas of* $\mathbf{H}_s$ *in case there exists a deduction in* $\mathbf{H}_s$ *of A from* Γ*. We write '*$\Gamma \vdash_{\mathbf{H}_s} A$*' to indicate that A is deducible in* $\mathbf{H}_s$ *from* Γ*, and as in the case of provability we will omit the subscript in this and the next two chapters.*

The proof of M1 consisted in displaying a proof-scheme that was a generalization of the proof given in vi. Here, as well, we can generalize the deduction viii to obtain the following *deduction-scheme.*

$$
\begin{array}{l}
\sim A \supset \sim B \\
\sim A \supset \sim B \supset . B \supset A \\
B \supset A \\
B \\
A
\end{array}
$$

(*ix*)

Example ix shows that for all formulas A and B of $\mathbf{H}_s$, $\sim A \supset \sim B, B \vdash A$. (We will often omit curly brackets in our notation for deducibility. There is no danger of ambiguity arising from this practice.)

We now state and prove a metatheorem that shows that deducibility is a generalization of provability; i.e., provability amounts to deducibility from *no* hypotheses. This is so straightforward it hardly requires proof at all, but since it's one of our first metatheorems we will give it special treatment.

M2. $\vdash A$ *if and only if* $\varnothing \vdash A$.

PROOF. As is explained in XIII.7, $\varnothing$ is the empty set: the set that contains no members. Suppose first that $\vdash A$; this means that there exists a proof $B_1, \ldots, B_n$ of A. But since this array is a proof, no hypotheses are used in it and hence by D3 it is a deduction of A from $\varnothing$. But then there is such a deduction, and so $\varnothing \vdash A$. Conversely, suppose that $\varnothing \vdash A$; this means there is a deduction $B_1, \ldots, B_n$ of A from $\varnothing$. But since $\varnothing$ has no members, no hypotheses can be used in this deduction and hence it is a proof of A. Since there is such a proof, A is provable; i.e., $\vdash A$.

This metatheorem justifies a certain laxity of notation; below, we will use '$\vdash A$' interchangeably with '$\varnothing \vdash A$'.

Our next metatheorem expresses an important property of deducibility; the proof of this metatheorem depends on the fact that all deductions in $\mathbf{H_s}$ are finite arrays of formulas. This follows from D1, which says that every proof in $\mathbf{H_s}$ is an array $A_1, \ldots, A_n$ of formulas. Thus every proof will only have some number n of steps and so will be finite.

M3. $\Gamma \vdash A$ *if and only if for some finite subset* Δ *of* Γ, $\Delta \vdash A$.

PROOF. Suppose first that $\Gamma \vdash A$; then there is a deduction $B_1, \ldots, B_n$ of A from Γ. Now, only finitely many members of Γ can be used in this deduction. In fact, let Γ' be $\Gamma \cap \{B_1, \ldots, B_n\}$; the array $B_1, \ldots, B_n$ is a deduction of A from Γ', and so $\Gamma' \vdash A$. But Γ' is finite; hence, for some finite subset Δ of Γ, $\Delta \vdash A$. Conversely, suppose that A is deducible from some finite subset Δ of Γ; then there is a deduction $B_1, \ldots, B_n$ of A from Δ. But since every member of Δ is also a member of Γ, $B_1, \ldots, B_n$ is also a deduction of A from Γ, and so $\Gamma \vdash A$.

It may help to clarify the point of M3 if we remark that this metatheorem is trivial if Γ is finite. In that case, if $\Gamma \vdash A$ then Γ itself would be a finite subset of Γ such that $\Gamma \vdash A$. The case in which M3 tells us something is the one in which Γ is an *infinite* set of formulas, such as $\{P_1, P_2, P_3, \ldots\}$. Here, M3 ensures that anything deducible from Γ must be deducible from a finite part of Γ. For instance, it is true that $\{P_1, P_2, P_3, \ldots\} \vdash {\sim}P_1 \supset P_2$, and so M3 guarantees that there must be a finite subset Γ of $\{P_1, P_2, P_3, \ldots\}$ such that $\Gamma \vdash {\sim}P_1 \supset P_2$. And indeed, $\{P_2\}$ is just such a set.

We will now list a number of further important properties of the deducibility relation $\vdash$; their proofs are not difficult, and are left as exercises. You may wish to try them before reading on.

M4. If $A \in \Gamma$, *then* $\Gamma \vdash A$.
M5. If $\Gamma \vdash A$, *then* $\Gamma \cup \Delta \vdash A$.
M6. If $\Gamma \vdash A$ *and* $\Delta \cup \{A\} \vdash B$, *then* $\Gamma \cup \Delta \vdash B$.
M7. If $\Gamma \vdash A \supset B$, *then* $\Gamma \cup \{A\} \vdash B$.

When $\vdash$ is thought of as deducibility with regard to informal reasoning, all of these four characteristics make sense. We have in mind situations where we single out a given set of sentences (e.g., the set of sentences believed by a person at a given time, or the set of sentences in *Paradise Lost*), and are interested in those sentences that can be deduced from the sentences of this set, taken together. The analogue of this for a set Γ of formulas of $\mathbf{H_s}$ would be those formulas A such that $\Gamma \vdash A$.

Now, it is reasonable to suppose that (M4) every sentence is deducible from any set of sentences of which it is a member, and that (M5) if every sentence in some set is contained in some larger set, then every sentence deducible from the smaller set is deducible from the larger one. Also (M6) if from one set we can deduce a sentence that together with another set yields a second sentence, then the two sets together should yield both sentences—and, in particular, the second. This is actually a general form of *modus ponens*. Finally (M7), if an implication is deducible from a set, then that set together with the antecedent of the implication should yield the consequent of the implication.

If nothing else, the above paragraph is a material lesson in the usefulness of specialized metalinguistic notation, as opposed to plain prose. Hardly more is said in the above paragraph than in the four lines constituting the statement of M4 to M7, and yet this paragraph is not wordy. There is just no clear way of saying this more briefly, without resorting to some notation or other.

9. This section is devoted to the explanation, proof, and application of a metatheorem about deductions, known as the *deduction theorem*. It will be our first metatheorem calling for a full-scale proof.

In the systems of Chapters III and IV, a deduction of a conclusion from some hypothesis is used to establish the corresponding implication; the hypothesis is discharged, and the implication asserted categorically. We discussed this in detail back in Chapter III. Now, if $\mathbf{H}_s$ is to be at all adequate as a characterization of such reasoning, it should have some property corresponding to the rule of implication introduction. We might formulate this property by saying that if $A \vdash B$, then $\vdash A \supset B$: if there is a deduction of B from A, then there is a proof of $A \supset B$. But a more general formulation would allow for other hypotheses besides A—say, a set Γ of them. We arrive in this way at the conjecture that if there is a deduction of B from the set consisting of A together with all the members of Γ, then there is a deduction of $A \supset B$ from Γ. We will state this conjecture as a metatheorem, and then proceed to prove it. A careful analysis of the proof, by the way, would show that all that is needed for this result is the presence of axiom-schemes 1 and 2, and the rule of *modus ponens*.

M8. (*Deduction theorem for* $\mathbf{H}_s$). *If* $\Gamma \cup \{A\} \vdash B$, *then* $\Gamma \vdash A \supset B$.

PROOF. We must show that if there is a deduction of B from $\Gamma \cup \{A\}$, then there is a deduction of $A \supset B$ from Γ. To accomplish this, we will present general instructions for transforming a deduction $C_1, \ldots, C_n$ of B from $\Gamma \cup \{A\}$ into a deduction of $A \supset B$ from Γ.

Let $C_1, \ldots, C_n$ be a deduction of B from $\Gamma \cup \{A\}$. (Note that under these circumstances C_n is the same formula as B.) In the first step of our transformation of $C_1, \ldots, C_n$ into a deduction of $A \supset B$ from Γ, we replace every entry C_i of the array by $A \supset C_i$, thus obtaining the array $A \supset C_1, \ldots, A \supset C_n$. This second array is no longer a deduction, but we now proceed systematically through it, inserting steps according to the set of instructions given below. The resulting array will be a deduction.

Since $C_1, \ldots, C_n$ is a deduction from $\Gamma \cup \{A\}$, we know that for every entry C_i, either (1) C_i is an axiom of $\mathbf{H}_s$; or (2) C_i is a member of $\Gamma \cup \{A\}$; or (3) for some $j, k < i$, C_j is $C_k \supset C_i$. Our instructions for inserting steps in the array $A \supset C_1, \ldots, A \supset C_n$ are divided into parts according to these three cases, as follows.

1. If C_i is an axiom of $\mathbf{H}_s$, insert the following steps before $A \supset C_i$.

$$C_i$$
$$C_i \supset . A \supset C_i$$

2. If C_i is a member of $\Gamma \cup \{A\}$, there are two subcases.

2.1. If C_i is a member of Γ, insert the following steps before $A \supset C_i$.

$$C_i$$
$$C_i \supset . A \supset C_i$$

2.2. If C_i is A, insert the following steps before $A \supset C_i$ (i.e., before $A \supset A$).

$$(A \supset . A \supset A \supset A) \supset . (A \supset . A \supset A) \supset . A \supset A$$
$$A \supset . A \supset A \supset A$$
$$(A \supset . A \supset A) \supset . A \supset A$$
$$A \supset . A \supset A$$

(Notice that in this case we are merely building a copy of example vii.)

3. If for some $j, k < i$, C_j is $C_k \supset C_i$, insert the following steps before $A \supset C_i$.

$$(A \supset . C_k \supset C_i) \supset . A \supset C_k \supset . A \supset C_i$$
$$A \supset C_k \supset . A \supset C_i$$

When we have finished going through $A \supset C_1, \ldots, A \supset C_n$ inserting steps according to the above rules we will obtain a longer array, say $D_1, \ldots, D_m$. Now, we claim that $D_1, \ldots, D_m$ is a deduction of $A \supset B$ from Γ. First, note that D_m is the same formula as $A \supset B$, since D_m is $A \supset C_n$ and C_n is B. Thus, to verify that $D_1, \ldots, D_m$ is a deduction of $A \supset B$ from Γ, we must go back and check that the conditions of D3 are met for each entry D_h of this array; this is accomplished by systematic inspection of cases 1 to 3 above. First, it is clear that if D_h is one of the entries inserted in applying the rules, then D_h is either an axiom of $\mathbf{H}_s$, or a member of Γ, or a consequence by *modus ponens* of previous entries in the array $D_1, \ldots, D_m$. For example, if D_h is inserted according to the instructions given in case 1, then D_h is either an axiom C_i, or else an instance $C_i \supset . A \supset C_i$ of AS1. And similarly in the remaining cases.

This leaves the entries D_h which were not inserted; here, D_h is $A \supset C_i$. Under these circumstances, we know that C_i is an axiom of $\mathbf{H}_s$ (case 1), or a member of $\Gamma \cup \{A\}$ (cases 2.1 and 2.2), or a consequence by *modus ponens* of two previous formulas, $C_k \supset C_i$ and C_k (case 3).

In cases 1 and 2.1, D_h (i.e., $A \supset C_i$) will follow by *modus ponens* from the two preceding steps inserted. In case 2.2, D_h will be $A \supset A$, which again follows by *modus ponens* from the two preceding steps. (Notice that since no hypotheses are inserted here, this case is engineered in such a way that A is discharged as a hypothesis.) And in case 3, D_h will be $A \supset C_i$, where $A \supset . C_k \supset C_i$ and $A \supset C_k$ occur previously in the array $D_1, \ldots, D_m$. But in this case D_h is a consequence by *modus ponens* of the entries $A \supset C_k$ and $A \supset C_k \supset . A \supset C_i$, both of which occur previously.

Thus, the array $D_1 \ldots, D_m$ meets the conditions of D3, and *is* in fact a deduction of D_m from Γ; but D_m is the desired formula, $A \supset B$. Thus, our proof of the metatheorem is complete.

If we are interested in what sorts of things are provable in $\mathbf{H_s}$, M8 is a very useful piece of information. Obviously, it's one thing to assure oneself that a certain action (like driving from Philadelphia to St. Louis in a day) can be done, and another thing to actually do it. Similarly, it's one thing to show that a formula is provable and another to actually produce a proof of the formula. Up to this point, the only way we had of telling that a theorem of $\mathbf{H_s}$ is provable was by actually producing a proof (or rather, a metalinguistic scheme standing for a proof), but M8 can assure us of the existence of such proofs without our having to go to this sort of trouble.

As an example, consider any formula of the sort $A \supset B \supset. B \supset C \supset. A \supset C$. M8 tells us that if $A \supset B \vdash B \supset C \supset. A \supset C$, then $\vdash A \supset B \supset. B \supset C \supset. A \supset C$. We are thus led to ask whether in fact $A \supset B \vdash B \supset C \supset. A \supset C$ and successive steps of this sort lead eventually to the question, whether $A \supset B, B \supset C, A \vdash C$.

But it's easy to see how the formula C can be obtained from hypotheses $A \supset B$, $B \supset C$, and A by means of the rules of $\mathbf{H_s}$. Such a deduction as the following does the job. For convenience, reasons are supplied in this example.

1	$A \supset B$	hyp
2	A	hyp
3	B	1, 2, m p
4	$B \supset C$	hyp
5	C	3, 4, m p

(*x*)

This shows that $A \supset B, B \supset C, A \vdash C$, and successive applications of M8 yield the result that $\vdash A \supset B \supset. B \supset C \supset. A \supset C$, as desired. So without ever having seen a proof of $A \supset B \supset. B \supset C \supset. A \supset C$, we have proved the following metatheorem.

M9. $\vdash A \supset B \supset. B \supset C \supset. A \supset C$.

Returning now to M8, it's important to observe that the argument given in the proof of this metatheorem yields a uniform method or recipe for constructing a deduction of B from $\Gamma \cup \{A\}$. If we wished, for instance, we could systematically transform any deduction of the sort given in example x into a proof of $A \supset B \supset. B \supset C \supset. A \supset C$ by means of this method, although, of course, the details of this transformation would be very tedious. But as an illustration of the general strategy, here is a deduction of $A \supset C$ from $\{A \supset B, B \supset C\}$, obtained from example x according to the instructions we gave in proving M8.

	1	$A \supset B$	hyp
	2	$A \supset B \supset. A \supset. A \supset B$	AS1
$A \supset. A \supset B$	3	$A \supset. A \supset B$	1, 2, m p
	4	$(A \supset. A \supset A \supset A) \supset. (A \supset. A \supset A) \supset. A \supset A)$	AS2
	5	$A \supset. A \supset A \supset A$	AS1
	6	$(A \supset. A \supset A) \supset. A \supset A$	4, 5, m p
	7	$A \supset. A \supset A$	AS1
$A \supset A$	8	$A \supset A$	6, 7, m p
	9	$(A \supset. A \supset B) \supset. A \supset A \supset. A \supset B$	AS2
	10	$A \supset A \supset. A \supset B$	3, 9, m p
$A \supset B$	11	$A \supset B$	8, 10, m p
	12	$B \supset C$	hyp
	13	$B \supset C \supset. A \supset. B \supset C$	AS1
$A \supset. B \supset C$	14	$A \supset. B \supset C$	12, 13, m p
	15	$(A \supset. B \supset C) \supset. A \supset B \supset. A \supset C$	AS2
	16	$A \supset B \supset. A \supset C$	14, 15, m p
$A \supset C$	17	$A \supset C$	11, 16, m p
(*xi*)		(*xii*)	

The column to the left, xi, is obtained from x by prefixing A as in the proof of M8, and xii is obtained from xi by inserting steps according to the instructions given in that proof. Clearly, xii is not the most economical deduction of $A \supset C$ from $\{A \supset B, B \supset C\}$. Step 11, for instance, is already a hypothesis, but it is obtained in xii by *modus ponens*. But what matters is that it *is* a deduction, and the mechanical method of M8 will always produce such a deduction.

If, by the way, you had any difficulty in following the proof of M8, examples xi and xii may help to make the argument clearer.

Before going on to other matters, we will record a useful corollary of M8, leaving its proof to you.

M10. If $A_1, \ldots, A_n \vdash B$, *then* $\vdash A_1 \supset. \cdots \supset. A_n \supset B$.

10. This section is devoted to extending our knowledge of the deductions that can be made in $\mathbf{H_s}$. We will have to wade through some tedious detail in obtaining the results of this section, but the metatheorems we have established above will be a great help to us.

M11. If $\Gamma \vdash A \supset B$ *and* $\Delta \vdash A$, *then* $\Gamma \cup \Delta \vdash B$.

PROOF. If there is a deduction of $A \supset B$ from Γ and one of A from Δ, the result of putting these two deductions together and following them by B will be a deduction of B from $\Gamma \cup \Delta$. (*Note:* there is another proof of this metatheorem, using M6 and M7.)

M12. If $\Gamma \cup \{\sim A\} \vdash \sim B$, *then* $\Gamma \cup \{B\} \vdash A$.

PROOF. If $\Gamma \cup \{\sim A\} \vdash \sim B$, then by M8 $\Gamma \vdash \sim A \supset \sim B$. But in view of AS3, $\vdash \sim A \supset \sim B \supset. B \supset A$; hence, by M11, $\Gamma \vdash B \supset A$. Thus, by M7, $\Gamma \cup \{B\} \vdash A$.

In proving the following metatheorems we will use a more tabular presentation that should help to make the proofs easier to follow.

M13. $\sim A, A \vdash B$.

PROOF.	1. $\sim A, \sim B \vdash \sim A$	M4
	2. $\sim A, A \vdash B$	1, M12

M14. $\sim\sim A \vdash A$.

PROOF.	1. $\sim\sim A, \sim A \vdash \sim\sim\sim A$	M13
	2. $\sim\sim A \vdash A$	1, M12

In this last demonstration, step 2 may require some explanation. Written out in full, step 1 is '$\{\sim\sim A, \sim A\} \vdash \sim\sim\sim A$'; using M12, we then see from step 1 that $\{\sim\sim A, \sim\sim A\} \vdash A$. But $\{\sim\sim A, \sim\sim A\}$ (i.e., the set containing just $\sim\sim A$ and $\sim\sim A$) is simply the set $\{\sim\sim A\}$. Thus, we have step 2.

M15. $\sim\sim A, \sim\sim(A \supset \sim A) \vdash \sim\sim\sim A$.

PROOF.	1. $\sim\sim A \vdash A$	M14
	2. $\sim\sim(A \supset \sim A) \vdash A \supset \sim A$	M14
	3. $\sim\sim A, \sim\sim(A \supset \sim A) \vdash \sim A$	1, 2, M11
	4. $\sim A, A \vdash \sim\sim\sim A$	M13
	5. $\sim\sim A, \sim A \vdash \sim\sim\sim A$	1, 4, M6
	6. $\sim\sim A, \sim\sim(A \supset \sim A) \vdash \sim\sim\sim A$	3, 5, M6

M16. $A \supset B, A \supset \sim B \vdash A \supset \sim A$.

PROOF.	1. $A \supset B, A \vdash B$	*modus ponens*
	2. $A \supset \sim B, A \vdash \sim B$	*modus ponens*
	3. $\sim B, B \vdash \sim A$	M13
	4. $A \supset B, A, \sim B \vdash \sim A$	1, 3, M6
	5. $A \supset B, A \supset \sim B, A \vdash \sim A$	2, 4, M6
	6. $A \supset B, A \supset \sim B \vdash A \supset \sim A$	5, M8

M17. $A \supset \sim A \vdash \sim A$.

PROOF.	1. $\sim\sim A, \sim\sim(A \supset \sim A) \vdash \sim\sim\sim A$	M15
	2. $\sim\sim A \vdash \sim(A \supset \sim A)$	1, M12
	3. $A \supset \sim A \vdash \sim A$	2, M12

M18. $A \supset B, A \supset \sim B \vdash \sim A$.

PROOF.	1. $A \supset B, A \supset \sim B \vdash A \supset \sim A$	M16
	2. $A \supset \sim A \vdash \sim A$	M17
	3. $A \supset B, A \supset \sim B \vdash \sim A$	1, 2, M6

M19. If $\Gamma \vdash \sim\sim A$, *then* $\Gamma \vdash A$.

PROOF.	1. $\Gamma \vdash \sim\sim A$	assumption
	2. $\sim\sim A \vdash A$	M14
	3. $\Gamma \vdash A$	1, 2, M6

M20. If $\Gamma \cup \{A\} \vdash B$ *and* $\Gamma \cup \{A\} \vdash \sim B$, *then* $\Gamma \vdash \sim A$.

PROOF.	1. $\Gamma \cup \{A\} \vdash B$	assumption
	2. $\Gamma \vdash A \supset B$	1, M8
	3. $A \supset B, A \supset \sim B \vdash \sim A$	M18
	4. $\Gamma \cup \{A \supset \sim B\} \vdash \sim A$	2, 3, M6
	5. $\Gamma \cup \{A\} \vdash \sim B$	assumption
	6. $\Gamma \vdash A \supset \sim B$	5, M8
	7. $\Gamma \vdash \sim A$	4, 6, M6

11. Back in Section 2 when we defined the formulas of $\mathbf{H}_s$, it may have seemed to you that our definition narrowed somewhat the logical horizons that had opened in Chapter IV. The system $\mathbf{H}_s$ has only two connectives—negation and implication—as compared with the five of $\mathbf{S}_s$. But as it turns out, the poverty of $\mathbf{H}_s$ is only apparent. In fact, we will show in this section that all the connectives that figure in $\mathbf{S}_s$ can be obtained in $\mathbf{H}_s$ by means of *definitions*.

Logicians differ in their policies concerning definition; the one to which we will subscribe in defining new connectives of $\mathbf{H}_s$ is perhaps the simplest of these. Rather than regarding defined connectives as new symbols introduced in the object-language by rules of definition, we regard them as abbreviative conventions made in our metalanguage. From our point of view, then, definitions make no change in the formulas of $\mathbf{H}_s$, only in our way of talking about them. In this regard they are just like the abbreviations for eliminating parentheses which we discussed in Section 6, above. What we will do in defining, say, the connective $\vee$ is to find a complex formula $f(A, B)$ of $\mathbf{H}_s$ depending on A and B, which will serve as a good definition of $\vee$. (Here, 'f' stands for an unspecified way of constructing a complex formula out of A and B; for instance, $A \supset . B \supset \sim A$ or $\sim(A \supset B)$.) Until we get to Chapter VI, we will not be able to give a really satisfactory account of what a good definition is. But we at least do know how disjunction behaves in $\mathbf{S_s}$, and the definition chosen below of $\vee$ does in fact behave in $\mathbf{H_s}$ just as disjunction behaves in $\mathbf{S_s}$.

In D5 and later definitions, '$=_{df}$' stands for 'is by definition'.

D5. '$A \vee B$' $=_{df}$ '$A \supset B \supset B$'.

