PERSPECTIVES IN LOGIC

Stephen Cook
Phuong Nguyen

LOGICAL FOUNDATIONS OF PROOF COMPLEXITY

CAMBRIDGE

CAMBRIDGE www.cambridge.org/9780521517294

This page intentionally left blank

Logical Foundations of Proof Complexity

This book treats bounded arithmetic and propositional proof complexity from the point of view of computational complexity. The first seven chapters include the necessary logical background for the material and are suitable for a graduate course.

Associated with each of many complexity classes are both a two-sorted predicate calculus theory, with induction restricted to concepts in the class, and a propositional proof system. The complexity classes range from AC^0 for the weakest theory up to the polynomial hierarchy. Each bounded theorem in a theory translates into a family of (quantified) propositional tautologies with polynomial size proofs in the corresponding proof system. The theory proves the soundness of the associated proof system.

The result is a uniform treatment of many systems in the literature, including Buss's theories for the polynomial hierarchy and many disparate systems for complexity classes such as AC^0, $AC^0(m)$, TC^0, NC^1, L, NL, NC, and P.

Stephen Cook is a professor at the University of Toronto. He is author of many research papers, including his famous 1971 paper "The Complexity of Theorem Proving Procedures," and the 1982 recipient of the Turing Award. He was awarded a Steacie Fellowship in 1977 and a Killam Research Fellowship in 1982 and received the CRM/Fields Institute Prize in 1999. He is a Fellow of the Royal Society of London and the Royal Society of Canada and was elected to membership in the National Academy of Sciences (United States) and the American Academy of Arts and Sciences.

Phuong Nguyen is a postdoctoral researcher at McGill University. He received his MSc and PhD degrees from University of Toronto in 2004 and 2008, respectively. He has been awarded postdoctoral fellowships by the Eduard Čech Center for Algebra and Geometry (the Czech Republic) and by the Natural Sciences and Engineering Research Council of Canada (NSERC).

PERSPECTIVES IN LOGIC

The *Perspectives in Logic* series publishes substantial, high-quality books whose central theme lies in any area or aspect of logic. Books that present new material not now available in book form are particularly welcome. The series ranges from introductory texts suitable for beginning graduate courses to specialized monographs at the frontiers of research. Each book offers an illuminating perspective for its intended audience.

The series has its origins in the old *Perspectives in Mathematical Logic* series edited by the Ω-Group for "Mathematische Logik" of the Heidelberger Akademie der Wissenchaften, whose beginnings date back to the 1960s. The Association for Symbolic Logic has assumed editorial responsibility for the series and changed its name to reflect its interest in books that span the full range of disciplines in which logic plays an important role.

For more information, see http://www.aslonline.org/books_perspectives.html

PERSPECTIVES IN LOGIC

Logical Foundations of Proof Complexity

STEPHEN COOK
University of Toronto

PHUONG NGUYEN
McGill University

ASSOCIATION FOR SYMBOLIC LOGIC


CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore,
São Paulo, Delhi, Dubai, Tokyo

Cambridge University Press
The Edinburgh Building, Cambridge CB2 8RU, UK

Published in the United States of America by Cambridge University Press, New York

www.cambridge.org
Information on this title: www.cambridge.org/9780521517294

First published in print format 2010

ISBN-13 978-0-511-68614-6 eBook (Adobe Reader)

ISBN-13 978-0-521-51729-4 Hardback

CONTENTS

PREFACE

"Proof complexity" as used here has two related aspects: (i) the complexity of proofs of propositional formulas, and (ii) the study of weak (i.e., "bounded") theories of arithmetic. Aspect (i) goes back at least to Tseitin [109], who proved an exponential lower bound on the lengths of proofs in the weak system known as regular resolution. Later Cook and Reckhow [46] introduced a general definition of propositional proof system and related it to mainstream complexity theory by pointing out that such a system exists in which all tautologies have polynomial length proofs iff the two complexity classes ***NP*** and *co-**NP*** coincide.

Aspect (ii) goes back to Parikh [88], who introduced the theory known as $\boldsymbol{I}\Delta_0$, which is Peano Arithmetic with induction restricted to bounded formulas. Paris and Wilkie advanced the study of $\boldsymbol{I}\Delta_0$ and extensions in a series of papers (including [90, 89]) which relate them to complexity theory. Buss's seminal book [20] introduced the much-studied interleaved hierarchies $\boldsymbol{S}_2^i$ and $\boldsymbol{T}_2^i$ of theories related to the complexity classes Σ_i^p making up the polynomial hierarchy. Clote and Takeuti [38] and others introduced a host of theories related to other complexity classes.

The notion of propositional translation, which relates aspects (i) and (ii), goes back to [39], which introduced the equational theory ***PV*** for polynomial time functions and showed how theorems of ***PV*** can be translated into families of tautologies which have polynomial length proofs in the extended Frege proof system. Later (and independently) Paris and Wilkie [90] gave an elegant translation of bounded theorems in the relativized theory $\boldsymbol{I}\Delta_0(R)$ to polynomial length families of proofs in the weak propositional system bounded-depth Frege. Krajíček and Pudlák [73] introduced a hierarchy of proof systems $\langle \boldsymbol{G}_i \rangle$ for the quantified propositional calculus and showed how bounded theorems in Buss's theory $\boldsymbol{T}_2^i$ translate into polynomial length proofs in $\boldsymbol{G}_i$.

The aim of the present book is, first of all, to provide a sufficient background in logic for students in computer science and mathematics to understand our treatment of bounded arithmetic, and then to give an original treatment of the subject which emphasizes the three-way relationship among complexity classes, weak theories, and propositional proof systems.

Our treatment is unusual in that after Chapters 2 and 3 (which present Gentzen's sequent calculus $\boldsymbol{LK}$ and the bounded theory $\boldsymbol{I}\Delta_0$) we present our theories using the two-sorted vocabulary of Zambella [112]: one sort for natural numbers and the other for binary strings (i.e., finite sets of natural numbers). Our point of view is that the objects of interest are the binary strings: they are the natural inputs to the computing devices (Turing machines and Boolean circuits) studied by complexity theorists. The numbers are there as auxiliary variables, for example, to index the bits in the strings and measure their length. One reason for using this vocabulary is that the weakest complexity classes (such as $\boldsymbol{AC}^0$) that we study do not contain integer multiplication as a function, and since standard theories of arithmetic include multiplication as a primitive function, it is awkward to turn them into theories for these weak classes. In fact, our theories are simpler than many of the usual single-sorted theories in bounded arithmetic, because there is only one primitive function $|X|$ (the length of X) for strings X, while the axioms for the number sort are just those for $\boldsymbol{I}\Delta_0$.

Another advantage of using the two-sorted systems is that our propositional translations are especially simple: they are based on the Paris-Wilkie method [90]. The propositional atoms in the translation of a bounded formula $\varphi(X)$ with a free string variable X simply represent the bits of X.

Chapter 5 introduces our base theory $\boldsymbol{V}^0$, which corresponds to the smallest complexity class $\boldsymbol{AC}^0$ which we consider. All two-sorted theories we consider are extensions of $\boldsymbol{V}^0$. Chapter 6 studies $\boldsymbol{V}^1$, which is a two-sorted version of Buss's theory $\boldsymbol{S}^1_2$ and is related to the complexity class $\boldsymbol{P}$ (polynomial time). Chapter 7 introduces propositional translations for some theories. These translate bounded predicate formulas to families of quantified Boolean formulas. Chapter 8 introduces "minimal" theories for polynomial time by a method which is used extensively in Chapter 9. Chapter 8 also presents standard results concerning Buss's theories $\boldsymbol{S}^i_2$ and $\boldsymbol{T}^i_2$, but in the form of the two-sorted versions $\boldsymbol{V}^i$ and $\boldsymbol{TV}^i$ of these theories. Chapter 9 is based on the second author's PhD thesis, and uses an original uniform method to introduce minimal theories for many complexity classes between $\boldsymbol{AC}^0$ and $\boldsymbol{P}$. Some of these are related to single-sorted theories in the literature. Chapter 10 gives more examples of propositional translations and gives evidence for the thesis that each theory has a corresponding propositional proof system which serves as a kind of nonuniform version of the theory.

One purpose of this book is to serve as a basis for a program we call "Bounded Reverse Mathematics". This is inspired by the Friedman/Simpson program Reverse Mathematics [101], where now "Bounded" refers to bounded arithmetic. The goal is to find the weakest theory capable of proving a given theorem. The theorems in question are those of interest in computer science, and in general these can be proved in weak

theories. From the complexity theory point of view, the idea is to find the smallest complexity class such that the theorem can be proved using concepts in that class. This activity not only sheds light on the role of complexity classes in proofs, it can also lead to simplified proofs. A good example is Razborov's [96] greatly simplified proof of Hastad's Switching Lemma, which grew out of his attempt to formalize the lemma using only polynomial time concepts. His new proof led to important new results in propositional proof complexity. Throughout the book we give examples of theorems provable in the theories we describe.

The first seven chapters of this book grew out of notes for a graduate course taught several times beginning in 1998 at the University of Toronto by the first author. The prerequisites for the course and the book are some knowledge of both mathematical logic and complexity theory. However, Chapters 2 and 3 give a complete treatment of the necessary logic, and the Appendix together with material scattered throughout should provide sufficient background in complexity theory. There are exercises sprinkled throughout the text, which are intended both to supplement the material presented and to help the reader master the material. The more difficult exercises are marked with an asterisk.