According to D5, whenever an expression of our metalanguage contains a part such as '$A \vee B$', this part will be replaceable without change of meaning by '$A \supset B \supset B$'. Thus, $(A \vee B) \supset B \supset C$ is the formula $A \supset B \supset B \supset B \supset C$, and $(\sim A \vee B) \supset (A \vee (B \vee A))$ is the formula $\sim A \supset B \supset B \supset . A \supset (B \vee A) \supset (B \vee A)$, which in turn is $\sim A \supset B \supset B \supset . A \supset (B \supset A \supset A) \supset (B \supset A \supset A)$. By the same token, when we say that $\vdash A \vee \sim A$, we mean that $\vdash A \supset \sim A \supset \sim A$, which can be shown using M17 and M8; this, then, shows that $\vdash A \vee \sim A$.

We want to define conjunction and equivalence as well; this is done in the following two definitions.

D6. '$A \wedge B$' $=_{df}$ '$\sim(A \supset \sim B)$'.
D7. '$A \equiv B$' $=_{df}$ '$(A \supset B) \wedge (B \supset A)$'.

In order to see what a formula such as $(A \equiv B) \equiv C$ amounts to in terms of implication and negation, we must use both D7 and D6. D7 tells us that

$$(A \equiv B) \equiv C$$

is

$$((A \supset B) \wedge (B \supset A)) \equiv C$$

which is

$$(((A \supset B) \wedge (B \supset A)) \supset C) \wedge (C \supset ((A \supset B) \wedge (B \supset A))).$$

D6 now tells us that this in turn is

$$(\sim(A \supset B \supset \sim(B \supset A)) \supset C) \wedge (C \supset \sim(A \supset B \supset \sim(B \supset A))).$$

Finally, another application of D6 yields

$$\sim(\sim(A \supset B \supset \sim(B \supset A)) \supset C \supset \sim(C \supset \sim(A \supset B \supset \sim(B \supset A)))).$$

As this example shows, D5 to D7 allow us to talk about some rather complex formulas much more briefly, inasmuch as '$(A \equiv B) \equiv C$' is much shorter than its unabbreviated equivalent.

12. To a certain extent our choice of the above definitions was arbitrary, but we could not have chosen just any definitions. For instance, '$B \supset B \supset \sim A$' would be wholly inappropriate as a definition of '$A \vee B$'. (We take it for granted here that the symbol '$\vee$' is to have something to do with disjunction as it is ordinarily understood.) This means that our task isn't ended once we have laid down definitions D5 to D7. We must go on to show somehow that they are appropriate. In order to accomplish this we will prove in this section some metatheorems which show in effect that the connectives $\vee$, $\wedge$, and $\equiv$, as defined above, satisfy analogues in $\mathbf{H_s}$ of the introduction and elimination rules of $\mathbf{S_s}$. For instance, we will show that if $\Gamma \vdash A$ and $\Gamma \vdash B$, then $\Gamma \vdash A \wedge B$, and that if $\Gamma \vdash A \wedge B$, then $\Gamma \vdash A$ and $\Gamma \vdash B$. These correspond to the rules of conjunction introduction and conjunction elimination of $\mathbf{S_s}$. Our first metatheorems accomplish this for disjunction.

M21. If $\Gamma \vdash A$ *then* $\Gamma \vdash A \vee B$; *and if* $\Gamma \vdash B$ *then* $\Gamma \vdash A \vee B$.

PROOF.

1.	$\Gamma \vdash A$	assumption
2.	$A, A \supset B \vdash B$	*modus ponens*
3.	$\Gamma \cup \{A \supset B\} \vdash B$	1, 2, M6
4.	$\Gamma \vdash A \supset B \supset B$	3, M8
5.	$\Gamma \vdash A \vee B$	4, D5
1.	$\Gamma \vdash B$	assumption
2.	$\Gamma \cup \{A \supset B\} \vdash B$	1, M5
3.	$\Gamma \vdash A \supset B \supset B$	2, M8
4.	$\Gamma \vdash A \vee B$	3, D5

M22. If $\Gamma \vdash A \vee B$ *and* $\Gamma \cup \{A\} \vdash C$ *and* $\Gamma \cup \{B\} \vdash C$, *then* $\Gamma \vdash C$.

PROOF.

1.	$\Gamma \cup \{A\} \vdash C$	assumption
2.	$\Gamma \cup \{\sim C, A\} \vdash C$	1, M5
3.	$\Gamma \cup \{\sim C, A\} \vdash \sim C$	M4
4.	$\Gamma \cup \{\sim C\} \vdash \sim A$	2, 3, M20
5.	$\sim A, A \vdash B$	M13
6.	$\sim A \vdash A \supset B$	5, M8
7.	$\Gamma \vdash A \vee B$	assumption
8.	$\Gamma \vdash A \supset B \supset B$	7, D5
9.	$\Gamma \cup \{\sim A\} \vdash B$	6, 8, M11
10.	$\Gamma \cup \{\sim C\} \vdash B$	4, 9, M6
11.	$\Gamma \cup \{B\} \vdash C$	assumption
12.	$\Gamma \cup \{\sim C\} \vdash C$	10, 11, M6
13.	$\Gamma \cup \{\sim C\} \vdash \sim C$	M4
14.	$\Gamma \vdash \sim\sim C$	12, 13, M20
15.	$\Gamma \vdash C$	14, M19

M23. If $\Gamma \vdash A$ *and* $\Gamma \vdash B$, *then* $\Gamma \vdash A \wedge B$.

PROOF. 1. $\Gamma \vdash A$
2. $A, A \supset \sim B \vdash \sim B$
3. $\Gamma \cup \{A \supset \sim B\} \vdash \sim B$
4. $\Gamma \vdash B$
5. $\Gamma \cup \{A \supset \sim B\} \vdash B$
6. $\Gamma \vdash \sim(A \supset \sim B)$
7. $\Gamma \vdash A \wedge B$

Filling in reasons in the above proof is left as an exercise, as are the entire proofs of M24 to M26, below.

M24. If $\Gamma \vdash A \wedge B$, *then* $\Gamma \vdash A$ *and* $\Gamma \vdash B$.
M25. If $\Gamma \cup \{A\} \vdash B$ *and* $\Gamma \cup \{B\} \vdash A$, *then* $\Gamma \vdash A \equiv B$.
M26. If $\Gamma \vdash A \equiv B$ *and* $\Gamma \vdash A$, *then* $\Gamma \vdash B$; *and if* $\Gamma \vdash A \equiv B$ *and* $\Gamma \vdash B$, *then* $\Gamma \vdash A$.

13. Among the more important characteristics of $\mathbf{H_s}$ which have not yet been established are admissible rules of *replacement of equals by equals* and of *substitution*. It is the business of this section to demonstrate that $\mathbf{H_s}$ has these characteristics.

An error that beginners sometimes make in working with the system $\mathbf{S_s}$ illustrates what we mean by 'replacement by equals'. Given a step in a derivation—say, $B \vee \sim\sim A$—containing a part $\sim\sim A$, they will proceed to infer $B \vee A$, giving in justification of this step the rule of negation elimination. This is a mistake, because the rule of negation elimination applies only when the *entire premiss* has the form $\sim\sim C$; but in this case, the premiss is a disjunction. The correct way to obtain $B \vee A$ from $B \vee \sim\sim A$ in $\mathbf{S_s}$ is by disjunction elimination, as follows.

1	$B \vee \sim\sim A$	hyp
2	B	hyp
3	$B \vee A$	2, dis int
4	$\sim\sim A$	hyp
5	A	4, neg elim
6	$B \vee A$	5, dis int
7	$B \vee A$	1, 2–3, 4–6, dis elim

Now, perhaps the reason why inferring $B \vee A$ directly from $B \vee \sim\sim A$ is an attractive mistake is that we know A and $\sim\sim A$ to be provably equivalent in $\mathbf{S_s}$: $A \equiv \sim\sim A$ is categorically derivable in that system. And under these circumstances it's natural to feel that the formulas A and $\sim\sim A$ should be interchangeable without affecting derivability. The mistake comes only when one supposes that this is a *primitive* rather than a *derived* rule of $\mathbf{S_s}$, passing directly to the conclusion $B \vee A$ without filling in the intervening steps.

Applying these ideas to $\mathbf{H_s}$, we arrive at the thought that if $\vdash A \equiv B$ and $\vdash C$, then any formula gotten by replacing A by B in C should also be provable in $\mathbf{H_s}$. For instance, if $\vdash A \equiv \sim\sim A$ and $\vdash D \supset A$, then $\vdash D \supset \sim\sim A$.

Before trying to show that this is so, it would be a good idea to obtain a clearer formulation of what is meant by 'a result of replacing A by B in C'. Now, when we say that A is in C, we mean that the formula A is a constituent or *subformula* of C, as, for instance, A is a subformula of $B \supset \sim A$. One way of defining this notion is to say that A is a subformula of C if A is a formula and is to be found among those consecutive strings of symbols which are parts of C. But a more useful characterization is one that employs the inductive technique that was used in the first place to define the formulas of $\mathbf{H_s}$.

D8. 1. *A is a subformula of A.*
2. *If* $B \supset C$ *is a subformula of A, then both B and C are subformulas of A.*
3. *If* $\sim B$ *is a subformula of A, then B is a subformula of A.*

A string of symbols of $\mathbf{H_s}$ qualifies as a subformula of A only if it can be shown to be a subformula of A by repeated applications of 1, 2, and 3. Note, by the way, that according to D8 every formula is a subformula of itself.

Let's take an example to illustrate how D8 works. Referring back to the column of formulas on p. 55, above, we'll use D8 to show that $(Q \supset Q)$ is a subformula of $\sim\sim(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$. First, clause 1 of D8 tells us that $\sim\sim(\sim P \supset \sim((Q \supset Q) \supset (\sim R \supset P)))$ (i.e., step 12) is a subformula of itself. Clause 3 then shows that step 11 is a subformula of step 12, and again that step 10 is a subformula of step 12. Clause 2 shows that step 8 is a subformula of step 12, and clause 3 shows that step 7 is a subformula of step 12. Finally, clause 2 shows that step 6 (i.e., $(Q \supset Q)$) is a subformula of step 12.

This example should make it clear that D8 works just like the inductive definition of the formulas of $\mathbf{H_s}$, only backwards.

Similarly, we can give an inductive definition of the notion of a result of replacing A in C by B. Here, it's important to realize that there can be many such results, since A can occur as a subformula at more than one place in C.

Thus, all three of the formulas $\sim(B \supset A \supset D)$, $\sim(A \supset B \supset D)$, and $\sim(B \supset B \supset D)$ are results of replacing A in $\sim(A \supset A \supset D)$ by B. A result of replacing A by B in C is accordingly a result of replacing any number of occurrences of A in C by B. We will understand this in such a way that zero is allowed as such a number. Since the result of replacing no occurrences of A in C by B is just C, C itself is always a result of replacing A in C by B. This allows for *vacuous* replacements, and these permit us to speak meaningfully of the result (namely, C) of replacing A in C by B even when A is not a subformula of C. The definition is as follows.

D9. 1. *If A is (the same formula as) C, then both C and B are results of replacing A by B in C.*

2. *If A is not (the same formula as) $C \supset D$, then if C' is a result of replacing A by B in C, and D' a result of replacing A by B in D, then $C' \supset D'$ is a result of replacing A by B in $C \supset D$.*
3. *If A is not (the same formula as) $\sim C$, then if C' is a result of replacing A by B in C, then $\sim C'$ is a result of replacing A by B in $\sim C$.*
4. *If A is not (the same formula as) P, then P is a result of replacing A by B in P.*

A string of symbols of $\mathbf{H_s}$ qualifies as a result of replacing A by B in C only if it can be shown to be such a result by repeated applications of clauses 1, 2, 3, and 4 of D9.

Armed with D9, we can return to the problem of showing that if $\vdash A \equiv B$ and $\vdash C$ and C' is a result of replacing A by B in C, then $\vdash C'$. It frequently happens that when one wishes to show something, the easiest and most straightforward way to do it is to prove something stronger and then obtain the desired result as a corollary. In the present case, what we will first show is M29. M27 and M28 are two minor metatheorems required in the demonstration of M29; their proofs are left as exercises.

M27. $A \equiv B, C \equiv D \vdash (A \supset C) \equiv (B \supset D)$.
M28. $A \equiv B \vdash \sim A \equiv \sim B$.

M29. For all formulas C of $\mathbf{H_s}$, if C' is any result of replacing A by B in C, then $A \equiv B \vdash C \equiv C'$.

PROOF. Our proof proceeds by induction on the complexity of C. This means that we will first establish that if C is a sentence parameter, then $A \equiv B \vdash C \equiv C'$ for all results C' of replacing A by B in C. Then we will let C be an arbitrary complex formula and assume, for all formulas D shorter than C, that $A \equiv B \vdash D \equiv D'$ for any result D' of replacing A by B in D; we will then show under this assumption that $A \equiv B \vdash C \equiv C'$ for any result C' of replacing A by B in C. The theorem will then follow by the principle of induction. (For discussion of this principle, see Chapter XIV.)

First, suppose that C is a sentence parameter P. Then by D9, either (1) C' is P or (2) A is P and C' is B. In case 1, since by M4 $P \vdash P$, we have $\vdash P \equiv P$ by M25. Hence, by M5, $A \equiv B \vdash P \equiv P$; i.e., $A \equiv B \vdash C \equiv C'$. In case 2, by M4, $P \equiv B \vdash P \equiv B$; i.e., $A \equiv B \vdash C \equiv C'$. This completes the *basis step* of the induction.

Now, for the *inductive step*, assume that C is not a sentence parameter and make the following hypothesis of induction: for all D shorter than C, $A \equiv B \vdash D \equiv D'$ for any result D' of replacing A by B in D. We know that either (1) C is an implication $C_1 \supset C_2$ or (2) C is a negation $\sim D$. In case 1, D9 guarantees that either (1.1) C' is C; or (1.2) A is C and C' is B; or (1.3) C' is $C_1' \supset C_2'$, where C_1' is a result of replacing A by B in C_1 and C_2' a result of replacing A by B in C_2. Cases 1.1 and 1.2 are just like cases 1 and 2 of the basis step. In case 1.3, we know that C_1 and C_2 are both shorter than C; hence, the hypothesis of induction ensures that $A \equiv B \vdash C_1 \equiv C_1'$ and $A \equiv B \vdash C_2 \equiv C_2'$. Now, by M27, $C_1 \equiv C_1', C_2 \equiv C_2' \vdash (C_1 \supset C_2) \equiv (C_1' \supset C_2')$; i.e., $C_1 \equiv C_1', C_2 \equiv C_2', \vdash C \equiv C'$. Using M6, then, we first obtain $A \equiv B, C_2 \equiv C_2' \vdash C \equiv C'$ and then $A \equiv B \vdash C \equiv C'$, as desired.

In case 2, D9 guarantees that either (2.1) C' is C; or (2.2) A is C and C' is B; or (2.3) C' is $\sim D'$, where D' is a result of replacing A by B in D. Again, cases 2.1 and 2.2 are like cases 1 and 2 of the basis step. In case 2.3, we know that D is shorter than C; hence, the hypothesis of induction ensures that $A \equiv B \vdash D \equiv D'$. Now, by M28, $D \equiv D' \vdash \sim D \equiv \sim D'$; i.e., $D \equiv D' \vdash C \equiv C'$. Using M6, we then have $A \equiv B \vdash C \equiv C'$, as desired.

This completes the inductive step, and so M29 is proved.

M30. If $\Gamma \vdash C$ and $\Gamma \vdash A \equiv B$, and C' is a result of replacing A by B in C, then $\Gamma \vdash C'$.

Given M29, the proof of M30 is straightforward; it is left as an exercise.

Although we've been acquainted with the notion of substitution ever since Chapter I, we will now have to demonstrate a metatheorem that involves this concept. For this reason it is necessary to write down a rigorous definition of the result of substituting A for P in B. Again, the definition is inductive with respect to the complexity of B.

D10. 1. *If B is a sentence parameter, the result of substituting A for P in B is A in case P is the same sentence parameter as B, and is B otherwise.*
2. *If B is $C \supset D$, the result of substituting A for P in B is $C' \supset D'$, where C' is the result of substituting A for P in C and D' the result of substituting A for P in D.*
3. *If B is $\sim C$, the result of substituting A for P in B is $\sim C'$ where C' is the result of substituting A for P in C.*

A string of symbols of $\mathbf{H_s}$ qualifies as the result of substituting A for P in B only if it can be shown to be so by applications of rules 1 to 3.

You will recall that the rule of substitution permits one to infer A' from A, where for some sentence parameter P, A' is the result of substituting B for P in A. We will now show that this rule, when applied to theorems of $\mathbf{H_s}$, yields only theorems of $\mathbf{H_s}$.

M31. If $\vdash A$, and A' is the result of substituting B for P in A, then $\vdash A'$.

PROOF. Suppose that $\vdash A$; then there is a proof of A. We will use induction on the number of steps in a proof of A with minimal length.

First, if this length is 1, A is an axiom of $\mathbf{H_s}$. But in that case, clearly A' is also an axiom of the same sort. For example, if A has the form $C \supset. D \supset C$, then A' has the form $C' \supset. D' \supset C'$. This completes the basis step.

Assume the following hypothesis of induction: for all formulas C, if C has a proof with fewer than n steps, and C' is the result of substituting B for P in C, then $\vdash C'$. Let A have a proof of length n, and let A' be the result of substituting B for P in A. If A is an axiom of $\mathbf{H_s}$, then by the argument of the basis case, $\vdash A'$. If A follows by *modus ponens* from two previous steps D and $D \supset A$, then both these steps have proofs of length less than n, and so by the hypothesis of induction $\vdash D'$ and $\vdash (D \supset A)'$, where D' and $(D \supset A)'$ are the results, respectively, of substituting B for P in D and in $D \supset A$. But by clause 2 of D10, $(D \supset A)'$ is $D' \supset A'$. Hence, by *modus ponens*, $\vdash A'$, as desired.

We have presented this proof in such detail here because it's our first example of an induction on length of proof. This is a commonly used technique, and a working acquaintance with it is desirable. In practice, there is no reason to go to the trouble of mentioning explicitly the length of proofs. One can look instead at such inductions as resting on the following principle: if all the axioms of a system have a certain property, and if whenever the premisses of a primitive rule of inference of the system have the property then the conclusion does, then every theorem of the system has the property. In the above argument, then, all we really needed to show was that all substitution instances of axioms $\mathbf{H_s}$ are theorems of $\mathbf{H_s}$, and that if all substitution instances of D and of $D \supset A$ are theorems of $\mathbf{H_s}$, then all substitution instances of A are theorems of $\mathbf{H_s}$.

Finally there is still another, more pictorial way to prove M31. This is to notice that any result of substituting B for P throughout a proof in $\mathbf{H_s}$ is still a proof in $\mathbf{H_s}$.

14. M31 shows that the addition of substitution to $\mathbf{H_s}$ as a primitive rule would not increase the theorems of the system. This feature of substitution is sometimes expressed by saying that it is an *admissible* rule of $\mathbf{H_s}$. (See E14, below.)

It is important to bear in mind, however, that substitution does *not* correspond to deductions which can be carried out in $\mathbf{H_s}$. This is shown by the fact that, whereas the inference

$$\frac{P}{\sim P}$$

is an instance of substitution, we certainly do not want $\sim P$ to be *deducible* from P; we want to reject the claim that $P \vdash \sim P$.

One reason for hoping that $\sim P$ isn't deducible from $\{P\}$ in $\mathbf{H_s}$ is that $\sim P$ is clearly not a logical consequence of P, and $\mathbf{H_s}$ is supposed to be a system of logic. We have to take it on faith that it isn't the case that $P \vdash \sim P$, since we don't yet have a way of proving that this is so. But at least we can use metatheorems we have already proved to show that if $\sim P$ is derivable in $\mathbf{H_s}$ from $\{P\}$, then any formula whatsoever is provable in $\mathbf{H_s}$. For, suppose that $P \vdash \sim P$. By M8 we would have $\vdash P \supset \sim P$, and so by M31, $\vdash P \supset P \supset \sim(P \supset P)$. Since $\vdash P \supset P$, we would have both $\vdash P \supset P$ and $\vdash \sim(P \supset P)$. But in view of M13, $P \supset P$, $\sim(P \supset P) \vdash A$ for all formulas A of $\mathbf{H_s}$. So it follows by M6 that $\vdash A$ for all formulas A of $\mathbf{H_s}$. On the basis of this argument, we know at least that something would be very wrong indeed with $\mathbf{H_s}$ if it were the case that $P \vdash \sim P$.

Contrast this situation with the rule of replacement: from $A \equiv B$ and C to infer C', where C' is any result of replacing A by B in C. M30 shows that the rule of replacement is admissible in $\mathbf{H_s}$; the rule does not lead us out of the theorems of $\mathbf{H_s}$. But we also showed something more than this: M29 guarantees that $A \equiv B$, $C \vdash C'$, where C' is any result of replacing A by B in C. (See E14(d).)

In contrast with the rule of substitution, then, replacement turns out to be sanctioned by deductions of $\mathbf{H_s}$ as a correct form of inference. That is, the conclusion of any instance of the rule of replacement is deducible in $\mathbf{H_s}$ from the premisses of that instance. Rules of this sort are said to be *derived rules* of $\mathbf{H_s}$; substitution is an admissible but not a derived rule of $\mathbf{H_s}$, whereas replacement is both an admissible and a derived rule of $\mathbf{H_s}$.

Since it's important to grasp the difference between admissible and derived rules, let's discuss this difference between substitution and replacement from a different point of view. The only reason why substitution is an admissible rule of $\mathbf{H_s}$ is that $\mathbf{H_s}$ is a system of logic. In such a system, the sentence parameters are allowed to stand for any indicative sentence whatsoever, including logically complex sentences. It follows that any substitution instance of a logical truth is also a logical truth. For instance, let B be any formula and P be any sentence parameter. If A is a logical truth involving P, and A' is the result of substituting B for P in A, then A' can't fail to be a logical truth, because this failure of a special case of A would show A not to be a logical truth. Since $\mathbf{H_s}$ is intended to capture truths of logic as its theorems, it is reasonable to expect that substitution is an admissible rule of $\mathbf{H_s}$. And this expectation is borne out by M31.

Although the above reasoning shows why substitution should be an admissible rule of $\mathbf{H_s}$, this reasoning does not carry over to deducibility. That is, it does not lead us to expect that if A' is the result of substituting B for P in A, then $A \vdash A'$. For instance, there is no reason to think that those formulas deducible from $\{P\}$ are all truths of logic, and hence no reason to think that the rule of substitution will yield only such formulas when applied to them.

Let's recapitulate. The rule of substitution works when applied to theorems of $\mathbf{H_s}$, yielding conclusions which in this case also are theorems of $\mathbf{H_s}$. (This is what is meant by saying it is admissible.) But this is no guarantee that what it does to nontheorems corresponds to any reasonable inference. In the case of substitution, this is just what happens with premisses such as P; it is not the case that $P \vdash {\sim}P$. Thus, substitution is an admissible but not a derived rule of $\mathbf{H_s}$. On the other hand, replacement is not only an admissible rule of $\mathbf{H_s}$, it is a derived rule. Most of the rules that we have discussed in connection with $\mathbf{H_s}$ are likewise derived rules of that system. For instance, since $A \supset B$, $A \vdash B$, *modus ponens* is a derived rule of $\mathbf{H_s}$. For a more precise account of the notions discussed in this section, see E14 below.