Two sources have been invaluable to the authors in writing this book. The first is Krajíček's monograph [72], which is an essential possession for anyone working in this field. The second source is Buss's chapters [27, 28] in the *Handbook of Proof Theory*. His chapter I provides an excellent introduction to the proof theory of $\boldsymbol{LK}$, and his chapter II provides a thorough introduction to the first-order theories of bounded arithmetic. And of course Buss's monograph [20] *Bounded Arithmetic* was the origin of much of the material in our book.

We are grateful to Sam Buss and Jan Krajíček not only for their books but also for their considerable encouragement and help during the lengthy process of writing our book.

This book includes valuable input from several students of the first author as well as material from their PhD theses. The students include (besides the second author) Antonina Kolokolova, Tsuyoshi Morioka, Steven Perron, and Michael Soltys.

We are indebted to many others who have provided us with feedback on earlier versions of the book. These include Noriko Arai, Toshi Arai, Anton Belov, Mark Braverman, Timothy Chow, Lila Fontes, Kaveh Ghasemloo, Remo Goetschi, Daniel Ivan, Emil Jeřábek, Akitoshi Kawamura, Markus Latte, Dai Tri Man Le, Leonid Libkin, Dieter van Melkebeek, Toni Pitassi, Francois Pitt, Pavel Pudlák, Alan Skelley, Robert Solovay, Neil Thapen, Alasdair Urquhart, and Daniel Weller.

Stephen Cook
Phuong Nguyen

Chapter I

INTRODUCTION

This book studies logical systems which use restricted reasoning based on concepts from computational complexity. The complexity classes of interest lie mainly between the basic class $\boldsymbol{AC}^0$ (whose members are computed by polynomial-size families of bounded-depth circuits), and the polynomial hierarchy $\boldsymbol{PH}$, and include the sequence

$$\boldsymbol{AC}^0 \subset \boldsymbol{AC}^0(m) \subseteq \boldsymbol{TC}^0 \subseteq \boldsymbol{NC}^1 \subseteq \boldsymbol{L} \subseteq \boldsymbol{NL} \subseteq \boldsymbol{P} \subseteq \boldsymbol{PH} \tag{1}$$

where $\boldsymbol{P}$ is polynomial time. (See the Appendix for definitions.)

We associate with each of these classes a logical theory and a proof system for the (quantified) propositional calculus. The proof system can be considered a nonuniform version of the universal (or sometimes the bounded) fragment of the theory. The functions definable in the logical theory are those associated with the complexity class, and (in some cases) the lines in a polynomial size proof in the propositional system express concepts in the complexity class. Universal (or bounded) theorems of the logical theory translate into families of valid formulas with polynomial size proofs in the corresponding proof system. The logical theory proves the soundness of the proof system.

Conceptually the theory $\boldsymbol{VC}$ associated with a complexity class $\mathbf{C}$ can prove a given mathematical theorem if the induction hypotheses needed in the proof can be formulated using concepts from $\mathbf{C}$. We are interested in trying to find the weakest class $\mathbf{C}$ needed to prove various theorems of interest in computer science.

Here are some examples of the three-way association among complexity classes, theories, and proof systems:

class	$\boldsymbol{AC}^0$	$\boldsymbol{TC}^0$	$\boldsymbol{NC}^1$	$\boldsymbol{P}$	$\boldsymbol{PH}$	
theory	$\boldsymbol{V}^0$	$\boldsymbol{VTC}^0$	$\boldsymbol{VNC}^1$	$\boldsymbol{VP}$	$\boldsymbol{V}^\infty$	(2)
system	$\boldsymbol{AC}^0$-$\boldsymbol{Frege}$	$\boldsymbol{TC}^0$-$\boldsymbol{Frege}$	$\boldsymbol{Frege}$	$\boldsymbol{eFrege}$	$\langle \boldsymbol{G}_i \rangle$.	

Consider for example the class $\boldsymbol{NC}^1$. The uniform version is $\boldsymbol{ALogTime}$, the class of problems solvable by an alternating Turing machine in time $\mathcal{O}(\log n)$. The definable functions in the associated theory $\boldsymbol{VNC}^1$ are the $\boldsymbol{NC}^1$ functions, i.e., those functions whose bit graphs are $\boldsymbol{NC}^1$ relations.

A problem in nonuniform $\boldsymbol{NC}^1$ is defined by a polynomial-size family of log-depth Boolean circuits, or equivalently a polynomial-size family of propositional formulas. The corresponding propositional proof systems are called ***Frege*** systems, and are described in standard logic textbooks: a ***Frege*** proof of a tautology A consists of a sequence of propositional formulas ending in A, where each formula is either an axiom or follows from earlier formulas by a rule of inference. Universal theorems of $\boldsymbol{VNC}^1$ translate into polynomial-size families of ***Frege*** proofs. Finally $\boldsymbol{VNC}^1$ proves the soundness of ***Frege*** systems, and any proof system whose soundness is provable in $\boldsymbol{VNC}^1$ can be p-simulated by a ***Frege*** system (Theorem X.3.11).

The famous open question in complexity theory is whether the conjecture that $\boldsymbol{P}$ is a proper subset of $\boldsymbol{NP}$ is in fact true (we know $\boldsymbol{P} \subseteq \boldsymbol{NP} \subseteq \boldsymbol{PH}$). If $\boldsymbol{P} = \boldsymbol{NP}$ then the polynomial hierarchy $\boldsymbol{PH}$ collapses to $\boldsymbol{P}$, but it is possible that $\boldsymbol{PH}$ collapses only to $\boldsymbol{NP}$ and still $\boldsymbol{P} \neq \boldsymbol{NP}$. What may be less well known is that not only is it possible that $\boldsymbol{PH} = \boldsymbol{P}$, but it is consistent with our present knowledge that $\boldsymbol{PH} = \boldsymbol{AC}^0(6)$, so that all classes in (1) might be equal except for $\boldsymbol{AC}^0$ and $\boldsymbol{AC}^0(p)$ for p prime. This is one motivation for studying the theories associated with these complexity classes, since it ought to be easier to separate the theories corresponding to the complexity classes than to separate the classes themselves (but so far the theories in (2) have not been separated, except for $\boldsymbol{V}^0$).

A common example used to illustrate the complexity of the concepts needed to prove a theorem is the Pigeonhole Principle (PHP). Our version states that if $n + 1$ pigeons are placed in n holes, then some hole has two or more pigeons. We can present an instance of the PHP using a Boolean array $\langle P(i, j)\rangle$ $(0 \leq i \leq n, 0 \leq j < n)$, where $P(i, j)$ asserts that pigeon i is placed in hole j. Then the PHP can be formulated in the theory $\boldsymbol{V}^0$ by the formula

$$\forall i \leq n\, \exists j < n\, P(i, j) \supset \exists i_1, i_2 \leq n\, \exists j < n\, (i_1 \neq i_2 \wedge P(i_1, j) \wedge P(i_2, j)). \tag{3}$$

Ajtai [5] proved (in effect) that this formula is not a theorem of $\boldsymbol{V}^0$, and also that the propositional version (which uses atoms p_{ij} to represent $P(i, j)$ and finite conjunctions and disjunctions to express the bounded universal and existential number quantifiers) does not have polynomial size $\boldsymbol{AC}^0$-***Frege*** proofs. The intuitive reason for this is that a counting argument seems to be required to prove the PHP, but the complexity class $\boldsymbol{AC}^0$ cannot count the number of ones in a string of bits. On the other hand, the class $\boldsymbol{NC}^1$ can count, and indeed Buss proved that the propositional PHP does have polynomial size ***Frege*** proofs, and his method shows that (3) is a theorem of the theory $\boldsymbol{VNC}^1$. (In fact it is a theorem of the apparently weaker theory $\boldsymbol{VTC}^0$.)

A second example comes from linear algebra. If A and B are $n \times n$ matrices over some field, then

$$AB = I \supset BA = I. \tag{4}$$

A standard proof of this uses Gaussian elimination, which is a polynomial-time process. Indeed Soltys showed that (4) is a theorem of the theory $\boldsymbol{VP}$ corresponding to polynomial-time reasoning, and it follows that its propositional translation (say over the field of two elements) has polynomial-size proofs in the corresponding proof system ***eFrege***. It is an open question whether (4) over $\boldsymbol{GF(2)}$ (or any field) can be proved in $\boldsymbol{VNC}^1$, or whether the propositional version has polynomial-size ***Frege*** proofs.

The preceding example (4) is a universal theorem, in the sense that its statement has no existential quantifier. Another class of examples comes from existential theorems. From linear algebra, a natural example about $n \times n$ matrices is

$$\forall A \exists B \neq 0 (AB = I \vee AB = 0). \tag{5}$$

The complexity of finding B for a given A, even over $\boldsymbol{GF(2)}$, is thought not to be in $\boldsymbol{NC}^1$ (it is hard for log space). Assuming that this is the case, it follows that (5) is not a theorem of $\boldsymbol{VNC}^1$, since only $\boldsymbol{NC}^1$ functions are definable in that theory. This conclusion is the result of a general witnessing theorem, which states that if the formula $\forall x \exists y \varphi(x, y)$ (for suitable formulas φ) is provable in the theory associated with complexity class $\boldsymbol{C}$, then there is a Skolem function $f(x)$ whose complexity is in $\boldsymbol{C}$ and which satisfies $\forall x \varphi(x, f(x))$.

The theory $\boldsymbol{VNC}^1$ proves that (4) follows from (5). Both (4) and (5) are theorems of the theory $\boldsymbol{VP}$ associated with polynomial time.

Another example of an existential theorem is "Fermat's Little Theorem", which states that if n is a prime number and $1 \leq a < n$, then $a^{n-1} \equiv 1 \pmod{n}$. Its existential content is captured by its contrapositive form

$$(1 \leq a < n) \wedge (a^{n-1} \not\equiv 1 \pmod{n}) \supset \exists d (1 < d < n \wedge d | n). \tag{6}$$

It is not hard to see that the function $a^{n-1} \bmod n$ can be computed in time polynomial in the lengths of a and n, using repeated squaring. If (6) is provable in $\boldsymbol{VP}$, then by the witnessing theorem mentioned above it would follow that there is a polynomial time function $f(a, n)$ whose value $d = f(a, n)$ provides a proper divisor of n whenever a, n satisfy the hypothesis in (6). With the exception of the so-called Carmichael numbers, which can be factored in polynomial time, every composite n satisfies the hypothesis of (6) for at least half of the values of a, $1 \leq a < n$. Hence $f(a, n)$ would provide a probabilistic polynomial time algorithm for integer factoring. Such an algorithm is not known to exist, and would provide a method for breaking the RSA public-key encryption scheme.

Thus Fermat's Little Theorem is not provable in $\boldsymbol{VP}$, assuming that there is no probabilistic polynomial time factoring algorithm.

Propositional tautologies can be used to express universal theorems such as (3) (in which the Predicate P is implicitly universally quantified and the bounded number quantifiers can be expanded in translation) and (4), but are not well suited to express existential theorems such as (5) and (6). However the latter can be expressed using formulas in the quantified propositional calculus (QPC), which extends the propositional calculus by allowing quantifiers $\forall p$ and $\exists p$ over propositional variables p. Each of the complexity classes in (2) has an associated QPC system, and in fact the systems $\langle \boldsymbol{G}_i \rangle$ mentioned for $\boldsymbol{PH}$ form a hierarchy of QPC systems.

Most of the theories presented in this book, including those in (2), have the same "second-order" underlying vocabulary $\mathcal{L}_A^2$, introduced by Zambella. The vocabulary $\mathcal{L}_A^2$ is actually a vocabulary for the two-sorted first-order predicate calculus, where one sort is for numbers in $\mathbb{N}$ and the second sort is for finite sets of numbers. Here we regard an object of the second sort as a finite string over the alphabet $\{0, 1\}$ (the i-th bit in the string is 1 iff i is in the set). The strings are the objects of interest for the complexity classes, and serve as the main inputs for the machines or circuits that determine the class. The numbers serve a useful purpose as indices for the strings when describing properties of the strings. When they are used as machine or circuit inputs, they are presented in unary notation.

In the more common single-sorted theories such as Buss's hierarchies $\boldsymbol{S}_2^i$ and $\boldsymbol{T}_2^i$ the underlying objects are numbers which are presented in binary notation as inputs to Turing machines. Our two-sorted treatment has the advantage that the underlying vocabulary has no primitive operations on strings except the length function $|X|$ and the bit predicate $X(i)$ (meaning $i \in X$). This is especially important for studying weak complexity classes such as $\boldsymbol{AC}^0$. The standard vocabulary for single-sorted theories includes number multiplication, which is not an $\boldsymbol{AC}^0$ function on binary strings.

Chapter II provides a sufficient background in first-order logic for the rest of the book, including Gentzen's proof system $\boldsymbol{LK}$. An unusual feature is our treatment of anchored (or "free-cut-free") $\boldsymbol{LK}$-proofs. The completeness of these restricted systems is proved directly by a simple term-model construction as opposed to the usual syntactic cut-elimination method. The second form of the Herbrand Theorem proved here has many applications in later chapters for witnessing theorems.

Chapter III presents the necessary background on Peano Arithmetic (the first-order theory of $\mathbb{N}$ under $+$ and $\times$) and its subsystems, including the bounded theory $\boldsymbol{I}\Delta_0$. The functions definable in $\boldsymbol{I}\Delta_0$ are precisely those in the complexity class known as $\boldsymbol{LTH}$ (the Linear Time Hierarchy). An important theorem needed for this result is that the predicate $y = 2^x$ is definable in the vocabulary of arithmetic using a bounded formula

(Section III.3.3). The universal theory $\overline{I\Delta_0}$ has function symbols for each function in the Linear Time Hierarchy, and forms a conservative extension of $I\Delta_0$. This theory serves as a prototype for universal theories defined in later chapters for other complexity classes.

Chapter IV introduces the syntax and intended semantics for the two-sorted theories, which will be used throughout the remaining chapters. Here Σ_0^B is defined to be the class of formulas with no string quantifiers, and with all number quantifiers bounded. The Σ_1^B-formulas begin with zero or more bounded existential string quantifiers followed by a Σ_0^B-formula, and more generally Σ_i^B-formulas begin with at most i alternating blocks of bounded string quantifiers $\exists\forall\exists\ldots$. Representation theorems are proved which state that formulas in the syntactic class Σ_0^B represent precisely the (two-sorted) AC^0 relations, and for $i \geq 1$, formulas in Σ_i^B represent the relations in the i-th level of the polynomial hierarchy.

Chapter V introduces the hierarchy of two-sorted theories $V^0 \subset V^1 \subseteq V^2 \subseteq \cdots$. For $i \geq 1$, V^i is the two-sorted version of Buss's single-sorted theory S_2^i, which is associated with the ith level of the polynomial hierarchy. In this chapter we concentrate on V^0, which is associated with the complexity class AC^0. All two-sorted theories considered in later chapters are extensions of V^0. A Buss-style witnessing theorem is proved for V^0, showing that the existential string quantifiers in a Σ_1^B-theorem of V^0 can be witnessed by AC^0-functions. Since Σ_1^B-formulas have all string quantifiers in front, both the statement and the proof of the theorem are simpler than for the usual Buss-style witnessing theorems. (The same applies to the witnessing theorems proved in later chapters.) The final section proves that V^0 is finitely axiomatizable.

Chapter VI concentrates on the theory V^1, which is associated with the complexity class P. All (and only) polynomial time functions are Σ_1^B-definable in V^1. The positive direction is shown in two ways: by analyzing Turing machine computations and by using Cobham's characterization of these functions. The witnessing theorem for V^1 is shown using (two-sorted versions of) the anchored proofs described in Chapter II, and implies that only polynomial time functions are Σ_1^B-definable in V^1.

Chapter VII gives a general definition of propositional proof system. The goal is to associate a proof system with each theory so that each Σ_0^B-theorem of the theory translates into a polynomial size family of proofs in the proof system. Further, the theory should prove the soundness of the proof system, but this is not shown until Chapter X. In Chapter VII, translations are defined from V^0 to bounded-depth PK-proofs (i.e. bounded-depth Frege proofs), and also from V^1 to extended Frege proofs. Systems G_i and $G_i^\star$ for the quantified propositional calculus are defined, and for $i \geq 1$ we show how to translate bounded theorems of V^i

to polynomial size families of proofs in the system $\boldsymbol{G}_i^\star$. The two-sorted treatment makes these translations simple and natural.

Chapter VIII begins by introducing other two-sorted theories associated with polynomial time. The finitely axiomatized theory $\boldsymbol{VP}$ and its universal conservative extension $\boldsymbol{VPV}$ both appear to be weaker than $\boldsymbol{V}^1$, although they have the same Σ_1^B theorems as $\boldsymbol{V}^1$. $\boldsymbol{VP} = \boldsymbol{TV}^0$ is the base of the hierarchy of theories $\boldsymbol{TV}^0 \subseteq \boldsymbol{TV}^1 \subseteq \cdots$, where for $i \geq 1$, $\boldsymbol{TV}^i$ is isomorphic to Buss's single-sorted theory $\boldsymbol{T}_2^i$. The definable problems in $\boldsymbol{TV}^1$ have the complexity of Polynomial Local Search. A form of the Herbrand Theorem known as KPT Witnessing is proved and applied to show independence of the Replacement axiom scheme from some theories, and to relating the collapse of the $\boldsymbol{V}^\infty$ hierarchy with the provable collapse of the polynomial hierarchy. The Σ_j^B-definable search problems in $\boldsymbol{V}^i$ and $\boldsymbol{TV}^i$ are characterized for many i and j. The RSUV isomorphism theorem between $\boldsymbol{S}_2^i$ and $\boldsymbol{V}^i$ is proved.

See Table 3 on page 250 for a summary of which search problems are definable in $\boldsymbol{V}^i$ and $\boldsymbol{TV}^i$.

Chapter IX gives a uniform way of introducing minimal canonical theories for many complexity classes between $\boldsymbol{AC}^0$ and $\boldsymbol{P}$, including those mentioned earlier in (1). Each finitely axiomatized theory is defined as an extension of $\boldsymbol{V}^0$ obtained by adding a single axiom stating the existence of a computation solving a complete problem for the associated complexity class. Evidence for the "minimality" of each theory is presented by defining a universal theory whose axioms are simply a set of basic axioms for $\boldsymbol{V}^0$ together with the defining axioms for all the functions in the associated complexity class. These functions are defined as the function $\boldsymbol{AC}^0$-closure of the complexity class, or (as is the case for $\boldsymbol{P}$) using a recursion-theoretic characterization of the function class. The main theorem in each case is that the universal theory is a conservative extension of the finitely axiomatized theory.

Table 1 on page 7 gives a summary of the two-sorted theories presented in Chapter IX and elsewhere, and Table 2 on page 8 gives a list of some theorems provable (or possibly not provable) in the various theories.

Chapter X extends Chapter VII by presenting quantified propositional proof systems associated with various complexity classes, and defining translations from the bounded theorems of the theories introduced in Chapter IX to the appropriate proof system. Witnessing theorems for subsystems of $\boldsymbol{G}$ (quantified propositional calculus) are proved. The notion of *reflection principle* (soundness of a proof system) is defined, and many results showing which kinds of reflection principle for various systems can (or probably cannot) be proved in various theories. It is shown how reflection principles can be used to axiomatize some of the theories.