15. Our treatment of deducibility in $\mathbf{H_s}$ enables us to give an account of the notion of *consistency*. First, it's important to realize that consistency is a quality that is properly ascribed to *sets* of sentences rather than to sentences themselves. When we tell someone that what he has said is inconsistent (i.e., when we deny that what he's said is consistent) we usually don't mean that it is inconsistent when taken by itself. Instead we mean that it's inconsistent when taken together with what he has said previously, and perhaps with other things assumed without question to be true. This is what happens, for instance, when a witness gives inconsistent evidence in court; in the course of giving evidence he says something that contradicts the body of testimony he has given previously.

In seeking to define consistency for sets of formulas of $\mathbf{H_s}$ it's easier to attend first to the opposite notion of inconsistency. The set of formulas $\{P, {\sim}P\}$ is a paradigm example of an inconsistent set; a particularly overt contradiction is exhibited between its members. However, we would also want to say that a set such as $\{P \supset Q, P, {\sim}Q\}$ is inconsistent even though none of its members is the negation of any other of its members. The inconsistency of this set is evident from the fact that we can deduce a contradiction from it in $\mathbf{H_s}$: both $P \supset Q, P, {\sim}Q \vdash Q$ and $P \supset Q, P, {\sim}Q \vdash {\sim}Q$.

This suggests that we should define consistency in such a way that a set Γ of formulas of $\mathbf{H_s}$ is inconsistent if and only if a contradiction is deducible from it: i.e., if and only if for some formula A of $\mathbf{H_s}$, $\Gamma \vdash A$ and $\Gamma \vdash {\sim}A$. In view of the fact that a contradiction is deducible from a set of formulas if and only if *every* formula of $\mathbf{H_s}$ is deducible from that set, there is a particularly simple way to make the definition. We can say that a set Γ is consistent if some formula isn't deducible from it; the consistent sets are those from which not everything is deducible.

D11. A set Γ of formulas of $\mathbf{H_s}$ *is consistent (in* $\mathbf{H_s}$*) if there is some formula A of* $\mathbf{H_s}$ *such that not $\Gamma \vdash A$. A set of formulas is said to be inconsistent if it is not consistent.*

We can now prove as a metatheorem our claim that a set of formulas is inconsistent if and only if some formula and its negation are deducible from it.

M32. A set Γ of formulas of $\mathbf{H_s}$ *is inconsistent if and only if for some formula A of* $\mathbf{H_s}$*, $\Gamma \vdash A$ and $\Gamma \vdash {\sim}A$.*

PROOF. If Γ is inconsistent then $\Gamma \vdash B$ for all B so that in particular $\Gamma \vdash P_1$ and $\Gamma \vdash {\sim}P_1$. Suppose, on the other hand, that for some A, $\Gamma \vdash A$ and $\Gamma \vdash {\sim}A$. By M13, $A, {\sim}A \vdash B$ for all formulas B of $\mathbf{H_s}$ and so by M6, $\Gamma \vdash B$ for all formulas B of $\mathbf{H_s}$; i.e., Γ is inconsistent. This proves the metatheorem.

Proving the following metatheorems about consistency is left as an exercise; the proof of M33 relies on M3.

M33. A set Γ *of formulas of* $\mathbf{H}_s$ *is consistent if and only if every finite subset of* Γ *is consistent.*
M34. $\Gamma \vdash A$ *if and only if* $\Gamma \cup \{\sim A\}$ *is inconsistent.*
M35. If Γ *is consistent, then for all formulas A of* $\mathbf{H}_s$*, either* $\Gamma \cup \{A\}$ *is consistent or* $\Gamma \cup \{\sim A\}$ *is consistent.*

If we like, we can say that a formula A of $\mathbf{H}_s$ is inconsistent if the set $\{A\}$ is inconsistent. It then turns out (E15(b)) that a formula A is inconsistent if and only if $\vdash \sim A$.

16. So far, we have dealt with two systems that purport to be formulations of the logic of implication and negation: $\mathbf{H}_s$ and $\mathbf{S}_{\supset\sim}$. Now clearly, these are very different systems, and yet it would be a strange thing if they turned out to sanction different arguments as logically correct.

In this, the final section of this chapter, we turn our attention to showing that in a sense of 'equivalence' to be made precise later, these systems are equivalent to one another. This will be our first really *intersystematic* discussion, since the metatheorems below will apply to both systems at once. Our previous metatheorems had to do only with $\mathbf{H}_s$.

Before proceeding any further, it will be a good precaution to return a moment to use and mention. This time, the problem is that $\mathbf{S}_{\supset\sim}$ was formulated before we had become self-conscious about our use of quotation marks and metavariables. Rather than being presented abstractly as the symbols of $\mathbf{H}_s$ were, via noncommital names of them, the symbols of $\mathbf{S}_{\supset\sim}$ were displayed and spoken about by means of quotation marks.

The most convenient way to eliminate the awkwardness that would result from having to speak of $\mathbf{S}_{\supset\sim}$ in one way and $\mathbf{H}_s$ in another is simply to reformulate $\mathbf{S}_{\supset\sim}$ so that its symbols and formulas are precisely those of $\mathbf{H}_s$. From here on, then, we will suppose that the definition of the formulas of $\mathbf{S}_{\supset\sim}$ is that given on p. 55, above. The deductive structure of $\mathbf{S}_{\supset\sim}$—the subordinate derivation framework of reiteration, and the four rules of implication and negation introduction and elimination—remain unchanged.

We can now say that the *formulas* of $\mathbf{S}_{\supset\sim}$ and $\mathbf{H}_s$ are the same: any formula of $\mathbf{S}_{\supset\sim}$ is a formula of $\mathbf{H}_s$, and conversely any formula of $\mathbf{H}_s$ is a formula of $\mathbf{S}_{\supset\sim}$. But of course, this isn't enough to show the two systems equivalent in any interesting sense. What we want to know is that any deduction that can be carried out in $\mathbf{H}_s$ corresponds to a derivation that can be carried out in $\mathbf{S}_{\supset\sim}$, and conversely that every derivation in $\mathbf{S}_{\supset\sim}$ corresponds to a deduction in $\mathbf{H}_s$. Let's write '$A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$' to mean that there is a derivation in $\mathbf{S}_{\supset\sim}$ of B from hypotheses $A_1, \ldots, A_n$. Thus if, for example, $A, B \vdash_{S_{\supset\sim}} C$, then there is some way to fill in the middle of

```
| A
|_B
| ⋮
| C
```

so that the result is a derivation in $\mathbf{S}_{\supset\sim}$. Our problem then reduces to two parts: to show that if $A_1, \ldots, A_n \vdash B$, then $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$, and to show that if $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$, then $A_1, \ldots, A_n \vdash B$.

The first of these parts is the easier, so we'll tackle it first. In order to show that if $A_1, \ldots, A_n \vdash B$ then $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$, we will first show something apparently not so strong: that if there is a proof in $\mathbf{H}_s$ of a formula B, then there is a categorical derivation of B in $\mathbf{S}_{\supset\sim}$.

M36. If $\vdash B$ *then* $\vdash_{S_{\supset\sim}} B$.

PROOF. We use the same technique that was employed in the proof of M31; thus, all we need to show is that if A is an axiom of $\mathbf{H}_s$ then A is derivable categorically in $\mathbf{S}_{\supset\sim}$, and that if A and $A \supset B$ are derivable categorically in $\mathbf{S}_{\supset\sim}$ then so is B.

But it's an easy matter to show that if A is an axiom of $\mathbf{H}_s$ then $\vdash_{S_{\supset\sim}} A$. For instance, if A is an instance of AS1, then A has the form $B \supset . C \supset B$; and the following derivation scheme shows it is derivable categorically in $\mathbf{S}_{\supset\sim}$.

```
| |_B
| | |_C
| | | B
| | C ⊃ B
| B ⊃. C ⊃ B
```

And if there is a categorical derivation of A in $\mathbf{S}_{\supset\sim}$, and another of $A \supset B$ in $\mathbf{S}_{\supset\sim}$, we merely have to put these two derivations together in tandem and follow them by B, in order to get a categorical derivation of B.

We now know that any theorem of $\mathbf{H}_s$ is derivable categorically in $\mathbf{S}_{\supset\sim}$. Let's go on to get a general result about deductions from this.

M37. If $A_1, \ldots, A_n \vdash B$ *then* $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$.

PROOF. Suppose $A_1, A_2, \ldots, A_n \vdash B$; then by M10, $\vdash A_1 \supset . A_2 \supset . \cdots \supset .$

$A_n \supset B$. Applying M36, we know that $\vdash_{S_{\supset\sim}} A_1 \supset . A_2 \supset . \cdots \supset . A_n \supset B$; this means that there is a categorical derivation of $A_1 \supset . A_2 \supset . \cdots \supset . A_n \supset B$ in $S_{\supset\sim}$. Coining a piece of notation, let's use '$\mathfrak{A}$' to stand for this derivation. Then the following will stand for a derivation of B from $A_1, A_2, \ldots, A_n$ in $S_{\supset\sim}$.

```
| A₁
| A₂
| ⋮
|_Aₙ
| 𝔄
| A₂ ⊃. ··· ⊃. Aₙ ⊃ B
| ⋮
| Aₙ ⊃ B
| B
```

Thus, there is a derivation of B from $A_1, A_2, \ldots, A_n$ in $S_{\supset\sim}$; i.e., $A_1, A_2, \ldots, A_n \vdash_{S_{\supset\sim}} B$, as desired.

To establish the converse of M37, we have to resort to a more roundabout reasoning. We will use an inductive argument in which a derivation is gradually transformed, by taking out one subordinated derivation after another, into a deduction in $\mathbf{H}_s$. Since this construction forces us to deal with intermediate steps in the process as well as the terminal ones, we must consider hybrid arrays in which the axioms of $\mathbf{H}_s$ may be used, as well as the rules of $S_{\supset\sim}$.

We will say, then, that an *augmented derivation* (in $S_{\supset\sim}$) is an array like a derivation in $S_{\supset\sim}$, except that any axiom of $\mathbf{H}_s$ may be introduced anywhere in it. The following, for example, schematizes an augmented derivation.

```
| |_A
| | |_~A
| | | ~B ⊃ ~A ⊃. A ⊃ B
| | | ~A ⊃. ~B ⊃ ~A
| | | ~B ⊃ ~A
| | | A ⊃ B
| | | A
| | | B
| | ~A ⊃ B
| A ⊃. ~A ⊃ B
```

(*xiii*)

Notice that in this example the axioms of $\mathbf{H}_s$ have been used in such a way that only two hypotheses need to be made in xiii; without using any axioms, three would be needed. The idea of our argument below is that this procedure can be carried out systematically, in such a way as to eliminate *all* hypotheses.

M38. If there is an augmented derivation (in $S_{\supset\sim}$*) of B from* $A_1, \ldots, A_n$, *then* $A_1, \ldots, A_n \vdash B$.

PROOF. Suppose that there is an augmented derivation of B from $A_1, \ldots, A_n$. In the event that this augmented derivation contains any subordinated derivations (i.e., any derivations in which hypotheses are made), there will be at least one *innermost* such derivation—one to which no derivations are subordinated. This situation can be pictured as follows.

```
| A₁
| ⋮
|_Aₙ
|      ·.
|         | ⋮
|         | |_C₁
|         | | C₂
|         | | ⋮
|         | | Cₘ
|         | ⋮
|      ·'
| B
```

(*xiv*)

Here, $C_1, C_2, \ldots, C_m$ are the successive steps of a particular innermost derivation.

We will now outline a procedure for transforming the augmented derivation presented in xiv into an augmented derivation in which the innermost subordinated derivation

```
|_C₁
| C₂
| ⋮
| Cₘ
```

(*xv*)

has been eliminated.

The first step is to erase the line demarcating the derivation to be eliminated, and to prefix C_1 to each of the entries of this derivation; this transforms xiv into the following array.

$$
\begin{array}{|l}
A_1 \\
\vdots \\
\underline{A_n} \\
\quad \ddots \\
\qquad \begin{array}{|l}
\vdots \\
C_1 \supset C_1 \\
C_1 \supset C_2 \\
\vdots \\
C_1 \supset C_m \\
\vdots
\end{array}
\end{array}
$$

(*xvi*)

Then, formulas are inserted as they are in our proof of the deduction theorem. More precisely, the step $C_1 \supset C_1$ is treated as in case 2.2 of our proof of M8; if C_i was justified by reit in xiv, steps are inserted as in case 2.1 of that proof; and if C_i was justified by *modus ponens* in xiv, it is treated as in case 3. Since xv has no derivation subordinate to it, none of its steps can be justified by an introduction rule and there is one remaining possibility: C_i may have been justified by negation elimination. In that case, a deduction in $\mathbf{H}_s$ of C_i from $\sim\sim C_i$ is to be inserted above C_i. (The existence of such a deduction is guaranteed by M14.)

The result of applying the procedure to the array presented in xv will be an augmented derivation, but one of a special sort: each of its steps will be an axiom of $\mathbf{H}_s$, or be justified by *modus ponens* or reiteration. Thus, the entire array obtained from the augmented derivation displayed in xiv will itself be an augmented derivation.

Clearly, this procedure can be applied repeatedly as long as there are subordinated derivations remaining to be eliminated. If we do this, eventually we will get an augmented derivation of B from $A_1, \ldots, A_n$ that looks like this.

$$
\begin{array}{|l}
A_1 \\
\vdots \\
\underline{A_n} \\
D_1 \\
\vdots \\
D_k \\
B
\end{array}
$$

(*xvii*)

The augmented derivation given by xvii contains no subordinated derivations, and hence no introduction rules are used in it. (This is true of $S_{\supset\sim}$ because the introduction rules for both $\supset$ and $\sim$ require subordinated derivations.) There may be uses of negation elimination, however, in xvii; these can be replaced by deductions of D_i from $\sim\sim D_i$, which, being deductions, use only axioms of $\mathbf{H}_s$ and *modus ponens*.

After this has been done, we obtain an array in which $A_1, \ldots, A_n$ are the only hypotheses made. Every other step is an axiom of $\mathbf{H}_s$ or a consequence by *modus ponens* of previous steps. In other words, the result is a *deduction* in $\mathbf{H}_s$ of B from $A_1, \ldots, A_n$.

Thus, $A_1, \ldots, A_n \vdash B$, as desired. The proof of M38 is now finished.

From M38 it is easy to get the converse of M37. Suppose that $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$; i.e., suppose that there is a derivation in $S_{\supset\sim}$ of B from $A_1, \ldots, A_n$. This derivation is itself an augmented derivation, though one in which no axioms of $\mathbf{H}_s$ are used. Hence, by M38, we know that $A_1, \ldots, A_n \vdash B$. Thus, we have the following metatheorem.

M39. If $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$, then $A_1, \ldots, A_n \vdash B$.

Finally, putting together M37 and M39, we have a metatheorem establishing the equivalence of $\mathbf{H}_s$ and $S_{\supset\sim}$.

M40. $A_1, \ldots, A_n \vdash B$ if and only if $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$.

Reflecting on M40, you may wonder what is the point of having two systems at all. In $\mathbf{H}_s$, we obtain as theorems precisely those formulas that are derivable in $S_{\supset\sim}$. So, especially since $S_{\supset\sim}$ is simpler and more natural to use, why should we bother with $\mathbf{H}_s$?

But there are various kinds of simplicity, and which is best on a given occasion can depend on what we want to do with a system. $S_{\supset\sim}$ is indeed close to the way we actually reason and is by far the better of the two systems for finding derivations or for evaluating arguments in natural language. On the other hand, $\mathbf{H}_s$ is simpler in that it is formulated more economically. It has only one rule of inference and three axiom-schemes, and its deductions involve no nesting, being linear arrays of formulas. Thus $\mathbf{H}_s$ can be set up more succinctly than $S_{\supset\sim}$. By the same token, it often is easier to prove metatheorems about $\mathbf{H}_s$ than about $S_{\supset\sim}$, and it is chiefly for this reason that when we began to concentrate on the metatheoretic aspects of logic we also directed our attention to $\mathbf{H}_s$.

Besides Hilbert-style or axiomatic systems like $\mathbf{H_s}$ and natural deduction systems like $S_{\supset \sim}$, there are many other sorts of systems that have been devised by logicians, many of them with special advantages of their own. And since there are many different things one might want of a system of logic, this is probably a good thing; it provides a flexibility that we would not otherwise have.

Exercises

1. What does it mean to say that A is the same as B, where 'A' and 'B' are meta-variables taking formulas of $\mathbf{H_s}$ as values? (The best answer would be an inductive definition.)

2. Insert quotes into the following text (*Through the Looking-Glass*, Chapter VIII).

   You are sad, the Knight said in an anxious tone: let me sing you a song to comfort you. Is it very long? Alice asked, for she had heard a great deal of poetry that day. It's long, said the Knight, but it's *very very* beautiful. Everybody that hears me sing it—either it brings the *tears* into their eyes or else—Or else what? said Alice, for the Knight had made a sudden pause. Or else it doesn't you know. The name of the song is called Haddock's Eyes. Oh, that's the name of the song, is it? Alice said, trying to feel interested. No, you don't understand, the Knight said, looking a little vexed. That's what the name of the song is *called*. The name really is The Aged Aged Man. Then I ought to have said That's what the song is called? Alice corrected herself. No, you oughtn't: that's quite another thing! The *song* is called Ways and Means: but that's only what it's called, you know! Well, what *is* the song, then? Said Alice, who by this time was completely bewildered. I was coming to that, the Knight said. The song really *is* A-sitting on a Gate: and the tune's my own invention.

3. Insert quotes in the following paragraph:
   We use $\{P, P \supset Q\} \vdash Q$ to say that there is a deduction of Q from hypotheses P and $P \supset Q$. This deduction will be an array $A_1, \ldots, A_n$ of formulas; in particular, $P, P \supset Q, Q$ is such an array. P can only take as values the sentence parameters of $\mathbf{H_s}$; however, the result of putting A and B, which represent arbitrary formulas, for P and Q in $P, P \supset Q, Q$ will represent a deduction. Hence, we know that $\{A, A \supset B\} \vdash B$ for all formulas A and B of $\mathbf{H_s}$.

4. Is $P_1 \vee P_2$ a formula of $\mathbf{H_s}$? What about '$P_1 \vee P_2$'?

5. Restore parentheses to the following abbreviations.

   (a) $P \supset P \supset Q$
   (b) $(P \supset . Q \supset P) \supset . P \supset . \sim P \supset Q$

   (c) $P \supset P \supset . P \supset P$
   (d) $A \supset . B \supset A$
   (e) $\sim(A \supset B) \supset A \supset \sim A \supset . A \supset B$
   (f) $\sim P \supset Q$

6. Rewrite the following, eliminating all uses of '$\vee$', '$\wedge$', or '$\equiv$'.

   (a) $(A \vee B) \supset . \sim A \supset B$
   (b) $A \supset (A \vee B)$
   (c) $(A \vee B) \supset (B \vee A)$
   (d) $(A \wedge B) \supset A$
   (e) $A \supset . B \supset (A \wedge B)$
   (f) $(\sim A \vee \sim B) \supset \sim(A \wedge B)$
   (g) $A \equiv \sim A$
   (h) $(A \equiv B) \supset (\sim A \equiv \sim B)$

7. Extend the language of $\mathbf{H_s}$ to include disjunction. Give an inductive definition of the formulas of the extended language, and find axiom-schemes which make it possible to prove theorems appropriate for disjunction. What is the difference between this and *defining* disjunction, as we did in Section 11?

8. Produce derivations in $\mathbf{H_s}$ establishing the following statements.

   (a) $\vdash P_1 \supset . P_1 \supset P_1 \supset . P_1 \supset P_1$
   (b) $\vdash \sim P_1 \supset . P_1 \supset P_2$
   (c) $\vdash P_1 \supset P_2 \supset . P_3 \supset P_1 \supset . P_3 \supset P_2$
   (d) $\vdash P_1 \supset P_2 \supset . P_2 \supset P_3 \supset . P_1 \supset P_3$
   (e) $P_1 \supset . P_1 \supset P_2, P_1 \vdash P_2$
   (f) $P_1 \supset . P_1 \supset P_2 \vdash P_1 \supset P_2$
   (g) $P_1 \supset \sim(P_2 \supset P_2) \vdash \sim P_1$
   (h) $P_1 \vdash \sim\sim P_1$
   (i) $\sim\sim\sim P_1 \vdash \sim P_1$
   (j) $P_1 \supset . P_2 \supset \sim P_3, P_1, P_3 \vdash \sim P_2$

9. Demonstrate the following metatheorems (in each case, you may use any results established before the metatheorem in question).

(a) M4	(b) M5
(c) M6	(d) M7
(e) M10	(f) M24
(g) M25	(h) M26
(i) M27	(j) M28
(k) M30	(l) M33
(m) M34	(n) M35

10. Could a formula be an instance of both AS1 and AS2? If so, find an example; if not, show that such a formula is impossible.

11. Show that every subformula of a formula of $\mathbf{H}_s$ is a formula of $\mathbf{H}_s$. (*Hint:* use an inductive argument.)

12. Using only M4 and M8, demonstrate the following metatheorems.

(a) $\vdash A \supset . B \supset A$ (b) $\vdash A \supset A$
(c) $\vdash A \supset . C \supset . B \supset C$ (d) $\vdash A \supset . B \supset B$

13. Using any metatheorem in the text (except M38, M39, and M40), establish the following.

(a) $A \supset B \vdash \sim B \supset \sim A$ (b) $A \vdash \sim\sim A$
(c) $\sim A \supset A \vdash A$ (d) $A \supset B, \sim A \supset B \vdash B$
(e) $A \vee B \vdash B \vee A$ (f) $\vdash \sim A \vee A$
(g) $A \supset B, \sim A \supset \sim B \vdash A \equiv B$ (h) $\vdash \sim(A \wedge \sim A)$
(i) $A \wedge A \vdash A$ (j) $\sim(A \wedge B), A \vdash A \wedge \sim B$
(k) $\sim(A \equiv B) \vdash A \equiv \sim B$ (l) $A \equiv B, \sim A \vdash \sim B$

14. In this exercise we will clarify what is meant by an *inference* and a *rule* of the system $\mathbf{H}_s$. By an inference of $\mathbf{H}_s$, we mean an ordered pair $\langle \Gamma, A \rangle$ where Γ is a set of formulas and A a formula of $\mathbf{H}_s$. Γ is the set of *premisses* and A is the *conclusion* of this inference. Thus, in the inference $\langle\{P \supset Q, P\}, Q\rangle$, Q is inferred from the premisses $P \supset Q$ and P. This inference is an instance of *modus ponens*.

A rule is a general way of passing from premisses to conclusion. We will therefore understand by a rule of $\mathbf{H}_s$ a set of inferences of $\mathbf{H}_s$. Those inferences which are members of a rule $\mathscr{R}$ are the *instances* of $\mathscr{R}$. The rule *modus ponens*, for instance, is the set $\{\langle\{A \supset B, A\}, B\rangle \mid A$ and B are formulas of $\mathbf{H}_s\}$ of inferences. This conception of a rule is so general that we can look at axiom-schemes as rules of a special sort: rules whose instances have empty premisses and so are inferences of the form $\langle \varnothing, A \rangle$.

To say that $\mathscr{R}$ is a rule of $\mathbf{H}_s$ is not to say that $\mathscr{R}$ is correct or valid in any way. For instance, the rule $\{\langle\{A\}, B\rangle \mid A$ and B are formulas of $\mathbf{H}_s\}$, "from anything to infer anything", is not a rule that is reasonable in any sense; nevertheless, it is a rule of $\mathbf{H}_s$. We can, however, talk about *primitive*, *admissible*, and *derived* rules of $\mathbf{H}_s$. The rule *modus ponens* is the only primitive rule of $\mathbf{H}_s$, unless the three axiom-schemes are also counted as primitive rules. A rule $\mathscr{R}$ is an *admissible rule* of $\mathbf{H}_s$ if $\vdash A$ whenever $\langle \Gamma, A \rangle \in \mathscr{R}$ and for all $B \in \Gamma$, $\vdash B$. Thus, $\mathscr{R}$ is admissible to $\mathbf{H}_s$ if its addition to $\mathbf{H}_s$ as a primitive rule would not increase the theorems of $\mathbf{H}_s$. A rule $\mathscr{R}$ is a *derived rule of* $\mathbf{H}_s$ if $\Gamma \vdash A$ whenever $\langle \Gamma, A \rangle \in \mathscr{R}$.

The following are exercises involving these notions.

(a) Show that if a rule is derivable in $\mathbf{H}_s$ then it is admissible in $\mathbf{H}_s$.
(b) Show that the rule $\{\langle\{A\}, B\rangle \mid A$ is a sentence parameter$\}$ is admissible in $\mathbf{H}_s$.

(c) The empty set $\varnothing$ is a rule of $\mathbf{H}_s$. Show that it is a derived rule of $\mathbf{H}_s$.
(d) Show that the rule $\{\langle\{C, A \equiv B\}, C'\rangle \mid C'$ is a result of replacing A by B in $C\}$ is a derived rule of $\mathbf{H}_s$. (Use M29 and M26.)

15. Prove the following.

(a) Γ is consistent if and only if it is not the case that $\Gamma \vdash P_1 \wedge \sim P_1$.
(b) $\{A\}$ is inconsistent if and only if $\vdash \sim A$.

16. Using M8 and M11, prove M6.

17. Say that $A \simeq B$ if $\vdash A \equiv B$. Show that $\simeq$ is an equivalence relation: i.e., that $A \simeq A$, if $A \simeq B$ then $B \simeq A$, and if $A \simeq B$ and $B \simeq C$ then $A \simeq C$.

18. Say that A and B are *synonymous* in $\mathbf{H}_s$ if replacement of A by B never affects provability; i.e., if whenever C' is a result of replacing A by B in C, then $\vdash C$ if and only if $\vdash C'$. Show that A and B are synonymous in $\mathbf{H}_s$ if and only if $\vdash A \equiv B$.

19. Let $\mathbf{H}_s^1$ be like $\mathbf{H}_s$, except that AS2 is replaced by the following schemes.

$$A \supset B \supset . B \supset C \supset . A \supset C$$
$$(A \supset . A \supset B) \supset . A \supset B$$

Show that $\mathbf{H}_s^1$ is equivalent to $\mathbf{H}_s$.

20. Let $\mathbf{H}_s^2$ be like $\mathbf{H}_s$ except that AS3 is replaced by the following three schemes.

$$(A \supset \sim A) \supset \sim A$$
$$A \supset . \sim A \supset B$$
$$A \supset B \supset A \supset A$$

Show that $\mathbf{H}_s^2$ is equivalent to $\mathbf{H}_s$.

21. Extend $\mathbf{H}_s$ by adding a further symbol $\wedge$ and stipulating that if A and B are formulas, then so is $(A \wedge B)$. Define a system $\mathbf{H}_s^3$ for this larger set of formulas by devising suitable axiom-schemes. Then (a) show that $\mathbf{H}_s^3$ is a *conservative extension* of $\mathbf{H}_s$; i.e., show that if A is a formula of $\mathbf{H}_s$, then $\vdash A$ if $\vdash_{\mathbf{H}_s^3} A$. (b) Using M40, go on to show that $\mathbf{H}_s^3$ is equivalent to $\mathbf{S}_{\supset \sim \wedge}$.

Problems

1. Where X and Y are strings of symbols, we say that XY is the result of *concatenating* X and Y. In order to make our treatment of $\mathbf{H}_s$ fully abstract and rigorous, we would have to make explicit the assumptions we have been making about concatenation. State some of these assumptions, and use them to show that if $A \supset B$ is $C \supset D$, then A is C and B is D, and that $A \supset \sim B$ is different from $A \supset . C \supset D$.

2. Devise an axiomatic system $\mathbf{H}_{\supset}$ (with *modus ponens* its only primitive rule of inference) which is equivalent to $\mathbf{S}_{\supset}$. Carry through the proof of equivalence.
3. Isolate the properties of $\mathbf{H}_s$ that are used in proving M29.
4. Let $\mathrm{S}^{B_1 \ldots B_n}_{Q_1 \ldots Q_n} A \mid$ be the result of ***simultaneously substituting*** $B_1, \ldots, B_n$ for the respective parameters $Q_1, \ldots, Q_n$ in A. Define (in a manner analogous to D10) this notion of substitution. Then show that if $\vdash A$, then $\vdash \mathrm{S}^{B_1 \ldots B_n}_{Q_1 \ldots Q_n} A \mid$. Try to find an argument that uses M31. Does this argument work for $\mathbf{S}_0$?
5. Using M4, M8, M11, M19, and M20, devise a proof of M39 simpler than the one given in the text. (*Hint:* consider the proof of M20, and note its relation to a derivation in $\mathbf{S}_{\supset\sim}$. Then state this relation generally.)
6. If the axiom-scheme

$$A \supset A$$

were to be added to $\mathbf{H}_s$, it would be *redundant*, since any instance of it is already provable in $\mathbf{H}_s$. Is any one of the three axiom schemes of $\mathbf{H}_s$ redundant?

VI Sentence Logic: Semantics

1. Up to this point we have made no direct use in our metatheory of the idea that language should be *about* something. In view of the fact that natural languages and mathematical notations are all used to say things (and this is surely their most important function), we should correct this neglect.

We have, of course, been careful to show how portions of English can be translated into our formal systems, and in Chapters III and IV we used this connection with natural language to justify our choice of primitive rules for $\mathbf{S}_s$. This justification, however, wasn't something that we *proved*, as metatheorems were proved in Chapter V. Instead, it took place on an entirely different level. The relationship of formal to natural language is rather like the relationship of geometry to surveying practices; one establishes this relation by acquiring some theoretical knowledge of geometry and some practical knowledge of surveying, and then connecting the two. Thus, one doesn't prove in geometry that the sum of the angles of this page is 360°; this is an *applied* piece of information that springs from imposing Euclidean

2. Devise an axiomatic system $\mathbf{H}_{\supset}$ (with *modus ponens* its only primitive rule of inference) which is equivalent to $\mathbf{S}_{\supset}$. Carry through the proof of equivalence.
3. Isolate the properties of $\mathbf{H}_s$ that are used in proving M29.
4. Let $S^{B_1 \ldots B_n}_{Q_1 \ldots Q_n} A \mid$ be the result of *simultaneously substituting* $B_1, \ldots, B_n$ for the respective parameters $Q_1, \ldots, Q_n$ in A. Define (in a manner analogous to D10) this notion of substitution. Then show that if $\vdash A$, then $\vdash S^{B_1 \ldots B_n}_{Q_1 \ldots Q_n} A \mid$. Try to find an argument that uses M31. Does this argument work for $\mathbf{S}_0$?
5. Using M4, M8, M11, M19, and M20, devise a proof of M39 simpler than the one given in the text. (*Hint:* consider the proof of M20, and note its relation to a derivation in $\mathbf{S}_{\supset\sim}$. Then state this relation generally.)
6. If the axiom-scheme

$$A \supset A$$

were to be added to $\mathbf{H}_s$, it would be *redundant*, since any instance of it is already provable in $\mathbf{H}_s$. Is any one of the three axiom schemes of $\mathbf{H}_s$ redundant?

VI Sentence Logic: Semantics

1. Up to this point we have made no direct use in our metatheory of the idea that language should be *about* something. In view of the fact that natural languages and mathematical notations are all used to say things (and this is surely their most important function), we should correct this neglect.

We have, of course, been careful to show how portions of English can be translated into our formal systems, and in Chapters III and IV we used this connection with natural language to justify our choice of primitive rules for $\mathbf{S}_s$. This justification, however, wasn't something that we *proved*, as metatheorems were proved in Chapter V. Instead, it took place on an entirely different level. The relationship of formal to natural language is rather like the relationship of geometry to surveying practices; one establishes this relation by acquiring some theoretical knowledge of geometry and some practical knowledge of surveying, and then connecting the two. Thus, one doesn't prove in geometry that the sum of the angles of this page is 360°; this is an *applied* piece of information that springs from imposing Euclidean theory on a body of practices involving measurement, pattern recognition, and the like.

Now, the point of all this is the following. If in our discussion of how $\mathbf{H}_s$ is to be interpreted we are to maintain the standards of rigor set in the previous chapter, we shouldn't simply return to what we did in Chapter III. What we must do is to draw the "aboutness" relation up to the level of theory, so that we will have a definition of what $\mathbf{H}_s$ is about that is fully as rigorous as our definition of $\mathbf{H}_s$ itself. In this way we will obtain a *semantic* theory of our formal languages, one that treats both the languages and their "aboutness".

In general, semantic and *syntactic* theories of language are distinguished by the fact that the latter do not include an account of how things may be said with language, whereas the former do. The theory of Chapter V was syntactic, and the notions of *formula, proof, theorem, deduction, deducibility,* and *consistency* are syntactic concepts. Semantic methods have been refined in recent years to a high state of development, and include some of the most powerful and useful techniques to be found in modern logic. Like their syntactic counterparts, these semantic theories employ mathematical methods and concepts; frequently, however, these are more advanced (and sometimes, more problematic) than those used for syntactic purposes.

2. Let's launch our semantic discussion informally, with an account of truth-tables. The idea behind this technique is simply that sentence parameters stand for sentences that may be *true* or *false* in some given situation. Since, for instance, we can think of sentence parameters standing for true sentences as true, this leads us to think of the sentence parameters themselves as being either true or false. Having gone this far, we then ask what will be the truth-values of complex formulas which are made up of these parameters.

Some cases of this general question are not at all hard to settle. For instance, if P is true then $\sim P$ will be false, and if P is false, then $\sim P$ will be true. This seems to accord well with our intuitive conception of truth and falsity. Implications, though, are a bit more difficult. Suppose that we have a formula $P \supset Q$; how will its truth-value depend on the truth-values of P and Q? Well, surely $P \supset Q$ will be *false* in case P is true and Q false; there seems to be no question about this.

But the remaining three cases (P true and Q true, P false and Q true, and P false and Q false) are still unsettled. In facing up to this question we must decide what to do with examples such as the following.

> If Tucson is a city, then there is no largest even number.
> If there is a largest even number, then Tucson is a city.
> If there is a largest even number, then Tucson is a vegetable.

Such examples do not come up often in everyday specimens of reasoning, and most people encountering them for the first time would be puzzled as to their truth-values. Some might want to say that they are false or nonsensical.

But on the other hand, one *can* find examples of implications of each of these three kinds which almost anyone would want to call true. Here are some.

> If there is no largest even number, then there is no largest number.
> If 5 is the largest even number, then 7 is not an even number.
> If 5 is the largest even number, then 6 is not an even number.

In view of these, it certainly seems as if we must *sometimes* make $P \supset Q$ true in each of these three cases. But, if we must do this *sometimes*, systematic considerations suggest that we ought to do it every time, in each of these cases.

There are several reasons that can be given in support of this claim. One of the most important of these is suggested by the inductive techniques that we developed in Chapter V. There we dealt with notions like the result of substituting A for P in B, which for complex formulas B depends on the result of substituting A for P in the component formulas of B. This leads us to be guided in our semantic theory by the *principle of truth-functionality*: the truth-value of a complex formula is completely determined by the truth-values of the simpler formulas that constitute it. This principle is the fundamental assumption on which the method of truth-tables rests. We could not compute the truth-values of formulas as we do below without relying on the principle of truth-functionality.

If we were to deny the principle of truth-functionality for formulas of the sort $P \supset Q$, we would have to say, for instance, that $P \supset Q$ is sometimes true and sometimes false when P is false and Q is false. This would force us to say that the truth-value of $P \supset Q$ depends on something besides the truth-values of P and Q. Whatever we decide this additional factor is, we would have to bring it into our semantic theory. And however we do this, we are bound to end up with a theory more complicated than one that depends on the principle of truth-functionality. This argument doesn't show that it would be foolish and misguided to deny the principle of truth-functionality, but it does indicate that it's a good idea to begin the study of logical semantics by accepting this principle.

Finally, we may note that the assumption that $P \supset Q$ is always true in case P is false or Q is true does no harm, at least in the following sense. A true implication can never allow us to infer a false conclusion by *modus ponens* from true premisses. The reason for this is that whenever $P \supset Q$ and P are both true, then Q is true, since we've already agreed to make $P \supset Q$ false in case P is true and Q false.

We can summarize the results of this discussion in tabular form as follows.

P	$\sim P$
T	F
F	T

(*i*)

P	Q	$P \supset Q$
T	T	T
T	F	F
F	T	T
F	F	T

(*ii*)

Table i gives the rule (or function) that determines the truth-value of $\sim P$ in terms of the truth-value of P. And table ii in the same way gives the function for determining the truth-value of $P \supset Q$ in terms of the truth-values of P and of Q.

3. Since P and Q are sentence parameters, what holds for them in i and ii will hold generally for *any* formulas of $\mathbf{H_s}$; for instance, we know that if $Q \supset R$ is true and $\sim P$ true, then $Q \supset R \supset \sim P$ is true. Thus, we can express i and ii more generally in the following schematic form.

A	$\sim A$
T	F
F	T

(*iii*)

A	B	$A \supset B$
T	T	T
T	F	F
F	T	T
F	F	T

(*iv*)

The rules of truth-valuation given by iii and iv are general enough to determine a unique truth-value for any formula of $\mathbf{H_s}$, given the truth-values of its constituent sentence parameters. For example, suppose P, Q, and R are assigned the truth-values T, F, and F, respectively, and consider the formula $\sim(P \supset Q) \supset \sim\sim Q \supset. P \supset R$. Since P takes T and Q takes F, $P \supset Q$ takes F (here, we use rule iv, letting A be P and B be Q). Since $P \supset Q$ takes F, $\sim(P \supset Q)$ takes T. Since Q takes F, $\sim Q$ takes T, and since $\sim Q$ takes T, $\sim\sim Q$ takes F. Proceeding in this way, we see that the truth-value of the whole formula is T. This procedure of successively determining the truth-values of larger and larger parts of the formula can be set out as follows.

$\sim (P \supset Q) \supset \sim \sim Q \supset. P \supset R$
T TF F F F T F T TF F
(*v*)

Here, the truth-values given to P, Q, and R are written under them, and the truth-values given to complex subformulas are written under their principal connectives. (The principal connective of a formula $\sim C$ is the occurrence of $\sim$ to the left of C; the principal connective of a formula $C \supset D$ is the occurrence of $\supset$ between C and D.)

As another example, consider $\sim P \supset Q \supset Q$, where P is given the value T and Q the value F.

$\sim P \supset Q \supset Q$
F T T F F F
(*vi*)

The value given to this formula will be F, as shown in vi.

4. Examples such as v and vi illustrate how one can determine the truth-value taken by formulas in a given situation; e.g., in vi, the truth-value of $\sim P \supset Q \supset Q$ is computed for the case in which P is true and Q false. In other circumstances, $\sim P \supset Q \supset Q$ might take different truth-values. For instance, if P is true and Q true, it will be true. It is a common practice to display all these various outcomes in tables resembling i and ii; the following is an example.

P	Q	$\sim P$	$\supset Q$	$\supset Q$
T	T	F	T	T
T	F	F	T	F
F	T	T	T	T
F	F	T	F	T

(*vii*)

In this *truth-table* for $\sim P \supset Q \supset Q$, all the various possible assignments of truth-values to P and Q appear to the left; the resultant values appear to the right. The table shows how the formula behaves truth-functionally; it is false if and only if P is true and Q is false. (In other words, it behaves just like $P \supset Q$.)

If we consider a formula in which three sentence parameters occur, we'll have to reckon with eight possible assignments of truth-values. The following is an example of a truth-table for such a formula, showing only the resultant value for the whole formula.

P	Q	R	$\sim(P \supset \sim R \supset . \sim(R \supset Q))$
T	T	T	F
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	T
F	T	F	T
F	F	T	F
F	F	F	T

(*viii*)

Where we have exactly one sentence parameter in a formula, the number of rows in its truth-table will be two; where there are two parameters, it will be four; where there are three, eight. In general, n sentence parameters will require 2^n rows. If we wished to make a truth-table for a formula having ten parameters, we would need 1024 rows!

5. When a natural language is used to say true or false things, it is *situations* that make its sentences true or false. For instance, the sentence 'I was in Rome yesterday' is not true or false until some situation is specified which determines who is uttering the sentence and when. It is true if he was in Rome the previous day, and false otherwise.

In the semantics of sentence logic we are only interested in what truth-values situations give to sentence parameters. For this reason, we will think of a situation as something that determines the truth-values of the sentence parameters of a formula, and does nothing else. The truth-value of the formula in a situation can then be calculated using rules iii and iv.

Now, consider an example such as the following.

P	Q	$P \supset \sim Q \supset . Q \supset \sim P$
T	T	T
T	F	T
F	T	T
F	F	T

(*ix*)

The table shows that $P \supset \sim Q \supset . Q \supset \sim P$ is true in all situations, so that

this formula cannot be false under any circumstances. Let's call such formulas *valid.*

The only way a formula $A \supset B$ can be invalid is for A to be true when B is false. For this reason, where there are many implication signs in a formula, it's often easier to show it valid by showing that its antecedent is false whenever its consequent is false. Thus, in example ix, if $Q \supset \sim P$ is false, P must be true and Q true. But in that case, $P \supset \sim Q$ is false. Hence, $P \supset \sim Q$ cannot be true when $Q \supset \sim P$ is false, so $P \supset \sim Q \supset . Q \supset \sim P$ is valid. The validity of $P \supset \sim Q \supset . Q \supset \sim P$ is thus shown by the following table.

$$P \supset \sim Q \supset . Q \supset \sim P$$
$$\text{T F F T T T F F T}$$

(*x*)

6. Now let's redo the above material and put it into more rigorous form. Our idea is to take the notions of being true in some situation and of being true in all situations and to define them in a way that will be appropriate for proving metatheorems about them.

First we should deal with the process of assigning truth-values to parameters. Here, we must ask ourselves how many parameters need to be given values in a situation; all of them, or just some? If we say that only some parameters need to be assigned truth-values, we will have to admit that some formulas are made neither true nor false by an assignment of truth-values; these will be the formulas containing parameters without truth-values. On the other hand, in examples such as ix, above, we only bothered to assign truth-values to the parameters occurring in the formulas under consideration. It would be awkward to have to assign values to all the other parameters of $\mathbf{H}_s$ (infinitely many of them) just to determine the truth-value taken by $P \supset \sim Q \supset . Q \supset \sim P$ under an assignment.

In order to get around this difficulty we will reformulate the system $\mathbf{H}_s$, stepping once more to a higher level of generality. Instead of considering one specific *vocabulary* (i.e., the class of formulas built out of one set of sentence parameters), we will allow the deductive framework of $\mathbf{H}_s$ to be imposed on an arbitrary vocabulary. To carry out this generalization, we introduce the notion of a *morphology* for $\mathbf{H}_s$. A morphology is simply a nonempty set **M** of sentence parameters, and may be finite or infinite. Given such a morphology, the formation rules of $\mathbf{H}_s$ will determine a unique set of formulas; these will be the formulas made up of parameters drawn from **M**. We can then speak of *valuations* as assigning truth-values to each parameter in a morphology **M**, and it will turn out that every valuation of a morphology **M** makes every formula of **M** either true or false. Let's record definitions of these notions.

D1. A morphology **M** *for* $\mathbf{H_s}$ *is a nonempty set of objects called sentence parameters. The notions of formula of* **M**, *theorem of* **M**, *and deducibility in* **M** *are defined as in Chapter V, the only change being that just sentence parameters in* **M** *are considered.*

As before, we use 'P', 'Q', and so on to range over sentence parameters and 'A', 'B', and so forth to range over formulas, but this time over sentence parameters and formulas of whatever morphology is under consideration.

D2. A valuation of a morphology **M** *for* $\mathbf{H_s}$ *is a function* V *which assigns each sentence parameter in* **M** *one (and only one) of the values* T *and* F. *Where* $P \in M$ *and* V *is a valuation of* **M**, V(P) *is the value assigned to* P *by* V. *Thus* V(P) = T *or* V(P) = F.

A valuation of **M** will determine a truth-value for each sentence parameter of **M**. In accordance with the law of noncontradiction (nothing is both true and false), we have specified in D2 that every parameter is assigned *at* most one of the values T and F. And observing the law of bivalence (everything is true or is false), we have ensured that every parameter of **M** is assigned *at least* one of these two values.

In accordance with rules iii and iv, every valuation of **M** assigns a truth-value T or F to every formula of **M**, however complex. When we reformulate this precisely it turns out to be an inductive definition, since the truth-value given to a formula depends on the truth-values given its subformulas.

D3. Let V *be a valuation of a morphology* **M** *for* $\mathbf{H_s}$. *The truth-value* V(A) *of a complex formula* A *of* **M** *under* V *is defined according to the following rules.*

1. *If* A *is* $B \supset C$, *then* V(A) = T *if* V(B) = F *or* V(C) = T, *and* V(A) = F *otherwise.*
2. *If* A *is* $\sim B$, *then* V(A) = T *if* V(B) = F, *and* V(A) = F *otherwise.*

A value T or F qualifies as the truth-value V(A) of A under the valuation V of **M** only if A is a sentence parameter of **M** and this value is the one assigned to A by V, or else A is a complex formula of **M** and this value can be shown by repeated applications of 1 and 2 to be the value assigned to A.

Recall that where A is a sentence parameter of **M**, V(A) is determined by decree; or, if you like, V(A) is specified as part of the definition of V. If A is complex, however (e.g., if A is $\sim P$ or $P \supset . P \supset \sim Q$, where P and Q are parameters of **M**), V(A) is computed from V(P) and V(Q) by means of clauses 1 and 2.

For example, suppose that V(P) = T and V(Q) = F. Then by clause 2, V($\sim P$) = F. It takes a few more steps to calculate V($P \supset . P \supset \sim Q$). By clause 2, V($\sim Q$) = T, and hence by clause 1, V($P \supset \sim Q$) = T; therefore, by clause 1, V($P \supset . P \supset \sim Q$) = T.