CLASS	THEORY	SEE
$\boldsymbol{AC}^0$	$\boldsymbol{V}^0$	Section V.1
	$\overline{\boldsymbol{V}}^0$	Section V.6
$\boldsymbol{AC}^0(2)$	$\boldsymbol{V}^0(2), \widehat{\boldsymbol{V}^0(2)}, \overline{\boldsymbol{V}^0(2)}$	Section IX.4.2
	$\boldsymbol{VAC}^0(2)\boldsymbol{V}$	Section IX.4.4
$\boldsymbol{AC}^0(m)$	$\boldsymbol{V}^0(m), \widehat{\boldsymbol{V}^0(m)}, \overline{\boldsymbol{V}^0(m)}$	Section IX.4.6
$\boldsymbol{AC}^0(6)$	$\boldsymbol{VAC}^0(6)\boldsymbol{V}$	Section IX.4.8
$\boldsymbol{ACC}$	$\boldsymbol{VACC}$	Section IX.4.6
$\boldsymbol{TC}^0$	$\boldsymbol{VTC}^0, \widehat{\boldsymbol{VTC}^0}, \overline{\boldsymbol{VTC}^0}$	Section IX.3.2
	$\boldsymbol{VTC}^0\boldsymbol{V}$	Section IX.3.4
$\boldsymbol{NC}^1$	$\boldsymbol{VNC}^1, \widehat{\boldsymbol{VNC}^1}, \overline{\boldsymbol{VNC}^1}$	Section IX.5.3
	$\boldsymbol{VNC}^1\boldsymbol{V}$	Section IX.5.5
$\boldsymbol{L}$	$\boldsymbol{VL}, \widehat{\boldsymbol{VL}}, \overline{\boldsymbol{VL}}$	Section IX.6.3
	$\boldsymbol{VLV}$	Section IX.6.4
$\boldsymbol{NL}$	$\boldsymbol{VNL}, \widehat{\boldsymbol{VNL}}, \overline{\boldsymbol{VNL}}$	Section IX.6.1
	$\boldsymbol{V}^1\textbf{-}\boldsymbol{KROM}$	Section IX.6.2
$\boldsymbol{AC}^k$ $(k \geq 1)$	$\boldsymbol{VAC}^k$	Section IX.5.6
$\boldsymbol{NC}^{k+1}$ $(k \geq 1)$	$\boldsymbol{VNC}^{k+1}$	Section IX.5.6
$\boldsymbol{NC}$	$\boldsymbol{VNC}$	Section IX.5.6
	$\boldsymbol{U}^1$	Section IX.5.6
$\boldsymbol{P}$	$\boldsymbol{VP}$	Section VIII.1
	$\boldsymbol{VPV}$	Section VIII.2
	$\boldsymbol{TV}^0$	Section VIII.3
	$\boldsymbol{V}^1\textbf{-}\boldsymbol{HORN}$	Section VIII.4
	$\boldsymbol{V}^1$	Chapter VI
$\boldsymbol{C}$ (for $\boldsymbol{C} \subseteq \boldsymbol{P}$)	$\boldsymbol{VC}, \widehat{\boldsymbol{VC}}, \overline{\boldsymbol{VC}}$	Section IX.2.1
$\boldsymbol{CC}(\boldsymbol{PLS})$	$\boldsymbol{TV}^1$	Section VIII.5
	$\boldsymbol{V}^2$	Section VIII.7.2

Table 1. Theories and their Σ_1^B-definable classes.

THEORY	(NON)THEOREM(?)	SEE
$\boldsymbol{V}^0$	(seq.) Jordan Curve Theorem	[84]
	$\nvdash \boldsymbol{PHP}$	Corollary VII.2.4
	$\nvdash$ onto $\boldsymbol{PHP}$, $\nvdash \boldsymbol{Count}_m$	Section IX.4.3
$\boldsymbol{V}^0(2)$	onto $\boldsymbol{PHP}$, $\boldsymbol{Count}_2$	Section IX.4.3
	(set) Jordan Curve Theorem	Section IX.4.5
	$\boldsymbol{PHP}$?, $\boldsymbol{Count}_3$?	Section IX.7.4
$\boldsymbol{V}^0(m)$	$\boldsymbol{Count}_{m'}$ (if $\gcd(m, m') > 1$)	Section IX.4.7
	$\boldsymbol{Count}_{m'}$? (if $\gcd(m, m') = 1$)	Section IX.7.4
	$\boldsymbol{PHP}$?	Section IX.7.4
$\boldsymbol{VTC}^0$	sorting	Exercise IX.3.9
	Reflection Principles for d-$\boldsymbol{PTK}$	Section X.4.2
	$\boldsymbol{PHP}$	Section IX.3.5
	Finite Szpilrajn's Theorem	Section IX.3.7
	Bondy's Theorem	Section IX.3.8
	define $\lfloor X/Y \rfloor$?	Section IX.7.3
$\boldsymbol{VNC}^1$	Reflection Principle for $\boldsymbol{PK}$	Theorem X.3.9
	Barrington's Theorem	Sec. IX.5.5 & [82]
	NUMONES	Section IX.5.4
$\boldsymbol{VL}$	Lind's characterization of $\boldsymbol{L}$	Section IX.6.4
	Reingold's Theorem?	Section IX.7.2
$\boldsymbol{VNL}$	Grädel's Theorem (for $\boldsymbol{NL}$)	Theorem IX.6.24
$\boldsymbol{VNC}^2$	Cayley–Hamilton Theorem?	Section IX.7.1
$\boldsymbol{VP} = \boldsymbol{TV}^0$	Reflection Principle for $\boldsymbol{ePK}$	Exercise X.2.22
	Grädel's Theorem (for $\boldsymbol{P}$)	Theorem VIII.4.8
	$\nvdash$ Fermat's Little Theorem (cond.)	page 3
$\boldsymbol{V}^1$	Prime Factorization Theorem	Exercise VI.4.4
$\boldsymbol{V}^i$ $(i \geq 1)$	Π^q_i-$RFN_{\boldsymbol{G}_{i-1}}$, Π^q_{i+2}-$RFN_{\boldsymbol{G}^\star_i}$	Theorem X.2.17
$\boldsymbol{TV}^i$ $(i \geq 0)$	Π^q_{i+2}-$RFN_{\boldsymbol{G}^\star_{i+1}}$, Π^q_{i+1}-$RFN_{\boldsymbol{G}_i}$	Theorem X.2.20

TABLE 2. Some theories and their (non)theorems/solvable problems (and open questions). ("cond." stands for conditional.) Many theorems of $\boldsymbol{VP}$, such as Kuratowski's Theorem, Hall's Theorem, Menger's Theorem are not discussed here.

Chapter II

THE PREDICATE CALCULUS AND THE SYSTEM *LK*

In this chapter we present the logical foundations for theories of bounded arithmetic. We introduce Gentzen's proof system ***LK*** for the predicate calculus, and prove that it is sound, and complete even when proofs have a restricted form called "anchored". We augment the system ***LK*** by adding equality axioms. We prove the Compactness Theorem for predicate calculus, and the Herbrand Theorem.

In general we distinguish between syntactic notions and semantic notions. Examples of syntactic notions are variables, connectives, formulas, and formal proofs. The semantic notions relate to meaning; for example truth assignments, structures, validity, and logical consequence.

The first section treats the simple case of propositional calculus.

II.1. Propositional Calculus

Propositional formulas (called simply *formulas* in this section) are built from the logical constants $\bot$, $\top$ (for False, True), propositional variables (or atoms) $P_1, P_2, \ldots$, connectives $\neg, \vee, \wedge$, and parentheses (,). We use $P, Q, R, \ldots$ to stand for propositional variables, $A, B, C, \ldots$ to stand for formulas, and $\Phi, \Psi, \ldots$ to stand for sets of formulas. When writing formulas such as $(P \vee (Q \wedge R))$, our convention is that $P, Q, R, \ldots$ stand for distinct variables.

Formulas are built according to the following rules:

- $\bot$, $\top$, P, are formulas (also called *atomic formulas*) for any variable P.
- If A and B are formulas, then so are $(A \vee B)$, $(A \wedge B)$, and $\neg A$.

The implication connective $\supset$ is not allowed in our formulas, but we will take $(A \supset B)$ to stand for $(\neg A \vee B)$. Also $(A \leftrightarrow B)$ stands for $((A \supset B) \wedge (B \supset A))$.

We sometimes abbreviate formulas by omitting parentheses, but the intended formula has all parentheses present as defined above.

A *truth assignment* is an assignment of truth values F, T to atoms. Given a truth assignment τ, the truth value A^τ of a formula A is defined

inductively as follows: $\bot^\tau = F$, $\top^\tau = T$, $P^\tau = \tau(P)$ for atom P, $(A \wedge B)^\tau = T$ iff both $A^\tau = T$ and $B^\tau = T$, $(A \vee B)^\tau = T$ iff either $A^\tau = T$ or $B^\tau = T$, $(\neg A)^\tau = T$ iff $A^\tau = F$.

DEFINITION II.1.1. A truth assignment τ *satisfies* A iff $A^\tau = T$; τ *satisfies* a set Φ of formulas iff τ satisfies A for all $A \in \Phi$. Φ is *satisfiable* iff some τ satisfies Φ; otherwise Φ is *unsatisfiable*. Similarly for A. $\Phi \models A$ (i.e., A is a *logical consequence* of Φ) iff τ satisfies A for every τ such that τ satisfies Φ. A formula A is *valid* iff $\models A$ (i.e., $A^\tau = T$ for all τ). A valid propositional formula is called a *tautology*. We say that A and B are *equivalent* (written $A \iff B$) iff $A \models B$ and $B \models A$.

Note that $\iff$ refers to semantic equivalence, as opposed to $=_{\text{syn}}$, which indicates syntactic equivalence. For example, $(P \vee Q) \iff (Q \vee P)$, but $(P \vee Q) \neq_{syn} (Q \vee P)$.

II.1.1. Gentzen's Propositional Proof System *PK*. We present the propositional part ***PK*** of Gentzen's sequent-based proof system ***LK***. Each line in a proof in the system ***PK*** is a *sequent* of the form

$$A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell \tag{7}$$

where $\longrightarrow$ is a new symbol and $A_1, \ldots, A_k$ and $B_1, \ldots, B_\ell$ are sequences of formulas ($k, \ell \geq 0$) called *cedents*. We call the cedent $A_1, \ldots, A_k$ the *antecedent* and $B_1, \ldots, B_\ell$ the *succedent* (or *consequent*).

The semantics of sequents is given as follows. We say that a truth assignment τ *satisfies* the sequent (7) iff either τ falsifies some A_i or τ satisfies some B_i. Thus the sequent is equivalent to the formula

$$\neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_k \vee B_1 \vee B_2 \vee \cdots \vee B_\ell. \tag{8}$$

(Here and elsewhere, a disjunction $C_1 \vee \cdots \vee C_n$ indicates parentheses have been inserted with association to the right. For example, $C_1 \vee C_2 \vee C_3 \vee C_4$ stands for $(C_1 \vee (C_2 \vee (C_3 \vee C_4)))$. Similarly for a disjunction $C_1 \wedge \cdots \wedge C_n$.) In other words, the conjunction of the A's implies the disjunction of the B's. In the cases in which the antecedent or succedent is empty, we see that the sequent $\longrightarrow A$ is equivalent to the formula A, and $A \longrightarrow$ is equivalent to $\neg A$, and just $\longrightarrow$ (with both antecedent and succedent empty) is false (unsatisfiable). We say that a sequent is *valid* if it is true under all truth assignments (which is the same as saying that its corresponding formula is a tautology).

DEFINITION II.1.2. A ***PK*** *proof* of a sequent S is a finite tree whose nodes are (labeled with) sequents, whose root (called the *endsequent*) is S and is written at the bottom, whose leaves (or *initial sequents*) are logical axioms (see below), such that each non-leaf sequent follows from the sequent(s) immediately above by one of the rules of inference given below.

The *logical axioms* are of the form

$$A \longrightarrow A, \qquad \bot \longrightarrow, \qquad \longrightarrow \top$$

where A is any formula. (Note that we differ here from most other treatments, which require that A be an atomic formula.) The rules of inference are as follows (here Γ and Δ denote finite sequences of formulas).

weakening rules

$$\text{left: } \frac{\Gamma \longrightarrow \Delta}{A, \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, A}$$

exchange rules

$$\text{left: } \frac{\Gamma_1, A, B, \Gamma_2 \longrightarrow \Delta}{\Gamma_1, B, A, \Gamma_2 \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta_1, A, B, \Delta_2}{\Gamma \longrightarrow \Delta_1, B, A, \Delta_2}$$

contraction rules

$$\text{left: } \frac{\Gamma, A, A \longrightarrow \Delta}{\Gamma, A \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, A, A}{\Gamma \longrightarrow \Delta, A}$$

$\neg$ introduction rules

$$\text{left: } \frac{\Gamma \longrightarrow \Delta, A}{\neg A, \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{A, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, \neg A}$$

$\wedge$ introduction rules

$$\text{left: } \frac{A, B, \Gamma \longrightarrow \Delta}{(A \wedge B), \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, A \qquad \Gamma \longrightarrow \Delta, B}{\Gamma \longrightarrow \Delta, (A \wedge B)}$$

$\vee$ introduction rules

$$\text{left: } \frac{A, \Gamma \longrightarrow \Delta \qquad B, \Gamma \longrightarrow \Delta}{(A \vee B), \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, A, B}{\Gamma \longrightarrow \Delta, (A \vee B)}$$

cut rule

$$\frac{\Gamma \longrightarrow \Delta, A \qquad A, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta}$$

The formula A in the cut rule is called the *cut* formula. A proof that does not use the cut rule is called *cut-free*. The new formulas in the bottom sequents of the introduction rules are called *principal formulas* and the formula(s) in the top sequent(s) that are used to form the principal formulas are called *auxiliary formulas*.

Note that there is one left introduction rule and one right introduction rule for each of the three logical connectives $\wedge, \vee, \neg$. Further, these rules seem to be the simplest possible, given the fact that in each case the bottom sequent is valid iff all top sequents are valid.

Note that repeated use of the exchange rules allows us to execute an arbitrary reordering of the formulas in the antecedent or succedent of a sequent. In presenting a proof in the system ***PK***, we will usually omit

mention of the steps requiring the exchange rules, but of course they are there implicitly.

Definition II.1.3. A ***PK*** proof of a formula A is a ***PK*** proof of the sequent $\longrightarrow A$.

As an example, we give a ***PK*** proof of one of De Morgan's laws:

$$\neg(P \wedge Q) \longrightarrow \neg P \vee \neg Q.$$

To find this (or any) proof, it is a good idea to start with the conclusion at the bottom, and work up by removing the connectives one at a time, outermost first, by using the introduction rules in reverse. This can be continued until some formula A occurs on both the left and right side of a sequent, or $\top$ occurs on the right, or $\bot$ occurs on the left. Then this sequent can be derived from one of the axioms $A \longrightarrow A$ or $\longrightarrow \top$ or $\bot \longrightarrow$ using weakenings and exchanges. The cut and contraction rules are not necessary, and weakenings are only needed immediately below axioms. (The cut rule can be used to shorten proofs, and contraction will be needed later for the predicate calculus.)

$$\dfrac{\dfrac{\dfrac{P \longrightarrow P}{P \longrightarrow P, \neg Q}\,(\text{weakening})}{\longrightarrow P, \neg P, \neg Q}\,(\neg \text{ right})\qquad \dfrac{\dfrac{Q \longrightarrow Q}{Q \longrightarrow Q, \neg P}\,(\text{weakening})}{\longrightarrow Q, \neg P, \neg Q}\,(\neg \text{ right})}{\longrightarrow P \wedge Q, \neg P, \neg Q}\,(\wedge \text{ right})$$

$$\dfrac{\dfrac{\longrightarrow P \wedge Q, \neg P, \neg Q}{\longrightarrow P \wedge Q, \neg P \vee \neg Q}\,(\vee \text{ right})}{\neg(P \wedge Q) \longrightarrow \neg P \vee \neg Q}\,(\neg \text{ left})$$

Exercise II.1.4. Give ***PK*** proofs for each of the following valid sequents:

(a) $\neg P \vee \neg Q \longrightarrow \neg(P \wedge Q)$.
(b) $\neg(P \vee Q) \longrightarrow \neg P \wedge \neg Q$.
(c) $\neg P \wedge \neg Q \longrightarrow \neg(P \vee Q)$.

Exercise II.1.5. Show that the contraction rules can be derived from the cut rule (with weakenings and exchanges).

Exercise II.1.6. Suppose that we allowed $\supset$ as a primitive connective, rather than one introduced by definition. Give the appropriate left and right introduction rules for $\supset$.

II.1.2. Soundness and Completeness of *PK*. Now we prove that ***PK*** is both sound and complete. That is, a propositional sequent is provable in ***PK*** iff it is valid.

Theorem II.1.7 (Soundness). *Every sequent provable in **PK** is valid.*

Proof. We show that the endsequent in every ***PK*** proof is valid, by induction on the number of sequents in the proof. For the base case, the proof is a single line: a logical axiom. Each logical axiom is obviously valid. For the induction step, one needs only verify for each rule that the bottom sequent is a logical consequence of the top sequent(s). □

Theorem II.1.8 (Completeness). *Every valid propositional sequent is provable in **PK** without using cut or contraction.*

Proof. The idea is discussed in the example proof above of De Morgan's laws. We need to use the inversion principle.

Lemma II.1.9 (Inversion Principle). *For each **PK** rule except for weakenings, if the bottom sequent is valid, then all top sequents are valid.*

This principle is easily verified by inspecting each of the eleven rules in question.

Now for the completeness theorem: We show that every valid sequent $\Gamma \longrightarrow \Delta$ has a ***PK*** proof, by induction on the total number of logical connectives $\wedge, \vee, \neg$ occurring in $\Gamma \longrightarrow \Delta$. For the base case, every formula in Γ and Δ is an atom or one of the constants $\bot, \top$, and since the sequent is valid, some atom P must occur in both Γ and Δ, or $\bot$ occurs in Γ or $\top$ occurs in Δ. Hence $\Gamma \longrightarrow \Delta$ can be derived from one of the logical axioms by weakenings and exchanges.

For the induction step, let A be any formula which is not an atom and not a constant in Γ or Δ. Then by the definition of propositional formula A must have one of the forms $(B \wedge C)$, $(B \vee C)$, or $\neg B$. Thus $\Gamma \longrightarrow \Delta$ can be derived from $\wedge$ introduction, $\vee$ introduction, or $\neg$ introduction, respectively, using either the left case or the right case, depending on whether A is in Γ or Δ, and also using exchanges, but no weakenings. In each case, each top sequent of the rule will have at least one fewer connective than $\Gamma \longrightarrow \Delta$, and the sequent is valid by the inversion principle. Hence each top sequent has a ***PK*** proof, by the induction hypothesis. □

The soundness and completeness theorems relate the semantic notion of validity to the syntactic notion of proof.