7. Now, if **M** is a morphology and V a valuation of **M**, it seems clear that every formula of **M** will be either true or false with respect to V and that no formula of **M** will be both true and false. Just to fix this in our minds, however, we will give a demonstration of it. As you might expect, the proof is inductive.

M1. Where M *is a morphology for* $\mathbf{H_s}$ *and* V *a valuation of* **M**, *every formula* A *of* **M** *is given a unique value* T *or* F *by* V.

PROOF. Induce on the complexity of A (i.e., on the number of occurrences of connectives in A). If A is a sentence parameter of **M**, V gives a unique value T or F to A by D2. This furnishes the basis clause of our argument. Suppose as hypothesis of induction that all formulas B of **M** less complex than A (i.e., containing fewer occurrences of $\supset$ and $\sim$ than A) are assigned one and only one truth-value by V. Then, in case A is an implication $C \supset D$, both C and D take a unique truth-value by our hypothesis of induction; therefore so does A, by clause 1 of D3. And in case A is a negation $\sim C$, C takes a unique truth-value by the hypothesis of induction, and therefore so does A by clause 2 of D3. This completes the proof of M1.

8. D3 characterizes the fundamental semantic notion of satisfaction by a valuation V. A formula A of a morphology **M** is said to be *satisfied* by a valuation V of **M** if V(A) = T. (Here, 'satisfaction' is used in the sense of 'making good', as when we say that a thing satisfies certain criteria. To be satisfied by a valuation is to be made true by that valuation.) Now that this relation of satisfaction has been specified, we can go on to account for other semantic concepts of importance in logic. In particular, the notions of *satisfiability* (or capability of being made true) and *validity* (or incapability of being made false) are especially worth defining and investigating.

D4. A formula A *of* **M** *is satisfiable if there exists a valuation* V *of* **M** *such that* V(A) = T.

D5. A formula A of M *is valid if for all valuations* V *of* M, V(*A*) = T.

Let's consider an example or two to illustrate these definitions. Suppose that M = {*P*, *Q*}, set $V_1(P) = T$, and $V_1(Q) = F$. Let $V_2(P) = F$ and $V_2(Q) = T$. Then the formula *P* is satisfiable, since some valuation of M (e.g., V_1) satisfies *P*. The negation ~*P* of *P* is also satisfiable, since some valuation of M (e.g., V_2) satisfies it. Likewise, *P* is not valid, since there is a valuation of M (e.g., V_1) which fails to satisfy it. Notice that there is an implicit appeal to M1 in this reasoning: V_1 doesn't satisfy ~*P* because $V_1(\sim P) = F$, so that by M1, $V_1(\sim P) \neq T$. These examples show that there is nothing wrong with both a formula and its negation being satisfiable, though, as M1 shows, they cannot both be satisfied by the very same valuation.

To take a slightly more complicated example, $\sim(P \supset Q)$ is satisfiable, because V_1 satisfies it. On the other hand, it's easy to see that $P \supset P$ is valid, since any valuation of M satisfies this formula. By the same token, $\sim(P \supset P)$ is not satisfiable as there is no way of making this formula true.

Probably you've noticed the relationship of truth-tables to all this. A valuation corresponds to a row of a truth-table. It turns out, then, that a formula is satisfiable if it takes a T in at least one row of its truth-table, and valid if it takes a T in all of these rows. For a simple example illustrating this, again let M = {*P*, *Q*}. Then there are four valuations of M in all, corresponding to the four rows of a truth-table for two parameters. Thus, table vii below shows that the formula $\sim(\sim P \supset Q)$ is satisfiable, since it takes the value T in the last row, and table viii shows $P \supset \sim\sim P$ to be valid.

P	*Q*	$\sim(\sim P \supset Q)$
T	T	F
T	F	F
F	T	F
F	F	T

(*vii*)

P	*Q*	$P \supset \sim\sim P$
T	T	T
T	F	T
F	T	T
F	F	T

(*viii*)

Notice that table viii is redundant, since *Q* does not appear in the formula $P \supset \sim\sim P$. A row for *Q* was included in this table because it is supposed to correspond to the valuations of the morphology M, and there are four such valuations.

9. People sometimes get confused about satisfaction and satisfiability, and it may be a good idea to distinguish between them even more carefully than we have. Satisfaction is a *relation* between a formula and a valuation; it makes no sense to speak of a formula being satisfied without regard to a valuation. On the other hand, satisfiability is a quality that may or may not be possessed by formulas; it is not a relation, and it would be meaningless to speak of a formula being "satisfiable by a valuation". In this regard *satisfaction* and *satisfiability* are like *proof* and *provability*. In at least one sense of 'proof', proof is a relation between arrays of formulas and formulas; an array of a certain sort is said to be a *proof of* its last formula. A formula is then said to be *provable* if there exists a proof of it. In general, when you have a relation that holds between things of one sort and things of another, you can always define a property pertaining to things of the first sort in terms of this relation. To do this, you consider those things of the first sort which bear the given relation to *some* thing of the second sort. This is what is done in the case of satisfiability and provability.

Satisfaction, as we have said, is a relation between formulas and valuations. Satisfiability is defined in terms of satisfaction, so that a formula is satisfiable if there is some valuation that satisfies it. Validity, on the other hand, is defined in terms of satisfaction in a slightly different way, by considering those formulas of a morphology M which are satisfied by *all* valuations of M.

10. Up to now we've been indulging in a certain amount of sloppiness, and it would be best to clear this up before going ahead. When we defined satisfiability and validity, we did this with respect to a morphology M. For instance, we said that a formula of M is valid if it is made true by all valuations of M. Nevertheless, we have not spoken of "validity with respect to M", but merely of validity. Of course, it would be very odd and unsettling if indeed it could happen that a formula were valid when regarded as a formula of one morphology and invalid when regarded as a formula of another. In order to justify our way of speaking, we must show that this sort of thing can't happen. This is just what our next metatheorem is meant to establish.

M2. Let M *and* M′ *be morphologies for* H_s, *and let* M′ ⊆ M (*i.e., let* M′ *be a submorphology of* M). *Where* V *is a valuation of* M, *let* V′ *be the restriction of* V *to* M′; *that is, let* V′(*P*) = V(*P*) *for all P* ∈ M′, *and* V′ *be undefined for P* ∉ M′. *Then for all formulas A of* M′, V(*A*) = V′(*A*).

PROOF. Let M, M′, V, and V′ be as described above. Our argument proceeds by induction on the complexity of formulas *A* of M′. If *A* is a sentence parameter, then V(*A*) = V′(*A*) by the assumptions of our theorem. Suppose as hypothesis of induction that for all formulas *B* of M′ less complex than *A*, V(*B*) = V′(*B*). Then if *A* is an implication $C \supset D$, V(*A*) and V′(*A*) are determined according to D3 from V(*C*) and V(*D*), V′(*C*) and V′(*D*). Since the hypothesis of induction guarantees that V(C) = V′(C) and V(D) = V′(D), we have V(A) = V′(A). Similarly, if A is a negation, V(A) = V′(A).

Using M2, it's easy to show, as we do in M3, that the satisfiability of a formula A does not depend on which morphology A is considered to be a formula of.

M3. Let **M** *and* **M′** *be morphologies for* $\mathbf{H}_s$ *and A be a formula of both* **M** *and* **M′**. *Then there is a valuation of* **M** *that satisfies A if and only if there is a valuation of* **M′** *that satisfies A.*

PROOF. Consider **M** ∩ **M′**; this is a submorphology of both **M** and **M′**, and A is a formula of it. It follows from M2 that there is a valuation V of **M** which satisfies A if and only if there is a valuation of **M** ∩ **M′** (namely, the restriction of V to **M** ∩ **M′**) which satisfies A. Similarly, there is a valuation of **M′** which satisfies A if and only if there is a valuation of **M** ∩ **M′** which satisfies A. Therefore, some valuation of **M** satisfies A if and only if some valuation of **M′** satisfies A.

In the same way, it's easy to show that under the same conditions every valuation of **M** satisfies A if and only if every valuation of **M′** satisfies A. The proof of this is left as an exercise.

M4. Let **M** *and* **M′** *be morphologies for* $\mathbf{H}_s$, *and A be a formula of both* **M** *and* **M′**. *Then every valuation of* **M** *satisfies A if and only if every valuation of* **M′** *satisfies A.*

Since satisfaction by some valuation amounts to satisfiability, and satisfaction by every valuation to validity, M3 and M4 show that we can consider these notions independently of this or that morphology; for instance, we can say simply that $P \supset Q$ is satisfiable, not satisfiable with respect to **M**. In fact, what the proof of M3 shows is that a formula A is satisfiable when considered as a formula of **M** if and only if it is satisfiable with respect to the morphology containing just the parameters in A. And M4 shows the same thing about validity. Thus, our characterization in Section 5, above, of satisfiability and validity in terms of truth-tables is also justified. Looking at the truth-table of A to see if A takes a T in some row amounts to asking whether A is satisfiable with respect to the morphology containing just the sentence parameters in A.

Notice that, although we haven't bothered to do so, the notions of *provability*, *derivability*, and *consistency* should also be shown to be independent of the morphology chosen. One of these problems is stated more clearly and assigned as a task to the reader in E8 below.

In view of the fact that checking the truth-table of a formula suffices to show whether it's satisfiable or not and whether it's valid or not, you may wonder why we have gone to the trouble of talking about morphologies and valuations, and choosing definitions of *satisfiability* and *validity* that must be shown to be independent of morphology. The answer is that we are interested not so much in checking whether particular formulas are satisfiable or valid as in proving general things about satisfiability and validity. For this purpose the definitions we have chosen turn out to be better than definitions using just truth-tables. And to develop the semantics of predicate logic, truth-tables no longer suffice and we will be forced to use definitions employing the notion of a valuation. So it's better to get used to them now.

11. Before turning to other matters, we should set down a few basic metatheorems concerning satisfiability and validity. The first two of these require no explanation and may have occurred to you already. Their proof is left as an exercise.

M5. A is valid if and only if $\sim A$ *is not satisfiable.*

M6. A is satisfiable if and only if $\sim A$ *is not valid.*

Another feature of validity is that it is closed under substitution. (That is, any result of substitution in a valid formula is also a valid formula.) For instance, as soon as we know that $P \supset Q \supset. P \supset \sim Q \supset \sim P$ is valid, we know that any formula $A \supset B \supset. A \supset \sim B \supset. \sim A$ is valid. The reason for this, of course, is that where A is a formula of **M**, any valuation V of **M** must give A and B truth-values V(A) and V(B). But the validity of $P \supset Q \supset. P \supset \sim Q \supset \sim P$ shows that $A \supset B \supset. A \supset \sim B \supset \sim A$ will take the value T no matter what these values are.

Closure of validity under substitution follows from M7 below. Its proof is left to you; to be strictly rigorous, you should refer back to V.D10 and furnish an inductive argument.

M7. Let A and B be formulas of **M**, *and let C be the result of substituting A for P in B, where P is a sentence parameter of* **M**. *Then if B is valid, so is C.*

12. Other notions of great importance in logical semantics are *simultaneous satisfaction*, *simultaneous satisfiability*, and (semantic) *implication.*

All of these apply to *sets* of formulas, rather than to formulas taken one at a time.

Interest in sets of formulas arises naturally. We may be interested, say, in the set of formulas that are deducible from certain formulas posited as hypotheses. This is a syntactically defined set of formulas, since the notion of deducibility is syntactic, but it's interesting to ask semantic questions about it; for instance, are all its members true in a given situation? In terms of our semantic theory, this amounts to asking of a valuation V whether it satisfies every formula in the set, or, as we will put it from now on, whether V *simultaneously satisfies* the set.

D6. Let Γ *be a set of formulas of a given morphology* **M**, *and* V *be a valuation of* **M**. *The valuation* V *simultaneously satisfies* Γ *if* V *satisfies every formula in* Γ.

For example, let $\Gamma = \{P \supset Q, \sim Q, P \supset P\}$, and let $\mathbf{M} = \{P, Q\}$. Let $V_1(P) = T$, $V_1(Q) = F$, and $V_2(P) = F$, $V_2(Q) = F$. Then V_2 simultaneously satisfies Γ, since clearly V_2 satisfies the three formulas $P \supset Q$, $\sim Q$, and $P \supset P$. But V_1 does not simultaneously satisfy Γ, since there is a formula in Γ (namely, $P \supset Q$) which V_1 doesn't satisfy.

Thus, we can think of V as simultaneously satisfying Γ if V satisfies "the conjunction of formulas in Γ". But a little reflection shows that this formulation is sloppy. In the first place, even if Γ is finite, there is in general no unique way of grouping the members of Γ into a conjunction. Even though all these conjunctions can be proved to be equivalent in $\mathbf{H}_s$, they are different formulas. And if Γ is infinite, things are even worse; there is no way of getting all the members of Γ into a conjunction, since all formulas (of *any* morphology) are of finite length. Nevertheless, our original idea has some value, in spite of its sloppiness, since in making it precise we arrive at some conjectures that can be proved as metatheorems.

First, we need some better notation for talking about conjunctions. Since the way we group parentheses in a conjunction really doesn't matter much, let's group at the left, so that the expression '$A_1 \wedge A_2 \wedge A_3 \wedge A_4$' abbreviates '$(((A_1 \wedge A_2) \wedge A_3) \wedge A_4)$'. So far as finite sets go, then, we have the following metatheorem.

M8. Let $\{A_1, \ldots, A_n\}$ *be a finite set of formulas of some morphology* **M**, *and let* V *be a valuation of* **M**. *Then* V *simultaneously satisfies* $\{A_1, \ldots, A_n\}$ *if and only if* V *satisfies* $A_1 \wedge \cdots \wedge A_n$.

Using the fact (which follows easily from D3) that any valuation V satisfies $A \wedge B$ if and only if V satisfies A and V satisfies B, it's simple to show inductively that V satisfies $A_1 \wedge \cdots \wedge A_n$ if and only if V satisfies A_i for all i, $1 \leq i \leq n$. From this and D6, M8 follows right away.

As far as infinite sets go, though we can't speak of conjunctions of all their members, we can speak of arbitrarily large finite conjunctions of formulas drawn from them. In this way we get the following generalization of M8.

M9. Let Γ *be a set of formulas of a morphology* **M**, *and let* V *be a valuation of* **M**. *Then* V *simultaneously satisfies* Γ *if and only if for all finite subsets* $\{A_1, \ldots, A_n\}$ *of* Γ, V *satisfies* $A_1 \wedge \cdots \wedge A_n$.

There are a number of ways to go about proving M9. Perhaps the simplest of these is to show (exercise 9 below), that V simultaneously satisfies Γ if and only if V simultaneously satisfies every finite subset of Γ. Then M8 can be used to get M9. But it's also easy to prove M9 directly.

Another metatheorem about simultaneous satisfaction is the following; its proof is left as an exercise.

M10. Let Γ *be a set of formulas of a morphology* **M** *and* V *be a valuation of* **M**. *Then if* V *simultaneously satisfies* Γ, V *simultaneously satisfies every subset of* Γ.

13. Simultaneous satisfiability stands to simultaneous satisfaction as satisfiability stands to satisfaction. When we wonder whether a set Γ of formulas is simultaneously satisfiable, we want to know whether there is some way of making all the formulas of Γ true.

D7. Let Γ *be a set of formulas of* **M**. Γ *is simultaneously satisfiable if there exists some valuation of* **M** *which simultaneously satisfies* Γ.

As in the case of satisfiability, we must face the task of showing that simultaneous satisfiability is independent of morphology. Here, what we first need is a result like M2.

M11. Let **M** *and* **M′** *be morphologies of* $\mathbf{H}_s$ *and let* **M′** *be a submorphology of* **M**. *Let* V *be a valuation of* **M** *and* V′ *the restriction of* V *to* **M′** (*so that* V′ *is like* V, *but is defined only for members of* **M′**). *Then for all sets* Γ *of formulas of* **M′**, V *simultaneously satisfies* Γ *if and only if* V′ *simultaneously satisfies* Γ.

No induction is needed to prove M11; M2 and D6 do the trick. Having obtained M11, we can use it in the same way M2 was used to obtain M3 to get the result we want. This time we will present a proof, though again it is not difficult.

M12. Let Γ be a set of formulas of both **M** *and* **M**′. *Then there is a valuation of* **M** *that simultaneously satisfies* Γ *if and only if there is a valuation of* **M**′ *that simultaneously satisfies* Γ.

PROOF. As in the proof of M3, consider the morphology **M** ∩ **M**′; Γ is a set of formulas of **M** ∩ **M**′. Now, by M11 there is a valuation of **M** which simultaneously satisfies Γ if and only if there is a valuation of **M** ∩ **M**′ that simultaneously satisfies Γ. Likewise, there is a valuation of **M**′ which simultaneously satisfies Γ if and only if there is a valuation of **M** ∩ **M**′ which simultaneously satisfies Γ. So at once M12 follows.

Other results that come to mind about simultaneous satisfiability are the following, which we state without proof.

M13. If a set Γ *of formulas is simultaneously satisfiable, then every subset of* Γ *is simultaneously satisfiable.*

M14. $\{A\}$ *is simultaneously satisfiable if and only if A is satisfiable.*

M15. If Γ *is simultaneously satisfiable and A is valid, then* $\Gamma \cup \{A\}$ *is simultaneously satisfiable.*

14. How is simultaneous satisfiability of Γ related to satisfiability of the members of Γ? Well, if Γ is simultaneously satisfiable, all its members are satisfiable. But the converse of this is false. For instance, consider P and $\sim P$. These are both satisfiable; we can make each of them true. But we can't make them both true *at once*; the set $\{P, \sim P\}$ is not simultaneously satisfiable. So simultaneous satisfiability has to be determined by looking at the set as a whole, not at its individual members separately.

There is a bit more that can be said about this question, though. In view of M8, we know that simultaneous satisfiability of a finite set amounts to satisfiability of a conjunction of its members. In other words, $\{A_1, \ldots, A_n\}$ is simultaneously satisfiable if and only if $A_1 \wedge \cdots \wedge A_n$ is satisfiable. However, this doesn't tell us anything about infinite sets; to do this, it must be generalized. The most promising way of seeking such a generalization is to try

to find an analogue of M9. In this way we arrive at the following conjecture: A set Γ of formulas is simultaneously satisfiable if and only if all of its finite subsets are simultaneously satisfiable.

Knowing as much as we do, we can say a good deal about this conjecture. On the basis of previous results (in particular, M8), we know it holds when Γ is finite. So any trouble that may arise in trying to prove it will come when Γ is infinite. Also, by M13 we can see that if Γ is simultaneously satisfiable all of its finite subsets are simultaneously satisfiable. This leaves us with the problem of settling (with special reference to the case in which Γ is infinite) whether Γ is simultaneously satisfiable if all its finite subsets are. This doesn't look so easy; consider, for instance, a case in which $\Gamma = \{A_1, A_2, A_3, \ldots\}$. Now, we can assume that for every n, $\{A_1, \ldots, A_n\}$ is simultaneously satisfiable. This only means that there is for each n a valuation V_n which simultaneously satisfies $\{A_1, \ldots, A_n\}$. But this certainly doesn't guarantee in any obvious way that there is a *single* valuation that simultaneously satisfies $\{A_1, \ldots, A_n\}$ for *every n*. And this last is what we need to prove.

For the time being, then, we will have to leave this question unsolved (it is presented, with a few hints, as a problem at the end of this chapter). But the property we are interested in here, often called *compactness*, is an extremely important one, and we will furnish in the next chapter a (rather devious) proof that a set is simultaneously satisfiable if all its finite subsets are.

15. The last semantic notion to be defined in terms of satisfaction is implication. In semantic theories implication is treated as a relation between sets of formulas, on the one hand, and formulas on the other. What we need to do is to give semantic conditions under which a formula A can be regarded as a consequence of a set Γ of formulas. We accomplish this by stipulating that Γ implies A in case A is true in every situation in which every formula in Γ is true.

D8. Let Γ *be a set of formulas of* **M**, *and A be a formula of* **M**. Γ (*semantically*) *implies A, i.e.,* $\Gamma \Vdash A$, *if every valuation of* **M** *which simultaneously satisfies* Γ *also satisfies A.*

For instance, let Γ be the set $\{P, Q \supset \sim P\}$, and let **M** consist of P and Q. Then the only valuation of **M** which simultaneously satisfies Γ will make P true and Q false, and hence we know that $\Gamma \Vdash \sim Q$. Or consider the set $\Delta = \{P \supset \sim P, \sim(P \supset \sim P)\}$. In view of M1 there is *no* valuation of **M** which simultaneously satisfies Δ. Thus, vacuously, every valuation of **M** which simultaneously satisfies Δ also satisfies Q. Therefore $\Delta \Vdash Q$. To take a case where we have an infinite set, it's easy to see that for all n, $\{P_1, P_2, P_3, \ldots\} \Vdash P_1 \wedge \cdots \wedge P_n$.

16. Notice the resemblance of '$\Vdash$' to the symbol '$\vdash$' we use in talking about deducibility. This notation was chosen as a reminder that, although there is a great difference in the way the two are defined, there are many similarities between deducibility and implication. Many of the metatheorems listed below are suggested by this analogy.

M16. Let Γ *be a set of formulas of both* **M** *and* **M**$'$, *and let* A *be a formula of both* **M** *and* **M**$'$. *Then* A *is satisfied by every valuation of* **M** *which simultaneously satisfies* Γ *if and only if* A *is satisfied by every valuation of* **M**$'$ *which simultaneously satisfies* Γ.

This metatheorem can be proved directly in much the same way as M12 or it can be obtained more easily by using the following metatheorem, together with M12.

M17. $\Gamma \Vdash A$ *if and only if* $\Gamma \cup \{\sim A\}$ *is not simultaneously satisfiable.*

PROOF. Suppose that $\Gamma \Vdash A$. Then, with respect to some given **M** such that A and every member of Γ are formulas of **M**, $V(A) = T$ for every valuation V of **M** which simultaneously satisfies Γ. Then, by M1, for every such valuation V, $V(A) \neq F$, and so, by D3, $V(\sim A) \neq T$. Therefore, $\Gamma \cup \{\sim A\}$ is not simultaneously satisfiable, since every valuation of **M** which simultaneously satisfies Γ must fail to satisfy $\sim A$. This argument also reverses: if $\Gamma \cup \{\sim A\}$ is not simultaneously satisfiable, then every valuation of **M** which simultaneously satisfies Γ must fail to satisfy $\sim A$, and hence must satisfy A. And so we've finished the proof of M17.

The next metatheorems are all straightforward consequences of our definitions, so their proofs are not written out below. You may wish to try some of them before reading on.

M18. $\varnothing \Vdash A$ *if and only if* A *is valid.*

We will use the notation '$\Vdash A$' to signify that A is valid, exploiting in this way the fact that validity is a special case of implication.

M19. If $A \in \Gamma$, *then* $\Gamma \Vdash A$.

M20. If $\Gamma \Vdash A$, *then* $\Gamma \cup \Delta \Vdash A$.

M21. If $\Gamma \Vdash A$ *and* $\Delta \cup \{A\} \Vdash B$, *then* $\Gamma \cup \Delta \Vdash B$.