II.1.3. *PK* Proofs from Assumptions. We generalize the (semantic) definition of logical consequence from formulas to sequents in the obvious way: A sequent S is a *logical consequence* of a set Φ of sequents iff every truth assignment τ that satisfies Φ also satisfies S. We generalize the (syntactic) definition of a ***PK*** proof of a sequent S to a ***PK*** proof of S *from a set* Φ *of sequents* (also called a ***PK***-Φ proof) by allowing sequents in Φ to be leaves (called *nonlogical axioms*) in the proof tree, in addition to the logical axioms. It turns out that soundness and completeness generalize to this setting.

Theorem II.1.10 (Derivational Soundness and Completeness). *A propositional sequent S is a logical consequence of a set Φ of sequents iff S has a **PK**-Φ proof.*

Derivational soundness is proved in the same way as simple soundness: by induction on the number of sequents in the ***PK***-Φ proof, using the fact that the bottom sequent of each rule is a logical consequence of the top sequent(s).

A remarkable aspect of derivational completeness is that a finite proof exists even in case Φ is an infinite set. This is because of the compactness theorem (below) which implies that if S is a logical consequence of Φ, then S is a logical consequence of some finite subset of Φ.

In general, to prove S from Φ the cut rule is required. For example, there is no ***PK*** proof of $\longrightarrow P$ from $\longrightarrow P \wedge Q$ without using the cut rule. This follows from the *subformula property*, which states that in a cut-free proof π of a sequent S, every formula in every sequent of π is a subformula of some formula in S. This is stated more generally in the Proposition II.1.15.

Exercise II.1.11. Let A_S be the formula giving the meaning of a sequent S, as in (8). Show that there is a cut-free ***PK*** derivation of $\longrightarrow A_S$ from S.

Proof of Theorem II.1.10 (Completeness). From the above easy exercise and from the earlier Completeness Theorem and from Theorem II.1.16, Form 2 (compactness), we obtain an easy proof of derivational completeness. Suppose that the sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of sequents $S_1, \ldots, S_k$. Then by the above exercise we can derive each of the sequents $\longrightarrow A_{S_1}, \ldots, \longrightarrow A_{S_k}$ from the sequents $S_1, \ldots, S_k$. Also the sequent

$$A_{S_1}, \ldots, A_{S_k}, \Gamma \longrightarrow \Delta \tag{9}$$

is valid, and hence has a ***PK*** proof by Theorem II.1.8. Finally from (9) using successive cuts with cut formulas $A_{S_1}, \ldots, A_{S_k}$ we obtain the desired ***PK*** derivation of $\Gamma \longrightarrow \Delta$ from the the sequents $S_1, \ldots, S_k$. □

We now wish to show that the cut formulas in the derivation can be restricted to formulas occurring in the hypothesis sequents.

Definition II.1.12 (Anchored Proof). An instance of the cut rule in a ***PK***-Φ proof π is *anchored* if the cut formula A (also) occurs as a formula (rather than a subformula) in some nonlogical axiom of π. A ***PK***-Φ proof π is *anchored* if every instance of cut in π is anchored.

Our *anchored* proofs are similar to *free-cut-free* proofs in [72] and elsewhere. Our use of the term *anchored* is inspired by [27].

The derivational completeness theorem can be strengthened as follows.

Theorem II.1.13 (Anchored Completeness). *If a propositional sequent S is a logical consequence of a set Φ of sequents, then there is an anchored **PK**-Φ proof of S.*

We illustrate the proof of the anchored completeness theorem by proving the special case in which Φ consists of the single sequent $A \longrightarrow B$. Assume that the sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of $A \longrightarrow B$. Then both of the sequents $\Gamma \longrightarrow \Delta, A$ and $B, A, \Gamma \longrightarrow \Delta$ are valid (why?). Hence by Theorem II.1.8 they have ***PK*** proofs π_1 and π_2, respectively. We can use these proofs to get a proof of $\Gamma \longrightarrow \Delta$ from $A \longrightarrow B$ as shown below, where the double line indicates the rules weakening and exchange have been applied.

$$\dfrac{\dfrac{\Gamma \longrightarrow \Delta, A \quad (\pi_1)}{\Gamma \longrightarrow \Delta} \qquad \dfrac{\dfrac{A \longrightarrow B}{A, \Gamma \longrightarrow \Delta, B} \qquad B, A, \Gamma \longrightarrow \Delta \quad (\pi_2)}{A, \Gamma \longrightarrow \Delta}\,(\text{cut})}{\Gamma \longrightarrow \Delta}\,(\text{cut})$$

Next consider the case in which Φ has the form

$$\{\longrightarrow A_1, \longrightarrow A_2, \ldots, \longrightarrow A_k\}$$

for some set $\{A_1, \ldots, A_k\}$ of formulas. Assume that $\Gamma \longrightarrow \Delta$ is a logical consequence of Φ in this case. Then the sequent

$$A_1, A_2, \ldots, A_k, \Gamma \longrightarrow \Delta$$

is valid, and hence has a ***PK*** proof π. Now we can use the assumptions Φ and the cut rule to successively remove $A_1, A_2, \ldots, A_k$ from the above sequent to conclude $\Gamma \longrightarrow \Delta$. For example, A_1 is removed as follows (the double line represents applications of the rule weakening and exchange):

$$\dfrac{\dfrac{\longrightarrow A_1}{A_2, \ldots, A_k, \Gamma \longrightarrow \Delta, A_1} \qquad A_1, A_2, \ldots, A_k, \Gamma \longrightarrow \Delta \quad (\pi)}{A_2, \ldots, A_k, \Gamma \longrightarrow \Delta}\,(\text{cut})$$

Exercise II.1.14. Prove the anchored completeness theorem for the more general case in which Φ is any finite set of sequents. Use induction on the number of sequents in Φ.

A nice property of anchored proofs is the following.

Proposition II.1.15 (Subformula Property). *If π is an anchored **PK**-Φ proof of S, then every formula in every sequent of π is a subformula of a formula either in S or in some nonlogical axiom of π.*

Proof. This follows by induction on the number of sequents in π, using the fact that for every rule other than cut, every formula on the top is a subformula of some formula on the bottom. For the case of cut we use the fact that every cut formula is a formula in some nonlogical axiom of π. □

The Subformula Property can be generalized in a way that applies to cut-free ***LK*** proofs in the predicate calculus, and this will play an important role later in proving witnessing theorems.

II.1.4. Propositional Compactness. We conclude our treatment of the propositional calculus with a fundamental result which also plays an important role in the predicate calculus.

Theorem II.1.16 (Propositional Compactness). *We state three different forms of this result. All three are equivalent.*

Form 1: *If Φ is an unsatisfiable set of propositional formulas, then some finite subset of Φ is unsatisfiable.*

Form 2: *If a formula A is a logical consequence of a set Φ of formulas, then A is a logical consequence of some finite subset of Φ.*

Form 3: *If every finite subset of a set Φ of formulas is satisfiable, then Φ is satisfiable.*

Exercise II.1.17. Prove the equivalence of the three forms. (Note that Form 3 is the contrapositive of Form 1.)

Proof of Form 1. Let Φ be an unsatisfiable set of formulas. By our definition of propositional formula, all propositional variables in Φ come from a countable list $P_1, P_2, \ldots$. (See Exercise II.1.19 for the uncountable case.) Organize the set of truth assignments into an infinite rooted binary tree B. Each node except the root is labeled with a literal P_i or $\neg P_i$. The two children of the root are labeled P_1 and $\neg P_1$, indicating that P_1 is assigned T or F, respectively. The two children of each of these nodes are labeled P_2 and $\neg P_2$, respectively, indicating the truth value of P_2. Thus each infinite branch in the tree represents a complete truth assignment, and each path from the root to a node represents a truth assignment to the atoms $P_1, \ldots, P_i$, for some i.

Now for every node v in the tree B, prune the tree at v (i.e., remove the subtree rooted at v, keeping v itself) if the partial truth assignment τ_v represented by the path to v falsifies some formula A_v in Φ, where all atoms in A_v get values from τ_v. Let B' be the resulting pruned tree. Since Φ is unsatisfiable, every path from the root in B' must end after finitely many steps in some leaf v labeled with a formula A_v in Φ. It follows from König's Lemma below that B' is finite. Let Φ' be the finite subset of Φ consisting of all formulas A_v labeling the leaves of B'. Since every truth assignment τ determines a path in B' which ends in a leaf A_v falsified by τ, it follows that Φ' is unsatisfiable. □

Lemma II.1.18 (König's Lemma). *Suppose T is a rooted tree in which every node has only finitely many children. If every branch in T is finite, then T is finite.*

Proof. We prove the contrapositive: If T is infinite (but every node has only finitely many children) then T has an infinite branch. We can define

an infinite path in T as follows: Start at the root. Since T is infinite but the root has only finitely many children, the subtree rooted at one of these children must be infinite. Choose such a child as the second node in the branch, and continue. □

Exercise II.1.19. (*For those with some knowledge of set theory or point set topology*) The above proof of the propositional compactness theorem only works when the set of atoms is countable, but the result still holds even when Φ is an uncountable set with an uncountable set $\mathcal{A}$ of atoms. Complete each of the two proof outlines below.

(a) Prove Form 3 using Zorn's Lemma as follows: Call a set Ψ of formulas *finitely satisfiable* if every finite subset of Ψ is satisfiable. Assume that Φ is finitely satisfiable. Let $\mathcal{C}$ be the class of all finitely satisfiable sets $\Psi \supseteq \Phi$ of propositional formulas using atoms in Φ. Order these sets Ψ by inclusion. Show that the union of any chain of sets in $\mathcal{C}$ is again in the class $\mathcal{C}$. Hence by Zorn's Lemma, $\mathcal{C}$ has a maximal element Ψ_0. Show that Ψ_0 has a unique satisfying assignment, and hence Φ is satisfiable.

(b) Show that Form 1 follows from Tychonoff's Theorem: The product of compact topological spaces is compact. The set of all truth assignments to the atom set $\mathcal{A}$ can be given the product topology, when viewed as the product for all atoms P in $\mathcal{A}$ of the two-point space $\{T, F\}$ of assignments to P, with the discrete topology. By Tychonoff's Theorem, this space of assignments is compact. Show that for each formula A, the set of assignments falsifying A is open. Thus Form 1 follows from the definition of compact: every open cover has a finite subcover.

II.2. Predicate Calculus

In this section we present the syntax and semantics of the predicate calculus (also called first-order logic). We show how to generalize Gentzen's proof system ***PK*** for the propositional calculus to the system ***LK*** for the predicate calculus, by adding quantifier introduction rules. We show that ***LK*** is sound and complete. We prove an anchored completeness theorem which limits the need for the cut rule in the presence of nonlogical axioms.

II.2.1. Syntax of the Predicate Calculus. A *first-order vocabulary* (or just *vocabulary*, or *language*) $\mathcal{L}$ is specified by the following:

1) For each $n \geq 0$ a set of n-ary function symbols (possibly empty). We use $f, g, h, \ldots$ as meta-symbols for function symbols. A zero-ary function symbol is called a constant symbol.
2) For each $n \geq 0$, a set of n-ary predicate symbols (which must be nonempty for some n). We use $P, Q, R, \ldots$ as meta-symbols for predicate symbols. A zero-ary predicate symbol is the same as a propositional atom.

In addition, the following symbols are available to build first-order terms and formulas:

1) An infinite set of variables. We use $x, y, z, \ldots$ and sometimes $a, b, c, \ldots$ as meta-symbols for variables.
2) Connectives $\neg, \wedge, \vee$ (not, and, or); logical constants $\bot, \top$ (for False, True).
3) Quantifiers $\forall, \exists$ (for all, there exists).
4) (,) (parentheses).

Given a vocabulary $\mathcal{L}$, *$\mathcal{L}$-terms* are certain strings built from variables and function symbols of $\mathcal{L}$, and are intended to represent objects in the universe of discourse. We will drop mention of $\mathcal{L}$ when it is not important, or clear from context.

DEFINITION II.2.1 ($\mathcal{L}$-Terms). Let $\mathcal{L}$ be a first-order vocabulary.

1) Every variable is an $\mathcal{L}$-term.
2) If f is an n-ary function symbol of $\mathcal{L}$ and $t_1, \ldots, t_n$ are $\mathcal{L}$-terms, then $f t_1 \ldots t_n$ is an $\mathcal{L}$-term.

Recall that a 0-ary function symbol is called a constant symbol (or sometimes just a *constant*). Note that all constants in $\mathcal{L}$ are $\mathcal{L}$-terms.

DEFINITION II.2.2 ($\mathcal{L}$-Formulas). Let $\mathcal{L}$ be a first-order vocabulary. First-order formulas in $\mathcal{L}$ (or *$\mathcal{L}$-formulas*, or just *formulas*) are defined inductively as follows:

1) $Pt_1 \cdots t_n$ is an *atomic* $\mathcal{L}$-formula, where P is an n-ary predicate symbol in $\mathcal{L}$ and $t_1, \ldots, t_n$ are $\mathcal{L}$-terms. Also each of the logical constants $\bot, \top$ is an atomic formula.
2) If A and B are $\mathcal{L}$-formulas, so are $\neg A$, $(A \wedge B)$, and $(A \vee B)$.
3) If A is an $\mathcal{L}$-formula and x is a variable, then $\forall x A$ and $\exists x A$ are $\mathcal{L}$-formulas.

Examples of formulas: $(\neg\forall x Px \vee \exists x \neg Px)$, $(\forall x \neg Pxy \wedge \neg\forall z P f yz)$.

As in the case of propositional formulas, we use the notation $(A \supset B)$ for $(\neg A \vee B)$ and $(A \leftrightarrow B)$ for $((A \supset B) \wedge (B \supset A))$.

It can be shown that no proper initial segment of a term is a term, and hence every term can be parsed uniquely according to Definition II.2.1. A similar remark applies to formulas, and Definition II.2.2.

NOTATION. $r = s$ stands for $= rs$, and $r \neq s$ stands for $\neg(r = s)$.

DEFINITION II.2.3 (The Vocabulary of Arithmetic).

$$\mathcal{L}_A = [0, 1, +, \cdot\,; =, \leq].$$

Here 0, 1 are constants; $+$, $\cdot$ are binary function symbols; $=$, $\leq$ are binary predicate symbols. In practice we use infix notation for $+, \cdot, =, \leq$. Thus, for example, $(t_1 \cdot t_2) =_{syn} \cdot t_1 t_2$ and $(t_1 + t_2) =_{syn} + t_1 t_2$.

Definition II.2.4 (Free and Bound Variables). An occurrence of x in A is *bound* iff it is in a subformula of A of the form $\forall xB$ or $\exists xB$. Otherwise the occurrence is *free*.

Notice that a variable can have both free and bound occurrences in one formula. For example, in $Px \wedge \forall xQx$, the first occurrence of x is free, and the second occurrence is bound.

Definition II.2.5. A formula is *closed* if it contains no free occurrence of a variable. A term is *closed* if it contains no variable. A closed formula is called a *sentence*.

II.2.2. Semantics of Predicate Calculus.

Definition II.2.6 ($\mathcal{L}$-Structure). If $\mathcal{L}$ is a first-order vocabulary, then an $\mathcal{L}$-structure $\mathcal{M}$ consists of the following:

1) A nonempty set M called the *universe*. (Variables in an $\mathcal{L}$-formula are intended to range over M.)
2) For each n-ary function symbol f in $\mathcal{L}$, an associated function $f^{\mathcal{M}} : M^n \to M$.
3) For each n-ary predicate symbol P in $\mathcal{L}$, an associated relation $P^{\mathcal{M}} \subseteq M^n$. If $\mathcal{L}$ contains $=$, then $=^{\mathcal{M}}$ must be the true equality relation on M.

Notice that the predicate symbol $=$ gets special treatment in the above definition, in that $=^{\mathcal{M}}$ must always be the true equality relation. Any other predicate symbol may be interpreted by an arbitrary relation of the appropriate arity.

Every $\mathcal{L}$-sentence becomes either true or false when interpreted by an $\mathcal{L}$-structure $\mathcal{M}$, as explained below. If a sentence A becomes true under $\mathcal{M}$, then we say $\mathcal{M}$ *satisfies* A, or $\mathcal{M}$ is a *model* for A, and write $\mathcal{M} \models A$.

If A has free variables, then these variables must be interpreted as specific elements in the universe M before A gets a truth value under the structure $\mathcal{M}$. For this we need the following:

Definition II.2.7 (Object Assignment). An *object assignment* σ for a structure $\mathcal{M}$ is a mapping from variables to the universe M.

Below we give the formal definition of notion $\mathcal{M} \models A[\sigma]$, which is intended to mean that the structure $\mathcal{M}$ satisfies the formula A when the free variables of A are interpreted according to the object assignment σ. First it is necessary to define the notation $t^{\mathcal{M}}[\sigma]$, which is the element of universe M assigned to the term t by the structure $\mathcal{M}$ when the variables of t are interpreted according to σ.

Notation. If x is a variable and $m \in M$, then the object assignment $\sigma(m/x)$ is the same as σ except it maps x to m.

DEFINITION II.2.8 (Basic Semantic Definition). Let $\mathcal{L}$ be a first-order vocabulary, let $\mathcal{M}$ be an $\mathcal{L}$-structure, and let σ be an object assignment for $\mathcal{M}$. Each $\mathcal{L}$-term t is assigned an element $t^{\mathcal{M}}[\sigma]$ in M, defined by structural induction on terms t, as follows (refer to the definition of $\mathcal{L}$-term):

(a) $x^{\mathcal{M}}[\sigma]$ is $\sigma(x)$, for each variable x.
(b) $(ft_1 \cdots t_n)^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma])$.

For A an $\mathcal{L}$-formula, the notion $\mathcal{M} \models A[\sigma]$ ($\mathcal{M}$ *satisfies* A *under* σ) is defined by structural induction on formulas A as follows (refer to the definition of formula):

(a) $\mathcal{M} \models \top$ and $\mathcal{M} \not\models \bot$.
(b) $\mathcal{M} \models (Pt_1 \cdots t_n)[\sigma]$ iff $\langle t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma]\rangle \in P^{\mathcal{M}}$.
(c) If $\mathcal{L}$ contains $=$, then $\mathcal{M} \models (s = t)[\sigma]$ iff $s^{\mathcal{M}}[\sigma] = t^{\mathcal{M}}[\sigma]$.
(d) $\mathcal{M} \models \neg A[\sigma]$ iff $\mathcal{M} \not\models A[\sigma]$.
(e) $\mathcal{M} \models (A \vee B)[\sigma]$ iff $\mathcal{M} \models A[\sigma]$ or $\mathcal{M} \models B[\sigma]$.
(f) $\mathcal{M} \models (A \wedge B)[\sigma]$ iff $\mathcal{M} \models A[\sigma]$ and $\mathcal{M} \models B[\sigma]$.
(g) $\mathcal{M} \models (\forall xA)[\sigma]$ iff $\mathcal{M} \models A[\sigma(m/x)]$ for all $m \in M$.
(h) $\mathcal{M} \models (\exists xA)[\sigma]$ iff $\mathcal{M} \models A[\sigma(m/x)]$ for some $m \in M$.

Note that item (c) in the definition of $\mathcal{M} \models A[\sigma]$ follows from (b) and the fact that $=^{\mathcal{M}}$ is always the equality relation.