M22. If $\Gamma \Vdash A \supset B$, *then* $\Gamma \cup \{A\} \Vdash B$.

M23. If $\Gamma \cup \{A\} \Vdash B$, *then* $\Gamma \Vdash A \supset B$.

An interesting notion related to implication is that of *implicative closure*. The implicative closure with respect to **M** of a set Γ of formulas of **M** is the set of all formulas A of **M** which are implied by Γ. If we think of Γ as determining the class of situations in which it is true, the implicative closure of Γ will consist of those formulas that are true in all of these situations. Clearly, Γ will always be a subset of its implicative closure. But instead of developing this and other properties of implicative closure as metatheorems, we will treat this topic in the exercises for this chapter.

17. Before going on to other matters, let's return briefly to the treatment of truth-tables which we gave in Sections 3 and 4 and discuss how this may be applied to complex formulas such as $P \vee Q$. If we wish to determine the truth-table of a formula such as $\sim Q \supset (P \vee Q)$, we can do it by calculating the truth-table of $\sim Q \supset (P \supset Q \supset Q)$, which of course is the same formula.

But this makes things more complicated than need be. Once we know the truth-table of $P \supset Q \supset Q$, we can treat the formula $P \vee Q$ as a unit and calculate its truth-table directly as a function of P and Q. In other words, we can use *derived* truth-tables, which are composed out of the *primitive* truth-tables assigned to negation and implication.

The truth-table corresponding to $P \vee Q$ is the following one.

P	Q	$P \supset Q \supset Q$
T	T	T
T	F	T
F	T	T
F	F	F

(*ix*)

This turns out just as we would expect in view of our discussion of disjunction in IV.5. There we said that we would construe disjunction in an inclusive sense, so that $P \vee Q$ would be true if P is true or if Q is true or if both are true, and false if both are false. And this is just what is stated in ix.

Here, then, we have another justification of our definition of disjunction: a *semantic* justification, rather than the syntactic one of V.12. Similar justifications can be given for the definitions of conjunction and equivalence. All that is needed is to work out the truth-tables for $P \wedge Q$ and $P \equiv Q$, and to relate these truth-tables to the intuitive discussion of conjunction and equivalence in Chapter IV.

Of course, all of this can also be expressed in terms of the more abstract notion of a valuation. We notice that the truth-value $V(A \vee B)$ of a formula $A \vee B$ of **M** (with respect to a valuation V of **M**) is a function of the values $V(A)$ and $V(B)$ which V gives to A and B. If we like, we can express this as a metatheorem.

M24. Let A and B be formulas of **M**, *and* V *a valuation of* **M**. *Then* $V(A \vee B) =$ T *if and only if* $V(A) =$ T *or* $V(B) =$ T *(or both).*

Similar metatheorems can be obtained for conjunction and equivalence.

18. Every formula of $\mathbf{H_s}$ corresponds to a truth-table which shows what truth-values the formula takes when its sentence parameters are assigned various truth-values. We will say that a formula *expresses* its truth-table; in case a formula has n sentence parameters it will express a truth-table for n parameters. In Section 4 we remarked that such a truth-table will have 2^n rows. In the preceding section we argued that because $P \supset Q \supset Q$ expresses the truth-table of disjunction, it is justifiable to regard '$P \vee Q$' as an abbreviation of '$P \supset Q \supset Q$'. It is this that allows us to regard formulas of the sort '$A \supset B \supset B$' as disjunctions.

This suggests that we can use truth-tables to add another dimension to the account of definition that was given in V.11. In seeking to define disjunction in $\mathbf{H_s}$, we start out with the notion of a particular truth-table for two parameters P and Q: the table ix for disjunction. We then try to find a formula of $\mathbf{H_s}$ which expresses this truth-table. In a semantic sense, what we define in D5, D6, and D7 of Chapter V are the truth-tables for disjunction, conjunction, and equivalence. In each case we do this by finding a formula of $\mathbf{H_s}$ which expresses the truth-table we have in mind.

All of this raises a general question: can a definition be found in $\mathbf{H_s}$ of *any* truth-table we can think of? More precisely, let's choose a particular infinite morphology $\{P_1, P_2, \ldots\}$. (This ensures that we have enough parameters to enable us to express truth-tables of arbitrary size.) Given any truth-

table for n parameters, is there a formula of this morphology which expresses that truth-table?

Although we won't bother to give a detailed proof of this, it is indeed true that $\mathbf{H_s}$ has this property. This is established by presenting a method of finding, given any truth-table, a formula that expresses that table. An example or two will indicate how the method works in general. Consider the following truth-table for two parameters.

P	Q	
T	T	F
T	F	T
F	T	F
F	F	T

(*x*)

The left-hand rows of the table correspond to situations (the situation in which P is true and Q is true, the situation in which P is true and Q false, etc.). On the right-hand side of the table are given the resultant truth-values for these situations. A formula expressing this truth-table can be true in only two situations: the one in which P is true and Q is false, or the one in which P is false and Q is false. To express this truth-table then, we choose a formula of $\mathbf{H_s}$ which says just this: $(P \wedge \sim Q) \vee (\sim P \wedge \sim Q)$. Since this formula is true if and only if P is true and Q is false, or P is false and Q is false, it expresses the truth-table x.

This method works for any truth-table having at least one T in its right-hand column. For example, take a truth-table for three parameters.

P	Q	R	
T	T	T	T
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	T
F	F	F	F

(*xi*)

In this case the method produces the formula

$$(((P \wedge Q \wedge R) \vee (P \wedge \sim Q \wedge R)) \vee (\sim P \wedge Q \wedge R)) \vee (\sim P \wedge \sim Q \wedge R).$$

In cases where there are no T's in the right-hand column, we will have tables such as the following one.

P	Q	
T	T	F
T	F	F
F	T	F
F	F	F

(*xii*)

These correspond to formulas that are not satisfiable, and we need only choose any such formula. For instance, $(P \wedge \sim P) \wedge (Q \wedge \sim Q)$ expresses the truth-table xii.

The general result whose proof is indicated by these examples is called the *expressive completeness* of the language of $\mathbf{H}_s$: a language, or way of constructing formulas by means of connectives, is *expressively complete* with respect to truth-tables if any truth-table is expressed by some formula of the language. To give a full-scale proof of this result we would have to furnish an explicit statement of the method exemplified above and then show, by an inductive argument, that this method always produces a formula that expresses the given truth-table. Expressive completeness, by the way, is just one of many technical senses of the word 'completeness'. Logicians tend to overwork this word, and we will run across a more important sense of it in the next chapter, where we prove the semantic completeness of $\mathbf{H}_s$.

There are, of course, languages other than that of $\mathbf{H}_s$ which are expressively complete. For instance, if we were to take negation and disjunction as primitive connectives, this language would be expressively complete. This can be shown by pointing out that implication is definable in terms of disjunction and negation, and then appealing to the expressive completeness of $\mathbf{H}_s$.

On the other hand, there are many languages which are *expressively incomplete* with regard to truth-tables. An example is the pure implicative language, which is based on just one binary connective, $\supset$. With respect to a morphology **M**, the formulas of this language will be the sentence parameters of **M** together with all expressions generated by the following rule: if A and B are formulas, then so is $(A \supset B)$. To show this language expressively incomplete, one has to find some truth-table that cannot be expressed by any formula of the language. Generally, the best way to do this is to search for some property not possessed by all truth-tables which can be shown by an inductive argument to be possessed by every truth-table expressible in the language.

In the case of the pure implicative language, a good property to use is that of yielding the value T when every parameter takes the value T. Certainly the truth-table expressed by a sentence parameter P has this property, and it looks as if every formula gotten by implication from formulas expressing such a truth-table will also do so.

To prove this rigorously we use an induction on the complexity of formulas of the pure implicative language, showing that if **M** is a morphology and V the valuation of **M** which assigns every sentence parameter in **M** the value T, then for all formulas A of **M** in this language, $V(A) = T$. In the basis step of the induction A will be a sentence parameter P of **M**, and by assumption $V(A) = T$. This completes the basis step. For the inductive step, let A be $B \supset C$ and assume as inductive hypothesis that for all formulas D less complex than A, $V(D) = T$. Then $V(B) = T$ and $V(C) = T$, and it follows by D3 that $V(A) = T$.

This shows that no truth-table that yields the value F when every parameter takes the value T can be expressed in the pure implicative language. For example, the table for negation, i, is such a truth-table; negation is therefore not definable in terms of implication.

There are many ways in which these ideas could be further developed. In particular, though it's interesting to find alternative languages for sentence logic which are expressively complete, it is often more rewarding to think about expressively incomplete languages. It frequently requires ingenuity to prove such systems incomplete, and sometimes in trying to find the right property of truth-tables to use in an inductive argument, one hits on some revealing results. Also, when a language is expressively incomplete, it is interesting to try to find an axiomatic system which generates the valid formulas of the language. But we will go on to other matters now and leave these topics to the exercises and problems of this and the next chapter.

19. In the above sections we have been speaking of *languages* as well as morphologies, and though this terminology is more general than the one we will use throughout the rest of the book, perhaps we should pause to explain it. The following account is very general and abstract; what we have been doing with $\mathbf{H}_s$ is only one of its many special cases.

By a (truth-functionally interpreted) *sentential language* we will understand a set of connectives that permit the construction of complex formulas out of sentence parameters. The pure implicative language and the language of $\mathbf{H}_s$ are both languages in this sense. Each of the connectives of a sentential language must have some *degree*, which tells how many formulas it connects. Disjunction, for example, has degree 2, or is a *2-ary* connective; negation is

1-ary. We will therefore stipulate that for each n, a sentential language possesses a set of n-ary connectives. For many numbers n, this set may be empty.

Besides these syntactic features, truth-functionally interpreted languages have a semantic side. Each of their connectives must be assigned a truth-table of appropriate size; i.e., an n-ary connective should be given a truth-table for n parameters.

We therefore say that a truth-functionally interpreted sentential language is to consist of the following components.

1. For each $n \geq 0$, a set C^n of n-ary *connectives*.
2. An assignment of a truth-value to each 0-ary connective of the language and of a truth-table having n parameters to each n-ary ($n \geq 1$) connective of the language.

In the case of $\mathbf{H}_s$, the set of 1-ary connectives is $\{\sim\}$, the set of 2-ary connectives is $\{\supset\}$, and all the other sets are empty; there are, for instance, no 3-ary or 0-ary connectives of $\mathbf{H}_s$. The truth-tables assigned to $\sim$ and $\supset$ are, of course, the tables iii and iv of Section 3, above.

The notion of a 0-ary connective may need some explaining. Since n-ary connectives correspond to functions which, given n truth-values, produce a truth-value, 0-ary connectives should correspond to functions which, given *no* truth-values, produce a truth-value. Thus, 0-ary connectives should correspond to a fixed truth-value T or F.

With respect to each morphology **M**, we want each sentential language to determine a unique set of formulas. One way of accomplishing this is to say that every parameter of the morphology and every 0-ary connective of the language is a formula, and that if $A_1, \ldots, A_n$ are formulas and C^n is an n-ary connective of the language ($n \geq 1$), then $C^n A_1 \cdots A_n$ is a formula of the language.

Using this general concept of a language, it is possible to give rigorous formulations of notions such as *expressibility of a truth-table in a language* and *expressive completeness*. But there is almost no place below where we need such a high degree of generality, and we will be able to proceed on the assumption that we are speaking only about one particular language at a time. Until Chapter VIII this will be the language of $\mathbf{H}_s$.

Exercises

1. Work out the truth-tables of the following formulas and determine whether

they are valid or invalid and satisfiable or not satisfiable. (For instance, $P \supset Q$ is invalid and satisfiable.)

(a) $P \supset \sim P \supset \sim P$	(b) $\sim(P \supset P)$
(c) $\sim\sim P \supset \sim P$	(d) $P \supset Q \supset P \supset P$
(e) $\sim P$	(f) $\sim(P \supset Q \supset . \sim Q \supset \sim P)$
(g) $P \supset Q \supset P \supset . \sim P \supset Q$	(h) $\sim(P \supset Q)$
(i) $\sim(P \supset Q) \supset P$	(j) $P \supset Q \supset R$
(k) $P \supset Q \supset R \supset . \sim P \supset R$	(l) $\sim(P \supset \sim(Q \supset R) \supset . P \supset Q)$

2. Show that every formula of $\mathbf{H}_s$ having the following forms is valid.

(a) $A \supset . B \supset A$	(b) $(A \supset . B \supset C) \supset . A \supset B \supset . A \supset C$
(c) $\sim A \supset \sim B \supset . B \supset A$	(d) $A \supset B \supset . A \supset \sim B \supset \sim A$
(e) $A \supset . A \supset B \supset B$	(f) $\sim A \supset . A \supset B$

3. In each of the following cases, determine whether every formula of $\mathbf{H}_s$ of the given sort is satisfiable. If every such formula is satisfiable, show that it is; if not, give an example of a formula having the given form which is not satisfiable.

(a) A	(b) $A \supset B$
(c) $A \supset B \supset A$	(d) $A \supset B \supset B \supset B$
(e) $A \supset A \supset A \supset A$	(f) $\sim A$

4. Show that for every **M** and every valuation V of **M**, either $V(A) = T$ or $V(\sim A) = T$. (If you wish, use M1.) Go on to show that for no such **M** and V does $V(A) = T$ and $V(\sim A) = T$.

5. Prove the following metatheorems (use any results stated prior to them in the text).

(a) M5	(b) M6	(c) M8
(d) M9	(e) M10	(f) M11
(g) M13	(h) M14	(i) M15
(j) M18	(k) M19	(l) M20
(m) M21	(n) M22	(o) M23

6. Show that every valuation of any morphology simultaneously satisfies the empty set $\varnothing$.

7. Let **M** and **M**′ be morphologies for $\mathbf{H}_s$, and let A be a formula of **M** and $A \supset B$ a formula of **M**′ such that every valuation of **M** satisfies A and every valuation of **M**′ satisfies $A \supset B$. Show that for all **M**″, if B is a formula of **M**″, then every valuation of **M**″ satisfies B.

8. Show that if A is a formula of both **M** and **M′**, then there is a proof of A consisting only of formulas of **M** if and only if there is a proof of A consisting only of formulas of **M′**.

9. Let Γ be a set of formulas of some morphology **M**, and let V be a valuation of **M**. Show that V simultaneously satisfies Γ if and only if V simultaneously satisfies every finite subset of Γ.

10. Decide the following questions by proofs or by counterexamples.

(a) If Γ is simultaneously satisfiable and Δ is simultaneously satisfiable, is $\Gamma \cup \Delta$ simultaneously satisfiable?
(b) If Γ and Δ are simultaneously satisfiable, is $\Gamma \cap \Delta$ simultaneously satisfiable?
(c) If Γ and Δ are simultaneously satisfiable, is $\{A \vee B \,/\, A \in \Gamma \text{ and } B \in \Delta\}$ simultaneously satisfiable?
(d) If Γ is simultaneously satisfiable, is $\{\sim A \,/\, A \in \Gamma\}$ simultaneously satisfiable?

11. Find a set Γ of formulas such that Γ is not simultaneously satisfiable, but for all members A and B of Γ, $\{A, B\}$ is simultaneously satisfiable.

12. Show that if A is not valid, then for any B, $A \supset B$ is satisfiable.

13. Show that if A is a formula of **M** such that any formula of **M** which results from A by simultaneous substitution is satisfiable, then A is valid.

14. Determine the truth-tables for conjunction and equivalence, as these connectives are defined in $\mathbf{H}_s$.

15. Give the truth-tables of the following formulas, and determine their status with regard to validity and satisfiability.

(a) $(P \vee Q) \supset P$
(b) $P \supset ((P \vee Q) \vee (P \vee \sim Q))$
(c) $(P \wedge Q) \wedge (R \equiv (R \supset (\sim P \vee \sim Q)))$
(d) $P \equiv (\sim P \supset P)$
(e) $((P \vee Q) \wedge R) \equiv ((P \wedge R) \vee (Q \wedge R))$
(f) $(P \wedge Q) \equiv P$
(g) $((P \wedge Q) \equiv P) \equiv Q$
(h) $(P \wedge Q) \vee ((\sim P \wedge Q) \vee (P \wedge \sim Q))$
(i) $(P \equiv Q) \equiv ((P \equiv R) \equiv R)$
(j) $\sim P \wedge Q$

16. Find a definition in $\mathbf{H}_s$ of the exclusive sense of disjunction.

17. Let $L_{\wedge\vee}$ be the language containing just two binary connectives, ∨ and ∧, standing for disjunction and conjunction, respectively. Show that $L_{\vee\wedge}$ is expressively incomplete.

18. Let $L_|$ consist of one binary connective |, which takes the following truth-table.

P	Q	$P \mid Q$
T	T	F
T	F	F
F	T	F
F	F	T

Show (using any results mentioned in the text) that $L_|$ is expressively complete.

Problems

1. Let $L_\equiv$ consist of one binary connective ≡, standing for equivalence. Show that $L_\equiv$ is expressively incomplete.
2. Show that a formula of $L_\equiv$ is valid if and only if each sentence parameter occurring in it occurs an even number of times.
3. Consider the following class-interpretation of the language of $\mathbf{H}_s$: a valuation V assigns to sentence parameters certain subsets of a set X, so that $V(P) \subseteq X$; $V(A \supset B) = \overline{V(A)} \cup V(B)$ and $V(\sim A) = \overline{V(A)}$, where $\overline{Y}$ is the set of members of X which are not in Y. Say that a formula A of **M** is *valid* if for all sets X and all valuations V of **M** in X, $V(A) = X$. Show that a formula is valid under the class-interpretation if and only if it is valid in the truth-functional sense.
4. Show that if all finite subsets of a set of formulas $\Gamma = \{A_1, A_2, \ldots\}$ are simultaneously satisfiable, then Γ is simultaneously satisfiable. (*Hint:* show that under the given conditions there exist morphologies $M_1, M_2, \ldots$ and valuations $V_1, V_2, \ldots$ such that for all n, (1) V_n is a valuation of M_n; (2) for all m, if $n < m$ then M_n is a submorphology of M_m and V_n the restriction of V_m to M_n; and (3) V_n simultaneously satisfies $\{A_1, A_2, \ldots A_n\}$. Define in terms of the sequence $V_1, V_2, \ldots$ a valuation V which simultaneously satisfies Γ.)
5. Develop the sketch given in Section 18 into a rigorous proof that the language of $\mathbf{H}_s$ is expressively complete.

VII Sentence Logic: Semantic Completeness

1. In the last chapter our discussion of $\mathbf{H_s}$ was almost exclusively semantic. We recalled in a few places that there is a deductive apparatus associated with $\mathbf{H_s}$, but at no point did we refer to this apparatus in developing definitions and metatheorems. Despite our neglect of this matter, however, it certainly looks as if there must exist close interrelationships of some sort between syntactic and semantic notions. Metatheorems VI.M17 to VI.M21, for instance, when compared with metatheorems V.M4 to V.M8, certainly suggest a connection between semantic implication and deducibility.

In fact, if we take a closer look at a special case of this and compare validity with provability, it begins to seem as if this relationship should be one of equivalence. It ought to be the case that every provable formula is valid and that every valid formula is provable. To see why this is so, let's try to establish a connection between validity and the notion of what a logical system is supposed to do. According to our intuitive idea of validity, a formula

is valid if it is true in any situation whatsoever; the truth of a valid formula is wholly independent of particular circumstances.

It's important to notice just how broad the conception of *situation* is that enters into this characterization of validity. According to the semantic theory of Chapter VI, which identified situations with valuations, situations are generated mechanically by assigning truth-values to sentence parameters; any such assignment generates a legitimate situation. This entails that the resulting conception of validity is exceedingly strict, so that most truths of common sense, of science, and even of mathematics will be rendered invalid. 'Water is wet', 'Force equals the product of mass and acceleration', and '2 + 2 = 4' will all be translated by formulas of $\mathbf{H_s}$ which are invalid; indeed, P is about the best translation we can manage of any of them. (This, of course, doesn't deny them status as truths, but means only that they cannot be regarded as valid in sentence logic.) However there will be some formulas, like $P \supset P$, which are valid even when situations are construed so broadly. These will be the formulas which, because of their structure as complexes built up by means of the connectives $\supset$ and $\sim$, are invariably true. The point of presenting things this way is to make it evident that the validity with which we are dealing is *logical* validity. A valid formula, in other words, is one that is true in any logically possible situation; an invalid formula is one that can be made false in some logically possible situation.

Let's relate this to the deductive apparatus of Chapter V. If this apparatus is to be at all successful in constituting a system of logic, it must not render any invalid formula provable. Something would be wrong with any logical theory that sanctioned in this way a formula that could be made false. Furthermore, a good logical theory ought to be powerful enough to generate *all* the valid formulas as theorems. It would be *incomplete* if there were some valid formula that could not be legitimized by the theory.

We arrive, then, at the following criterion for a deductive system of logic such as $\mathbf{H_s}$: $\mathbf{H_s}$ is an adequate system of logic in case for all formulas A (of any morphology), $\vdash A$ if and only if $\Vdash A$. The relationship we are demanding between provability and validity is thus a very intimate one; we are requiring that these two concepts determine precisely the same class of formulas.

The chief business of this chapter will be to prove that this requirement is met. This divides into two problems: to show that if $\vdash A$ then $\Vdash A$, and that if $\Vdash A$ then $\vdash A$. The first of these tasks is rather easy; the second is difficult, and requires a long argument. There exist many methods of carrying out this second proof; the one we will use is among the most powerful and flexible of these. The key idea of this method is a mediating concept having both a semantic and a syntactic side: the concept of a *saturated set* of formulas.

Using this amphibious notion we will be able to pass from validity to provability.

Much of the material in this chapter will be a meditation on saturated sets. Once we have unfolded enough of the properties of these sets, the result we want will appear as a corollary.

2. Intuitively, a (sententially) saturated set of formulas is a set having the structural properties of a set of truths in some situation. That is, we arrive at the notion of a saturated set by reflecting on the properties possessed by sets of formulas of **M** such that for some valuation V of **M**, $\Gamma = \{A \mid A$ is a formula of **M** and $V(A) = T\}$. Of the many properties one might expect of such a set, we need only single out two for purposes of definition, since these two properties suffice to characterize the notion we are seeking. One of these is *completeness with respect to negation* (a set Γ of formulas of **M** is complete with respect to negation if for all formulas A of **M**, $A \in \Gamma$ or $\sim A \in \Gamma$). The other is consistency. Notice that the latter property (see V.D11) is defined in terms of the deductive apparatus of $\mathbf{H}_s$.

D1. A set Γ of formulas of **M** *is* **M**-*saturated if:*

1. *For all formulas A of* **M**, *$A \in \Gamma$ or $\sim A \in \Gamma$;*
2. *Γ is consistent.*

From D1 a number of important characteristics of **M**-saturated sets can be developed; we begin by showing that **M**-saturated sets are deductively closed.

M1. If Γ is **M**-*saturated, A is a formula of* **M**, *and $\Gamma \vdash A$, then $A \in \Gamma$.*

PROOF. Suppose that Γ is **M**-saturated and that $\Gamma \vdash A$, and assume for *reductio ad absurdum* that $A \notin \Gamma$. Then by clause 1 of D1, $\sim A \in \Gamma$, so by V.M32, Γ would be inconsistent; but this contradicts clause 2 of D1. Therefore, $A \in \Gamma$.

Using V.M4, it is easy to strengthen M1 a bit.

M2. If Γ is **M**-*saturated and A is a formula of* **M**, *then $\Gamma \vdash A$ if and only if $A \in \Gamma$.*

Other characteristics of **M**-saturated sets which follow easily from the definition are the following. Their proofs are left as exercises.

M3. If Γ is **M**-*saturated and A is a formula of* **M**, *then $A \in \Gamma$ if and only if $\sim A \notin \Gamma$.*

M4. If Γ is **M**-*saturated and A and B are formulas of* **M**, *then $A \supset B \in \Gamma$ if and only if $A \notin \Gamma$ or $B \in \Gamma$.*

M5. If Γ is **M**-*saturated and $\Gamma \subseteq \Delta$, where Δ is a consistent set of formulas of* **M**, *then $\Gamma = \Delta$.*

In view of M5, an **M**-saturated set is a set of formulas of **M** that is as large as it can get without being inconsistent. For this reason saturated sets are sometimes called *maximal consistent* sets.

3. In this section we are finally going to take notice of the fact that the theorems of $\mathbf{H}_s$ are all valid formulas. This is easy to establish by an induction on length of proof in $\mathbf{H}_s$. (To carry out such an induction, it suffices to show that every axiom of $\mathbf{H}_s$ is valid and that if premisses of *modus ponens* are valid, then the conclusion is valid as well. Recall our discussion of this following the proof of V.M31.)

M6. If $\vdash A$ then $\Vdash A$.

PROOF. Using truth-tables, it's easy to see that every axiom of $\mathbf{H}_s$ is valid (see exercises VI.E2(a) to (c)). But also if $\Vdash A$ and $\Vdash A \supset B$ then $\Vdash B$ (see VI.E7). Therefore every theorem of $\mathbf{H}_s$ is valid.

Using M6 we can secure a number of more general results which relate syntactic and semantic notions. First, an analogue of M6 can be obtained relating deducibility and implication.

M7. If $\Gamma \vdash A$ then $\Gamma \Vdash A$.

PROOF. Suppose that $\Gamma \vdash A$. Then by V.M3, there is a finite subset Γ' of Γ such that $\Gamma' \vdash A$. Let $\Gamma' = \{B_1, \ldots, B_n\}$. By V.M10, $\vdash B_1 \supset. \cdots \supset. B_n \supset A$, and so, using M6, $\Vdash B_1 \supset. \cdots \supset. B_n \supset A$. Thus, by repeated uses of VI.M22, $\Gamma' \Vdash A$. Finally, by VI.M20, $\Gamma \Vdash A$.

Notice that M7 is a generalization of M6. M6 is just the special case of M7 in which $\Gamma = \varnothing$.

Using M7, we can get a result linking consistency and simultaneous satisfiability; its proof is left to you.

M8. If Γ is simultaneously satisfiable then Γ is consistent.

When in Section 2 we gave a rough-and-ready characterization of M-saturation, we said that an M-saturated set is one having the structural properties of a set of formulas true in some situation. At this point we can use M8 to cash in part of this idea in the form of a metatheorem.

M9. Let V *be a valuation of some morphology* **M**, *and let* Γ *be the set of formulas satisfied by* V; *i.e.*, $\Gamma = \{A \mid A$ *is a formula of* **M** *and* $V(A) = T\}$. *Then* Γ *is* **M**-*saturated.*

PROOF. By VI.E4, $B \in \Gamma$ or $\sim B \in \Gamma$ for all formulas B of **M**, and by M8 Γ is consistent. Hence, Γ is **M**-saturated.

4. Let's digress for a short time to reflect on what we've just done and explore some of its applications. What M6 shows is that $\{A \mid A$ is a formula of **M** and $\vdash A\} \subseteq \{A \mid A$ is a formula of **M** and $\Vdash A\}$; i.e., every provable formula is valid. Thus, if we agree that a valid formula, as we have defined validity, is a logically good formula, we know that every theorem of $\mathbf{H}_s$ is something worth proving in a system of logic. No theorem of $\mathbf{H}_s$, in other words, is undesirable.

This is sometimes expressed by saying that $\mathbf{H}_s$ has been shown to be *sound*. But if we wish to be more circumspect (and this is a good precaution to take), we will recall that M6 assumes a notion of validity that is dependent on the two-valued interpretation we developed in Chapter VI. If we had used *three* truth-values (say, T, F, and ?) or had changed the definition of validity in some other way, we might not have been able to get M6. For this reason it's better to say that $\mathbf{H}_s$ has been shown to be sound with regard to two-valued truth-tables, or with regard to its *intended interpretation*.

Now, if we read M6 negatively, it says that any formula that is *invalid* is *not* a theorem of $\mathbf{H}_s$. With this version of M6 we have at once an easy method of settling questions that have bothered us from time to time—questions as to whether certain formulas are not provable.

In Chapter V we developed elaborate techniques for showing that various formulas are provable in $\mathbf{H}_s$. Like the deduction theorem, these techniques all amount to ways of showing the existence of proofs in $\mathbf{H}_s$ by furnishing general methods for constructing these proofs. But when, for instance, the question arose in V.14 whether $P \supset \sim P$ was provable, we were stymied. We *could* show, using V.M31, that if $P \supset \sim P$ were provable, every formula would be provable. But there we had to let the matter rest, since at that time we had no method of showing the nonexistence of proofs. Certainly, manipulation of columns of formulas will not yield results of this sort. One has to look elsewhere to find some guarantee that no array of formulas can consti-

tute a proof in $\mathbf{H}_s$ of $P \supset \sim P$. And our semantic interpretation provides just this sort of touchstone.

For instance, if we let $\mathbf{M} = \{P\}$ and set $V(P) = T$, it turns out that $V(P \supset \sim P) = F$. Therefore $P \supset \sim P$ is an invalid formula and by M6 is not a theorem of $\mathbf{H}_s$; no column of formulas of $\mathbf{H}_s$, however long and complicated, can be a proof of $P \supset \sim P$.

In a more general vein, M7 can be used to show that certain things aren't *deducible* in $\mathbf{H}_s$, since this metatheorem guarantees that if A is not implied by Γ then A is not deducible from Γ. In order to show that $P \supset Q, Q \vdash P$ is not the case, for instance, all we have to do is note that P is not implied by Q and $P \supset Q$; so we need only find a valuation (say, of $\{P, Q\}$) which makes P false and $P \supset Q$ and Q true. Well then, let $V(P) = F$ and $V(Q) = T$—this does the trick.

Similarly, M8 can be used to show certain sets of formulas consistent. Take the same morphology $\mathbf{M} = \{P, Q\}$ as in the above example, and let Γ be the deductive closure of $\{P\}$; i.e., $\Gamma = \{A \mid A$ is a formula of **M** and $P \vdash A\}$. Now, the valuation V of **M** which makes P true and (say) Q false simultaneously satisfies Γ. We know this because V simultaneously satisfies $\{P\}$ and, by M7, $P \Vdash A$ for all $A \in \Gamma$, so that V must also satisfy every such A. Therefore Γ is simultaneously satisfiable, and so by M 8 is consistent. On the other hand, Γ is not **M**-saturated, since, for instance, neither $P \vdash Q$ nor $P \vdash \sim Q$, so that neither $Q \in \Gamma$ nor $\sim Q \in \Gamma$. This can be shown using M7 again; it is easy to see that neither $P \Vdash Q$ nor $P \Vdash \sim Q$, and M7 therefore entails that neither $P \vdash Q$ nor $P \vdash \sim Q$.

M9, however, allows us to construct many examples of sets that are saturated. This metatheorem guarantees, for instance, that the set $\Delta = \{A \mid A$ is a formula of **M** and $V(A) = T\}$ is **M**-saturated, where V is the valuation of the above example. By seeing whether or not V satisfies a given formula of **M**, we can easily check whether or not that formula is a member of Δ; formulas like P, $\sim Q$, $P \wedge \sim Q$, and $Q \supset \sim Q$ are members of Δ, while $\sim P$, $P \wedge Q$, and $\sim P \vee Q$ are not.

5. By definition, every **M**-saturated set is consistent. But it is by no means the case that every consistent set of formulas of **M** is **M**-saturated. As in the above examples, let $\mathbf{M} = \{P, Q\}$. Then we know by M8 that the set $\{P, \sim Q\}$ of formulas of **M** is consistent, but this set is clearly not complete with respect to negation and so isn't **M**-saturated. For instance, neither $P \supset Q$ nor $\sim(P \supset Q)$ is a member of $\{P, \sim Q\}$.

But suppose that we enlarge $\{P, \sim Q\}$; suppose, for instance, we take all the formulas of **M** which are deducible from this set. As it turns out, this deductive closure of $\{P, \sim Q\}$ is **M**-saturated. (It's consistent because $\{P, \sim Q\}$ is, and can be shown complete with respect to negation by an inductive argument. See E11 in this connection.)

Well, then, is it true in general that the set of formulas of **M** deducible from a consistent set will be **M**-saturated? No. For example, consider the set $\{P\}$ containing just P, and let $\Gamma = \{A \mid A$ is a formula of **M** and $P \vdash A\}$. We showed, in Section 4, that Γ isn't **M**-saturated because it contains neither Q nor $\sim Q$.

Nevertheless, it remains true that the set $\{P\}$ can be enlarged so as to obtain an **M**-saturated set. We can add $\sim Q$, for instance, and then as before take all the formulas of **M** deducible from this set. It is always true that a consistent set of formulas can be *extended to* (that is, can be enlarged to obtain) an **M**-saturated set. This is a fundamental property of saturated sets and is one of the most important steps in our proof of semantic completeness.

In carrying through a proof of this result, M11, we must venture very close to certain details of the theory of sets which are not likely to interest the nonspecialist. To keep these complications within bounds, we will assume in proving M11 that the formulas of **M** can be ordered by means of the positive integers, so that there is a list $A_1, A_2, A_3, \ldots$ of all the formulas of **M**. We will call this list the *alphabetical ordering* of the formulas of **M**.

In Chapter XIII we introduce a piece of terminology which enables us to state this assumption succinctly: if there is a list $A_1, A_2, A_3, \ldots$ of all the formulas of **M**, the set of formulas of **M** is said to be *denumerable*. The assumption that the set of formulas of **M** is denumerable may not appear to be very restrictive, and as a matter of fact most morphologies that we are apt to think of will meet this condition. However, as we will show in Chapter XIII, there do exist sets that are nondenumerably infinite—sets too large to be ordered by means of the positive integers. And if **M** itself is such a set it can be shown that the set of formulas of **M** will also not be denumerable. Although such morphologies are not covered by our proof of M11, a set-theoretic principle called the *axiom of choice* can be used to show that M11 holds for all morphologies whatsoever. We therefore will state M11 generally, though its proof for the nondenumerable case is left as a problem (P7) at the end of this chapter.

Our method of proving M11 is the one that naturally suggests itself. Given an arbitrary consistent set Γ, we try to obtain an **M**-saturated set from it by throwing formulas into it one at a time. Enlarging Γ in this way, we obtain a sequence $\Gamma_1, \Gamma_2, \Gamma_3, \ldots$ of consistent sets of formulas, where $\Gamma = \Gamma_1$. Since each stage of this sequence is obtained by adding a formula to the previous

stage, $\Gamma_i \subseteq \Gamma_j$ whenever $i \leq j$: earlier members of the sequence are subsets of later members.

Since for every morphology **M** the set of formulas of **M** is infinite, in general there will be no largest member of this sequence. Given any step Γ_i of the sequence, there may be a larger set Γ_j later on. We can't therefore obtain our **M**-saturated extension of Γ by talking about the "last" or "largest" step of the sequence. We can, however, collect together the formulas selected in the course of the sequence; this yields the set $\Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \cup \cdots$ of formulas of **M**. And if we construct the original sequence properly, this set will be complete with respect to negation. To show it consistent, and hence **M**-saturated, we appeal to the following metatheorem.

M10. Let $\Gamma_1, \Gamma_2, \Gamma_3, \ldots$ be any sequence of consistent sets of formulas of **M** *such that if $i \leq j$ then $\Gamma_i \subseteq \Gamma_j$, and let $\Delta = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \cup \cdots$. Then Δ is consistent.*

PROOF. We will show Δ consistent by proving that all its finite subsets are consistent. Let $\Theta = \{A_1, \ldots, A_n\}$ be any finite subset of Δ. Since $\Theta \subseteq \Delta$, there is for each A_k a set $\Gamma_{f(k)}$ such that $A_k \in \Gamma_{f(k)}$. Let m be the largest of the numbers $f(1), \ldots, f(n)$; since $\Gamma_i \subseteq \Gamma_j$ whenever $i \leq j$, it follows that $\Theta \subseteq \Gamma_m$. Then Θ is consistent, because Γ_m is consistent. In general, then, any finite subset of Δ is also a subset of some Γ_i and so is consistent. But then Δ is consistent, in view of V.M33.

We can now proceed with the metatheorem we first set out to prove.

M11. Any consistent set of formulas of **M** *has at least one* **M**-*saturated extension.*

PROOF. Let Γ be a consistent set of formulas of some morphology **M**, and let $A_1, A_2, A_3, \ldots$ be the alphabetical ordering of the formulas of **M**. Define a sequence $\Gamma_1, \Gamma_2, \Gamma_3, \ldots$ of sets inductively, by letting $\Gamma_1 = \Gamma$ and continuing the sequence according to the following rule.

> Let $\Gamma_{i+1} = \Gamma_i \cup \{A_i\}$ if $\Gamma_i \cup \{A_i\}$ is consistent, and let $\Gamma_{i+1} = \Gamma_i \cup \{\sim A_i\}$ otherwise.

The set Γ_{i+1} is thus constructed from Γ_i by testing the alphabetically ith formula A_i of **M** for consistency with Γ_i. If this formula can be added consistently to Γ_i, Γ_{i+1} is obtained by doing so. If A_i cannot be added consistently to Γ_i, then $\sim A_i$ is added to Γ_i.

We will use induction on i to show that for every i, the set Γ_i is consistent. The fact that Γ_1 (i.e., Γ) is consistent furnishes the basis step of the induction.

Suppose as inductive hypothesis that Γ_i is consistent. Then by V.M35 either $\Gamma_i \cup \{A_i\}$ is consistent or $\Gamma_i \cup \{\sim A_i\}$ is consistent. Now, if $\Gamma_i \cup \{A_i\}$ is consistent then Γ_{i+1} by definition is $\Gamma_i \cup \{A_i\}$ and so is consistent. If $\Gamma_i \cup \{A_i\}$ is not consistent, then $\Gamma_i \cup \{\sim A_i\}$ is consistent and is Γ_{i+1} by definition, and so again Γ_{i+1} is consistent. In either case, Γ_{i+1} is consistent, and the induction is complete.

Let $\Delta = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3 \cup \cdots$, and let B be any formula of **M**; B must appear in the alphabetical ordering of the formulas of **M**, and so is A_j for some j. We have defined things so that either $B \in \Gamma_{j+1}$ or $\sim B \in \Gamma_{j+1}$, and thus $B \in \Delta$ or $\sim B \in \Delta$. The set Δ is therefore complete with respect to negation. Since Δ meets the conditions of M10 it also is consistent and so is **M**-saturated. We have arranged things from the beginning so that $\Gamma \subseteq \Delta$, and Δ is therefore an **M**-saturated extension of Γ, as desired. This completes the proof of M11.

Since every **M**-saturated set is consistent by definition, we can make M11 a bit stronger.

M12. A set of formulas of **M** *is consistent if and only if it has an* **M**-*saturated extension.*

6. Continuing our investigation of **M**-saturated sets, let's return to the original intuitive characterization we gave of them in Section 2, where we said that **M**-saturated sets were to be like sets that consist of those formulas true in some situation. In M9 we were able to make good a part of this characterization by showing that every set of formulas which is the set of formulas made true by some valuation of **M** is **M**-saturated. But if our definition of **M**-saturation is really successful, the converse of M9 should also hold: it should be true that every **M**-saturated set of formulas can be characterized as the set of formulas made true by some valuation or other of **M**.

Our next metatheorem establishes that this is so. The central idea of its proof is to use the properties of an **M**-saturated set itself in order to define a valuation of **M** which makes true every formula in the given set.

M13. Let Γ be an **M**-*saturated set of formulas of* **M**. *Then there exists a valuation* V *of* **M** *such that* $\Gamma = \{A \mid A$ *is a formula of* **M** *and* $V(A) = T\}$.

PROOF. Let V be defined by letting $V(P) = T$ if $P \in \Gamma$ and $V(P) = F$ if $P \notin \Gamma$.

This determines the valuation V exhaustively, since all there is to a valuation of **M** is an assignment which gives truth-values to sentence parameters

in **M**. Therefore, in view of VI.D3, the truth-value $V(A)$ assigned to any formula A of **M** by V is also determined, so that these values are now beyond our control. We can find things out about them, but can't change them. We will now go on to discover that, in fact, satisfaction by V coincides with membership in Γ for all formulas of **M**, however complex; that is, for all formulas A of **M**, $V(A) = T$ if and only if $A \in \Gamma$.

Since our definition of V guarantees that this is so in case A is a sentence parameter of **M**, an inductive argument suggests itself. For the basis step of the induction, we want to show that for all $P \in$ **M**, $V(P) = T$ if and only if $P \in \Gamma$. And this holds because that is the way we defined V.

In the inductive step, we assume that for all formulas B less complex than A, $V(B) = T$ if and only if $B \in \Gamma$. We want to show that $V(A) = T$ if and only if $A \in \Gamma$. Here, there are two cases: either A is an implication or a negation.

First, then, if A is $B \supset C$, we know by VI.D3 that $V(A) = T$ if and only if $V(B) = F$ or $V(C) = T$. But by VI.M1 and the hypothesis of induction, $V(B) = F$ if and only if $B \notin \Gamma$ and $V(C) = T$ if and only if $C \in \Gamma$. Hence, $V(B) = F$ or $V(C) = T$ if and only if $B \notin \Gamma$ or $C \in \Gamma$. But by M4 this holds if and only if $B \supset C \in \Gamma$; i.e., if and only if $A \in \Gamma$. Therefore, $V(A) = T$ if and only if $A \in \Gamma$.

Second, if A is $\sim B$, we know by VI.D3 that $V(A) = T$ if and only if $V(B) = F$. But by VI.M1 and the hypothesis of induction, $V(B) = F$ if and only if $B \notin \Gamma$. But by M3, this holds if and only if $\sim B \in \Gamma$; i.e., if and only if $A \in \Gamma$. Therefore, again $V(A) = T$ if and only if $A \in \Gamma$.

So in both cases we have $V(A) = T$ if and only if $A \in \Gamma$, and our inductive argument is complete. We have shown that for all formulas A of **M**, $V(A) = T$ if and only if $A \in \Gamma$. But this means that $\Gamma = \{A \mid A$ is a formula of **M** and $V(A) = T\}$; thus, there is a valuation of **M** (namely, V) such that Γ is the set of formulas of **M** made true by V. And this is what was to be shown.

Putting together M 9 and M13, we have the following necessary and sufficient semantic condition for **M**-saturation.

M14. A set Γ of formulas of **M** *is* **M**-*saturated if and only if there exists some valuation* V *of* **M** *such that Γ is the set of formulas of* **M** *satisfied by* V: $\Gamma = \{A \mid A$ *is a formula of* **M** *and* $V(A) = T\}$.

7. Now let's return to the problem of semantic completeness and try to locate ourselves with respect to the overall argument. In Section 4, we showed that every formula provable in $\mathbf{H_s}$ is valid, so that every theorem of $\mathbf{H_s}$ is something we want to be able to prove. On the other hand, it would be nice

to know whether every formula we want to be able to prove is in fact provable: is every valid formula a theorem of $\mathbf{H}_s$? This is called the problem of *completeness*, because if some valid formula weren't provable in $\mathbf{H}_s$ there would then be something partial and inadequate about our formulation of the system. It would be *incomplete* with regard to its semantic interpretation. If this were so, we'd have to reformulate $\mathbf{H}_s$—perhaps by adding more axioms —in order to be able to prove all the valid formulas.

The question of soundness (of a deductive system, with respect to a semantic interpretation) thus amounts to whether we can prove *too much*; i.e. whether we can prove any formulas that are invalid with respect to the interpretation. The question of semantic completeness, on the other hand, amounts to whether we can prove *enough*; i.e., whether we can prove all the valid formulas.

We now want to know whether $\mathbf{H}_s$ is semantically complete with respect to its intended interpretation, and so we ask whether all valid formulas A (of some arbitrary **M**) are also provable. If we were to seek a direct proof of this, we would try to use the fact that A is valid to show that A is provable. We would thus look for a general method for turning a truth-table for A which takes only T's on its right-hand side into a proof of A in $\mathbf{H}_s$. It is possible to formulate such a method, although the details are rather complicated and tedious.

An indirect method of proof, on the other hand, turns out to be more fruitful. In this approach we try to show that if A is not provable in $\mathbf{H}_s$ it is invalid. This suggests the following sort of argument. First, try to devise a systematic procedure of trying to find a proof in $\mathbf{H}_s$ of an arbitrary formula A. Then show that if this procedure fails to work, there is a valuation that makes A false. Actually, systems of natural deduction lend themselves to this approach better than Hilbert-style systems such as $\mathbf{H}_s$. This is indicated by the fact that in the natural-deduction systems one has a good idea, just from looking at a formula A, of what a proof of A must look like if there is a proof at all. And this sort of demonstration of semantic completeness can be made to work for the systems of Chapter IV (see P1 for hints).

But we'll use still another argument, one that turns on the metatheorem we proved in the last section. This argument uses only results that we have already obtained: M13, in combination with M11, enables us to show that any formula of $\mathbf{H}_s$ that is not provable has a falsifying valuation. The idea is this: consider a formula A (of some morphology **M**) which is not provable in $\mathbf{H}_s$. Then $\{\sim A\}$ is consistent. (This follows from the special case of V.M34 in which Γ is $\varnothing$.) Then, by M11, $\{\sim A\}$ has an **M**-saturated extension Δ. But by M13, there is a valuation V of **M** which satisfies those and only those

formulas of **M** which are in Δ. So V doesn't satisfy A, since A is not in Δ (if A were in Δ, this set would be inconsistent). So A is invalid. Hence, every formula that is not provable is invalid, and therefore (finally) every valid formula is provable. This, in a nutshell, is our proof of semantic completeness. In the next section we will present this proof more generally and so obtain completeness as a corollary of a stronger result.

8. The first metatheorem of this section is the converse of M8.

M15. Let Γ be any set of formulas of a morphology **M**. *If Γ is consistent, then Γ is simultaneously satisfiable.*

PROOF. Suppose that Γ is consistent. By M11, Γ has an **M**-saturated extension Δ. By M13, there is a valuation V of **M** such that $\Delta = \{A \mid A$ is a formula of **M** and $V(A) = T\}$. Now, if $B \in \Gamma$ then $B \in \Delta$, and hence $V(B) = T$. So V satisfies every formula in Γ; i.e., simultaneously satisfies Γ. Therefore Γ is simultaneously satisfiable.

Together with M8, M15 establishes that consistency and simultaneous satisfiability are equivalent in a very strong sense; these notions apply to precisely the same sets of formulas. These two notions are therefore, at bottom, different ways of looking at the same thing. And M16, below, is thus our first result stating the equivalence of syntactic and semantic concepts.

M16. Let Γ be any set of formulas of a morphology **M**. *Γ is consistent if and only if Γ is simultaneously satisfiable.*

Using M16, we can go on to verify similar relationships between other pairs of syntactic and semantic concepts. First, let's make good the "analogy" discussed in VI.15 between deducibility and implication by showing that these are also equivalent.

M17. Let Γ be any set of formulas of a morphology **M** *and A be any formula of* **M**. *Then $\Gamma \vdash A$ if and only if $\Gamma \Vdash A$.*

PROOF. By V.M34, $\Gamma \vdash A$ if and only if $\Gamma \cup \{\sim A\}$ is inconsistent. But by M16, $\Gamma \cup \{\sim A\}$ is inconsistent if and only if $\Gamma \cup \{\sim A\}$ is not simultaneously satisfiable. And by VI.M17, $\Gamma \cup \{\sim A\}$ is not simultaneously satisfiable if and only if $\Gamma \Vdash A$. Putting these together, it follows that $\Gamma \vdash A$ if and only if $\Gamma \Vdash A$, and M17 is proved.

When we proved M7, we already obtained half of M17. What is new about

M17 is the following piece of information, which is sufficiently important to record separately as a metatheorem.

M18. If $\Gamma \Vdash A$ *then* $\Gamma \vdash A$.

This metatheorem states that if $\Gamma \Vdash A$, then there exists a deduction in $\mathbf{H_s}$ of A from Γ. But intuitively, to say that $\Gamma \Vdash A$ is to say that A is true in every situation in which all the members of Γ are true; and, surely, this means that (in a semantic sense) A is entailed by any theory in which all the members of Γ are postulated. What M18 tells us is that whenever A is a consequence of Γ in this sense, there is a deduction in $\mathbf{H_s}$ of A from Γ. According to M18, then, every argument that is worth deducing in $\mathbf{H_s}$ *can* be deduced in $\mathbf{H_s}$. Thus, what is established here is a sort of completeness, more general than the sort we discussed above in Section 7. Sometimes this strong kind of completeness is called *completeness as to consequences*, or *argument-completeness*. We will simply call it *strong semantic completeness*. M18 shows the strong semantic completeness of $\mathbf{H_s}$ with respect to its intended interpretation. In contrast with this, a system is said to be *weakly complete* relative to an interpretation if every formula of the system which is valid with respect to the interpretation is provable. There exist systems of logic which are weakly complete with respect to certain interpretations, but not strongly complete with respect to these interpretations (see XII.P1,below). Hence, it's important to distinguish these two notions of completeness.

Finally, the equivalence of provability in $\mathbf{H_s}$ with validity (M19) and the weak semantic completeness of $\mathbf{H_s}$ (M20) can be obtained as special cases of M17 and M18.

M19. $\vdash A$ *if and only if* $\Vdash A$.

PROOF. By M17, $\varnothing \vdash A$ if and only if $\varnothing \Vdash A$. Therefore, by V.M2 and VI.M18, $\vdash A$ if and only if $\Vdash A$.

M20. If $\Vdash A$ *then* $\vdash A$.

Since in Chapter V we showed directly that the natural deduction system $S_{\supset\sim}$ is equivalent to $\mathbf{H_s}$, the results of this chapter apply also to the former system. In particular, using M17 and V.M40, we have the following metatheorem.

M21. $\{A_1, \ldots, A_n\} \Vdash B$ *if and only if* $A_1, \ldots, A_n \vdash_{S_{\supset\sim}} B$.

9. Besides being an interesting and important result (or cluster of results) in its own right, the semantic completeness of $\mathbf{H_s}$ is also useful in solving other problems concerning $\mathbf{H_s}$ and its intended interpretation. In this section we'll discuss several of these applications.

One semantic problem that we have not yet settled is the question of compactness raised in VI.14. We are now in a position to show that $\mathbf{H_s}$ is indeed compact with respect to its intended interpretation.

M22. (Compactness). A set Γ *of formulas is simultaneously satisfiable if and only if every finite subset of* Γ *is simultaneously satisfiable.*

PROOF. By M16, Γ is simultaneously satisfiable if and only if Γ is consistent. By V.M33, Γ is consistent if and only if every finite subset of Γ is consistent. And again by M16, every finite subset of Γ is consistent if and only if every finite subset of Γ is simultaneously satisfiable. Hence, Γ is simultaneously satisfiable if and only if every finite subset of Γ is simultaneously satisfiable, as desired.

Using M22, it is easy to get the following result, which is another form of compactness. Its proof is left as an exercise.

M23. If $\Gamma \Vdash A$, *then* $\Gamma' \Vdash A$ *for some finite subset* Γ' *of* Γ.

As another example of how to apply semantic completeness, we will show how to use it to obtain the *interpolation theorem* for $\mathbf{H_s}$. This theorem says that if $\vdash A \supset B$, then if A and B share any sentence parameters there is a formula C containing only parameters common to A and B and such that $\vdash A \supset C$ and $\vdash C \supset B$. (We leave it as an exercise to show that if A and B share no parameters and $\vdash A \supset B$, then $\vdash {\sim}A$ or $\vdash B$.) Interpolation theorems are very important results in their own right, and have many uses in advanced areas of logic.

M24. If $\vdash A \supset B$ *and there are some parameters occurring in both* A *and* B, *then there is a formula* C *which contains only parameters common to* A *and* B *and such that* $\vdash A \supset C$ *and* $\vdash C \supset B$.

PROOF. Let $\mathbf{M}$ be the set of parameters occurring in A, and $\mathbf{M}'$ be the set of parameters occurring in B. By hypothesis, $\mathbf{M} \cap \mathbf{M}' \neq \varnothing$, so let $\mathbf{M} \cap \mathbf{M}' = \{Q_1, \ldots, Q_n\}$; also, let $\mathbf{M} \cup \mathbf{M}'$ be $\mathbf{M}''$. Since $\mathbf{M}''$ is finite, there is only a finite number of valuations of $\mathbf{M}''$ and hence there are only finitely many of these valuations which satisfy A; let these be $V_1, \ldots, V_k$. (If there are *no* such valuations, then ${\sim}A$ is valid, and hence by M18, $\vdash {\sim}A$. We can therefore take

C to be $Q_1 \wedge \sim Q_1$, so that we will have $\vdash A \supset C$ and $\vdash C \supset B$. Thus we can suppose without loss of generality that there is at least one valuation of $\mathbf{M}''$ satisfying A.) Use these valuations V_i in the following way to construct a formula C: let $D_1{}^i$ be Q_1 if $V_i(Q_1) = T$ and $\sim Q_1$ if $V_i(Q_1) = F$; let $D_2{}^i$ be Q_2 if $V_i(Q_2) = T$ and $\sim Q_2$ if $V_i(Q_2) = F$, and so forth. Let C^i be $D_1{}^i \wedge \cdots \wedge D_n{}^i$, and finally take C to be $C^1 \vee \cdots \vee C^k$. The formula C has been constructed (as you may verify) so that all of the valuations V_1 through V_k satisfy C; hence, for all valuations V of $\mathbf{M}''$, $V(C) = T$ if $V(A) = T$, and so $A \supset C$ is valid. Therefore, by M18, $\vdash A \supset C$.

Now, all that is left to be shown is that $\vdash C \supset B$. Suppose for *reductio* that for some valuation V of $\mathbf{M}'$, $V(C \supset B) = F$; then $V(C) = T$ and $V(B) = F$. Since $V(C) = T$, $V(C^i)$ must be T for some i, and hence for all j, $1 \leq j \leq n$, $V(Q_j) = V_i(Q_j)$. That is, V coincides on $\mathbf{M} \cap \mathbf{M}'$ with V_i. Now let V' act like V on $\mathbf{M}'$, and like V_i on $\mathbf{M} - \mathbf{M}'$; that is, let $V'(P) = V(P)$ if P is in $\mathbf{M}'$, and let $V'(P) = V_i(P)$ if P is in $\mathbf{M}$ but not in $\mathbf{M}'$. The valuations V' and V_i are identical on $\mathbf{M}$, and so, by VI.M2, V' satisfies A if and only if V_i does, since A is a formula of $\mathbf{M}$. But V_i was chosen so that $V_i(A) = T$; hence, $V'(A) = T$. And by the same reasoning, since V' and V coincide on $\mathbf{M}'$ and B is a formula of $\mathbf{M}'$, $V'(B) = F$. But this is impossible, since by assumption $A \supset B$ is valid.

Therefore for every valuation V of $\mathbf{M}'$, $V(B) = T$ if $V(C) = T$; hence, $C \supset B$ is valid. By M6, $\vdash C \supset B$, and our proof is finished.

We've saved the most obvious application of the completeness results for last. This, of course, is their use in deciding whether various formulas of $\mathbf{H}_s$ are provable or not. Since M19 tells us that validity is a necessary and sufficient condition of provability, we need only check the truth-table of a formula in order to see whether or not it's provable in $\mathbf{H}_s$ or, for that matter, in $\mathbf{S}_{\supset\sim}$. When confronted, for instance, with a formula such as $\sim(P \equiv Q) \supset . P \supset Q \supset Q$, we no longer need to try to find a proof of it in $\mathbf{H}_s$ or even to show by means of the metatheorems of Chapter V that there is a proof of it. All we need do is to verify that this formula is valid and at once we know it's provable in $\mathbf{H}_s$. Similarly, when we are given $\sim(P \equiv Q) \supset . P \supset Q \supset P$, we check its truth-table and, finding it invalid, we know it isn't provable in $\mathbf{H}_s$.

It is generally easier to test for validity by means of truth-tables than to try to show more directly that the given formula is provable. But besides being easier, the semantic method has a far greater advantage: in case a formula is *not* provable, the truth-table test will show that it isn't provable. The syntactic methods of Chapter V are no good for this at all. They only work when one wants to show of a provable formula that it does, in fact, have a proof. If in the exercises for Chapters IV and V we had mixed in unprovable formulas with provable ones and asked you to prove the ones that could be proved, you'd have had a hard time, unless you already knew something about truth-tables.

What the method of truth-tables gives us, then, is a mechanical or automatic way of deciding, in a finite number of steps, whether any given formula of $\mathbf{H}_s$ is provable or not. The procedure is mechanical because the rules for constructing truth-tables are completely explicit and leave no room for ingenuity. And it halts in a finite number of steps because every formula of $\mathbf{H}_s$ by definition contains a finite number of parameters, and a truth-table for n parameters will contain 2^n rows. Thus, every truth-table of a formula of $\mathbf{H}_s$ is finite. Such a method is called a *decision procedure*, and a system is said to be *decidable* if there exists a decision procedure for it. We have, then, the following metatheorems.

M25. $\mathbf{H}_s$ *is decidable.*

M26. $\mathbf{S}_{\supset\sim}$ *is decidable.*

These metatheorems may seem less useful to you than the actual procedure itself, which certainly is a handy thing to have in working with these systems. Nevertheless, the mere existence of such procedures is of interest, because some well-known systems of logic can be shown to be *undecidable*. Thus M25 and M26 do serve to contrast $\mathbf{H}_s$ and $\mathbf{S}_{\supset\sim}$ with other systems that have no decision procedures.

10. Considering the fact that checking for validity is a much easier way of telling whether or not something is provable, you may have begun to wonder why we went to so much trouble in previous chapters to develop the syntactic notions of provability and deducibility. Don't M19 and M17 show that these notions are redundant? It's almost as if we were taking pains to make the study of logic difficult by treating everything in two different ways and by choosing to discuss the harder way first.

Well, it *is* true, of course, that it would have been much easier to discuss sentence logic in terms of truth-tables alone. And, if the purpose of everything we have done up to now had been only to acquaint you with this sort of logic, that would probably have been a better way to do it. But all along our aim has been to do much more than just this. We've tried to do the theory of classical sentence logic in such a way that the techniques and results we developed in the course of our investigation will apply to other areas of logic. In other words, all this material has really been an introduction to the concepts and methods used by modern logicians, as applied to just one kind of logical system. There really would have been no point in devoting so many pages to classical sentence logic otherwise; as the truth-table technique shows, the whole thing is pretty simple. But precisely because it *is* so simple, it's a good example to cut one's logical teeth on.

This does not yet settle the question of why we need two distinct approaches which turn out to characterize the same thing. We can begin to answer this question by saying that logicians really do use both syntactic and semantic techniques, so that we must discuss both in order to give a comprehensive account of the concepts actually employed by logicians. But this doesn't go far enough; we should try to explain *why* both approaches are used.

One thing that necessitates the use of both is the fact that in some cases they are *not* equivalent. For instance, in so-called *higher-order* logics—logics in which one can quantify over predicates and so say things such as 'John has all the qualities of a good executive'—it turns out that there must be a discrepancy between any notion of provability remotely resembling the provability of $\mathbf{H_s}$ and the corresponding notion of validity. These systems are semantically incomplete. In situations of this sort one *has* to use both methods, since they give different results.

But even in cases where one has semantic completeness, there is some point in doing logic both ways. Even though they are equivalent, the two approaches are different enough so that they complement one another. Each furnishes valuable sources of insight into the subject that would be lacking if only one method were used. Thus, in M23 we employed a syntactic result to obtain a semantic metatheorem, and in M24 we used semantic methods to obtain a syntactic result. Direct proofs of either of these metatheorems would not have been easy. So the result is mutual support rather than redundancy. (Compare this to the situation in quantum physics, where wave mechanics and matrix mechanics complement each other in a similar way). Together, the semantic and the syntactic approaches provide much more stability to a logical theory than either would afford by itself.

Exercises

1. Check the following with regard to provability in $\mathbf{H_s}$.

   (a) $\sim P \vee (\sim P \supset (P \vee Q))$
   (b) $(P \equiv Q) \equiv (R \equiv (R \equiv Q))$

   (c) $((P \equiv Q) \equiv (R \equiv (R \equiv Q))) \equiv P$
   (d) $(P \wedge (Q \wedge (R \vee P))) \equiv ((P \wedge Q) \wedge R)$
   (e) $\sim(P \vee (P \supset Q)) \supset \sim R$
   (f) $(P \equiv (P \vee Q)) \vee Q$

2. Decide the following statements, one way or another.

   (a) $\{P \supset Q, P \equiv \sim R\} \vdash R \vee Q$
   (b) $\{P \vee (Q \wedge R), P \equiv \sim Q\} \vdash R$
   (c) $\{P \wedge (Q \vee R), \sim(R \vee P)\} \vdash \sim Q$
   (d) $\{P \equiv \sim Q\} \vdash \sim(P \equiv Q)$
   (e) $\{P \supset Q, \sim Q \supset R, R \supset \sim P\} \vdash \sim P$
   (f) $\{P \vee (Q \wedge (R \vee S))\} \vdash (P \vee Q) \wedge (P \vee R)$

3. Check the following with regard to consistency in $\mathbf{H_s}$.

   (a) $\{P, Q \supset P, \sim Q \supset \sim R, R\}$
   (b) $\{\sim(P \supset (Q \vee P)), P \equiv R\}$
   (c) $\{\sim\sim P, \sim(R \supset P)\}$
   (d) $\{A \supset B \mid A, B$ are formulas of $\{P, Q\}\}$
   (e) $\{A \vee B \mid A, B$ are formulas of $\{P, Q\}$ and $P \vdash A$ and $Q \vdash B\}$
   (f) $\{A \mid A$ is a formula of $\{P, Q\}$ and not $\vdash A\}$

4. Is there a derivation in $\mathbf{S}_{\supset\sim}$ of P from $P \vee Q$? Give reasons for your answer.

5. Show that a set Γ of formulas of **M** is simultaneously satisfiable if and only if Γ has at least one **M**-saturated extension. Use any metatheorems proved in this chapter.

6. Choose an appropriate formula of some morphology and show that neither it nor its negation is provable in $\mathbf{H_s}$.

7. Show that if a formula A is provable in $\mathbf{H_s}$, then $\sim A$ is not provable in $\mathbf{H_s}$.

8. Prove the following metatheorems (use any results established previously).

   (a) M3 (b) M4 (c) M5 (d) M8 (e) M23

9. Settle the following questions by proofs or counterexamples.

   (a) If $\vdash A \vee B$ then $\vdash A$ or $\vdash B$.
   (b) If A and B are formulas of **M** and $\vdash A \equiv B$, then, where V_1 and V_2 are any valuations of **M**, $V_1(A) = V_2(B)$.
   (c) If A is satisfiable, then any substitution instance of A is satisfiable.
   (d) A is satisfiable if and only if not $\vdash \sim A$.
   (e) Let **X** be a set consisting of sets of formulas of **M**, such that for all $\Gamma, \Delta \in \mathbf{X}$, $\Gamma \subseteq \Delta$ or $\Delta \subseteq \Gamma$. Let $\Theta = \{A \mid$ for some $\Gamma \in \mathbf{X}, A \in \Gamma\}$. Then

Θ is simultaneously satisfiable if every set in X is simultaneously satisfiable.

10. Show that if $\Gamma \vdash A$ (where Γ is a set of formulas of **M** and A a formula of **M**) and V is a valuation of **M** which simultaneously satisfies Γ, then V satisfies A.

11. Let Γ be a set of formulas of **M** which is deductively closed (if $\Gamma \vdash A$ and $A \in$ **M** then $A \in \Gamma$). Show if for all parameters P of **M** either $P \in \Gamma$ or $\sim P \in \Gamma$ (and not both), then Γ is **M**-saturated. (*Hint:* Use induction on the complexity of A to show that for all formulas A of **M**, $A \in \Gamma$ or $\sim A \in \Gamma$.)

12. Show that a set Γ of formulas of **M** is **M**-saturated if and only if (1) Γ is deductively closed; (2) for some formula A of **M**, $A \notin \Gamma$; and (3) for all formulas A and B of **M**, if $A \vee B \in \Gamma$ then $A \in \Gamma$ or $B \in \Gamma$.

13. Let **M** $= \{P, Q, R\}$. Prove M11 for **M**, without using any results of this chapter (i.e., show directly that any consistent set Γ of formulas of **M** has an **M**-saturated extension. The fact that **M** is finite in this special case makes possible a much simpler proof than the one given in the test). *Hint:* See E11.

14. Use compactness and weak completeness to obtain strong completeness (i.e., using M22 or M23 and M20 and results of previous chapters, prove M18).

15. Let V_1 and V_2 be valuations of **M** and let $\Gamma = \{A \mid A$ is a formula of **M** and $V_1(A) = T$ and $V_2(A) = T\}$. Show that if $\Gamma \vdash A$ and A is a formula of **M**, then $A \in \Gamma$.

16. Show that if $\vdash A \supset B$ and A and B share no sentence parameters, then either $\vdash \sim A$ or $\vdash B$.

17. Let the language of $\mathbf{H}_{\supset}$ consist only of the binary connective $\supset$, and let $\mathbf{H}_{\supset}$ have the following axiom-schemes.

$$A \supset . B \supset A$$
$$(A \supset . B \supset C) \supset . A \supset B \supset . A \supset C$$
$$A \supset B \supset A \supset A$$

The one rule of inference of $\mathbf{H}_{\supset}$ is *modus ponens*. Show, by an argument similar to the one given in this chapter, that $\mathbf{H}_{\supset}$ is semantically complete in the strong sense: if $\Gamma \Vdash_{\mathbf{H}_{\supset}} A$, then $\Gamma \vdash_{\mathbf{H}_{\supset}} A$.

Problems

1. Give a direct proof that if $A_1, \ldots, A_n \Vdash B$, where $A_1, \ldots, A_n$, and B are formulas of $S_{\vee\sim}$, then $A_1, \ldots, A_n \vdash_{S_{\vee\sim}} B$. *Hint:* Devise a systematic procedure for

eliciting a contradiction from $\{A_1, \ldots, A_n, \sim B\}$ using dis elim, neg elim, reit, and the procedure of obtaining $\sim C$ and $\sim D$ by subordinating derivations

$$\begin{array}{|l} \underline{C} \\ \sim(C \vee D) \\ C \vee D \end{array} \quad \text{and} \quad \begin{array}{|l} \underline{D} \\ \sim(C \vee D) \\ C \vee D \end{array}$$

wherever one has a step $\sim(C \vee D)$. Show if this procedure succeeds, there is a derivation of B from $A_1, \ldots, A_n$ in $S_{\vee\sim}$ and that if it fails, there is a valuation falsifying $(A_1 \wedge \cdots \wedge A_n) \supset B$.

2. Use the above result to show directly that any valid formula is a theorem $S_{\supset\sim}$.

3. Show (using any results established in the text) that $\mathbf{S}_s$ is semantically complete.

4. Show that if not $\vdash A$ then the system $\mathbf{H}_s^A$ obtained from $\mathbf{H}_s$ by adding every substitution instance of A as an axiom is *absolutely inconsistent*. That is, show that any formula whatsoever is provable in $\mathbf{H}_s^A$.

5. Let L be a language, say with two connectives, one 1-ary connective N and one 2-ary connective O. L is to be interpreted by correlating N and O with operations in a *matrix* for L. Such a matrix is a nonempty set M, together with two operations (or functions) f and g (f taking members of M into members of M, and g taking pairs of members of M into members of M), and a subset D of M (the *designated* elements of M). The notion of a valuation in M of a morphology and of the value V(A) assigned to a formula A of a morphology M by a valuation V of M are defined by generalizing the corresponding definitions in Chapter VI. A formula A of M is satisfied by a valuation V of M if the value V(A) is designated, and is *valid* if it is satisfied by every valuation of M. Let Γ be a set of formulas of L (Γ may be thought of as a set of "theorems"). A matrix M for L is said to be *characteristic* for Γ if for all formulas A of L, A is valid in M if and only if $A \in \Gamma$.
   Show that if Γ is closed under substitution (i.e., if whenever $A \in \Gamma$ and B is a substitution instance of A, then $B \in \Gamma$) then there is some characteristic matrix for Γ.
   (*Hint:* Define a relation $\simeq$ of synonymy relative to Γ, as in V.E18, and show that $\simeq$ is an equivalence relation. Let the equivalence classes of this relation be the elements of a matrix M, and define operations and a set of designated elements of this matrix so that it characterizes Γ.)

6. Show that the system $\mathbf{S}_0$ of Chapter I is semantically *incomplete*; i.e., find a valid formula of $\mathbf{S}_0$ that is not provable in $\mathbf{S}_0$.

7. Where X is a set of sets, a *chain* in X is a subset Y of X such that for all U, V $\in$ Y, U $\subseteq$ V, or V $\subseteq$ U. A chain Y of X is *maximal in* X if for all chains U of X, U = Y if Y $\subseteq$ U. The *Hausdorff maximal principle* states that every set X of sets possesses at least one chain which is maximal in X; this principle