If t is a closed term (i.e., contains no variables), then $t^{\mathcal{M}}[\sigma]$ is independent of σ, and so we sometimes just write $t^{\mathcal{M}}$. Similarly, if A is a sentence, then we sometimes write $\mathcal{M} \models A$ instead of $\mathcal{M} \models A[\sigma]$, since σ does not matter.

DEFINITION II.2.9 (Standard Model). The *standard model* $\underline{\mathbb{N}}$ for the vocabulary $\mathcal{L}_A$ is a structure with universe $M = \mathbb{N} = \{0, 1, 2, \ldots\}$, where $0, 1, +, \cdot, =, \leq$ get their usual meanings on the natural numbers.

As an example, $\underline{\mathbb{N}} \models \forall x \forall y \exists z(x + z = y \vee y + z = x)$ (since either $y - x$ or $x - y$ exists) but $\underline{\mathbb{N}} \not\models \forall x \exists y(y + y = x)$ since not all natural numbers are even.

In the future we sometimes assume that there is some first-order vocabulary $\mathcal{L}$ in the background, and do not necessarily mention it explicitly.

NOTATION. In general, Φ denotes a set of formulas, $A, B, C, \ldots$ denote formulas, $\mathcal{M}$ denotes a structure, and σ denotes an object assignment.

DEFINITION II.2.10. (a) $\mathcal{M} \models \Phi[\sigma]$ iff $\mathcal{M} \models A[\sigma]$ for all $A \in \Phi$.
(b) $\mathcal{M} \models \Phi$ iff $\mathcal{M} \models \Phi[\sigma]$ for all σ.
(c) $\Phi \models A$ iff for all $\mathcal{M}$ and all σ, if $\mathcal{M} \models \Phi[\sigma]$ then $\mathcal{M} \models A[\sigma]$.
(d) $\models A$ (A is *valid*) iff $\mathcal{M} \models A[\sigma]$ for all $\mathcal{M}$ and σ.
(e) $A \Longleftrightarrow B$ (A and B are *logically equivalent*, or just *equivalent*) iff for all $\mathcal{M}$ and all σ, $\mathcal{M} \models A[\sigma]$ iff $\mathcal{M} \models B[\sigma]$.

$\Phi \models A$ is read "A is a logical consequence of Φ". Do not confuse this with our other use of the symbol $\models$, as in $\mathcal{M} \models A$ ($\mathcal{M}$ satisfies A). In the latter, $\mathcal{M}$ is a structure, rather than a set of formulas.

If Φ consists of a single formula B, then we write $B \models A$ instead of $\{B\} \models A$.

Definition II.2.11 (Substitution). Let s, t be terms, and A a formula. Then $t(s/x)$ is the result of replacing all occurrences of x in t by s, and $A(s/x)$ is the result of replacing all *free* occurrences of x in A by s.

Lemma II.2.12. *For each structure $\mathcal{M}$ and each object assignment σ,*

$$(s(t/x))^{\mathcal{M}}[\sigma] = s^{\mathcal{M}}[\sigma(m/x)]$$

where $m = t^{\mathcal{M}}[\sigma]$.

Proof. Structural induction on s. □

Definition II.2.13. A term t is *freely substitutable for x in A* iff no free occurrence of x in A is in a subformula of A of the form $\forall y B$ or $\exists y B$, where y occurs in t.

Theorem II.2.14 (Substitution). *If t is freely substitutable for x in A then for all structures $\mathcal{M}$ and all object assignments σ, $\mathcal{M} \models A(t/x)[\sigma]$ iff $\mathcal{M} \models A[\sigma(m/x)]$, where $m = t^{\mathcal{M}}[\sigma]$.*

Proof. Structural induction on A. □

Remark (Change of Bound Variable). If t is not freely substitutable for x in A, it is because some variable y in t gets "caught" by a quantifier, say $\exists y B$. Then replace $\exists y B$ in A by $\exists z B$, where z is a new variable. Then the meaning of A does not change (by the Formula Replacement Theorem below), but by repeatedly changing bound variables in this way t becomes freely substitutable for x in A.

Theorem II.2.15 (Formula Replacement). *If B and B' are equivalent and A' results from A by replacing some occurrence of B in A by B', then A and A' are equivalent.*

Proof. Structural induction on A relative to B. □

II.2.3. The First-Order Proof System *LK*. We now extend the propositional proof system ***PK*** to the first-order sequent proof system ***LK***. For this it is convenient to introduce two kinds of variables: *free variables* denoted by $a, b, c, \ldots$ and *bound variables* denoted by $x, y, z, \ldots$. A first-order sequent has the form

$$A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell$$

where now the A_i and B_j are first-order formulas satisfying the restriction that they have no free occurrences of the "bound" variables $x, y, z, \ldots$ and no bound occurrences of the "free" variables $a, b, c, \ldots$.

The sequent system ***LK*** is an extension of the propositional system ***PK***, where now all formulas are first-order formulas satisfying the restriction explained above.

In addition to the rules given for ***PK***, the system ***LK*** has four rules for introducing the quantifiers.

Important Remark. In the rules below, t is any term not involving any bound variables $x, y, z, \ldots$ and $A(t)$ is the result of substituting t for all free occurrences of x in $A(x)$. Similarly $A(b)$ is the result of substituting b for all free occurrences of x in $A(x)$. Note that t and b can always be freely substituted for x in $A(x)$ when $\forall x A(x)$ or $\exists x A(x)$ satisfy the free/bound variable restrictions described above.

$\forall$ introduction rules

$$\text{left: } \frac{A(t), \Gamma \longrightarrow \Delta}{\forall x A(x), \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, A(b)}{\Gamma \longrightarrow \Delta, \forall x A(x)}$$

$\exists$ introduction rules

$$\text{left: } \frac{A(b), \Gamma \longrightarrow \Delta}{\exists x A(x), \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, A(t)}{\Gamma \longrightarrow \Delta, \exists x A(x)}$$

Restriction. The free variable b is called an *eigenvariable* and must not occur in the conclusion in $\forall$-right or $\exists$-left. Also, as remarked above, the term t must not involve any bound variables $x, y, z, \ldots$.

The new formulas in the bottom sequents ($\exists x A(x)$ or $\forall x A(x)$) are called *principal formulas*, and the corresponding formulas in the top sequents ($A(b)$ or $A(t)$) are called *auxiliary formulas*.

Definition II.2.16 (Semantics of first-order sequents). The semantics of first-order sequents is a natural generalization of the semantics of propositional sequents. Again the sequent $A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell$ has the same meaning as its associated formula

$$\neg A_1 \vee \neg A_2 \vee \cdots \vee \neg A_k \vee B_1 \vee B_2 \vee \cdots \vee B_\ell.$$

In particular, we say that the sequent is *valid* iff its associated formula is valid.

Theorem II.2.17 (Soundness for ***LK***). *Every sequent provable in* ***LK*** *is valid.*

Proof. This is proved by induction on the number of sequents in the ***LK*** proof, as in the case of ***PK***. However, unlike the case of ***PK***, not all of the four new quantifier rules satisfy the condition that the bottom sequent is a logical consequence of the top sequent. In particular this may be false for $\forall$-right and for $\exists$-left. However it is easy to check that each rule satisfies the weaker condition that if the top sequent is valid, then the bottom sequent is valid, and this suffices for the proof. □

Exercise II.2.18. Give examples to show that the restriction given on the quantifier rules, that b must not occur in the conclusion in $\forall$-right and $\exists$-left, is necessary to ensure that these rules preserve validity.

*Example of an **LK** proof.* An ***LK*** proof of a valid first-order sequent can be obtained using the same method as in the propositional case: Write the goal sequent at the bottom, and move up by using the introduction rules in reverse. A good heuristic is: if there is a choice about which quantifier to remove next, choose $\forall$-right and $\exists$-left first (working backward), since these rules carry a restriction.

Here is an ***LK*** proof of the sequent $\forall xPx \vee \forall xQx \longrightarrow \forall x(Px \vee Qx)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{Pb \longrightarrow Pb}{Pb \longrightarrow Pb, Qb}\,(\text{weakening})}{\forall xPx \longrightarrow Pb, Qb}\,(\forall \text{ left}) \qquad \dfrac{\dfrac{\dfrac{Qb \longrightarrow Qb}{Qb \longrightarrow Pb, Qb}\,(\text{weakening})}{\forall xQx \longrightarrow Pb, Qb}\,(\forall \text{ left})}{\forall xPx \vee \forall xQx \longrightarrow Pb, Qb}\,(\vee \text{ left})}{\forall xPx \vee \forall xQx \longrightarrow Pb \vee Qb}\,(\vee \text{ right})}{\forall xPx \vee \forall xQx \longrightarrow \forall x(Px \vee Qx)}\,(\forall \text{ right})$$

Exercise II.2.19. Give ***LK*** proofs for the following valid sequents:

(a) $\forall xPx \wedge \forall xQx \longrightarrow \forall x(Px \wedge Qx)$.
(b) $\forall x(Px \wedge Qx) \longrightarrow \forall xPx \wedge \forall xQx$.
(c) $\exists x(Px \vee Qx) \longrightarrow \exists xPx \vee \exists xQx$.
(d) $\exists xPx \vee \exists xQx \longrightarrow \exists x(Px \vee Qx)$.
(e) $\exists x(Px \wedge Qx) \longrightarrow \exists xPx \wedge \exists xQx$.
(f) $\exists y\forall xPxy \longrightarrow \forall x\exists yPxy$.
(g) $\forall xPx \longrightarrow \exists xPx$.

Check that the rule restrictions seem to prevent generating ***LK*** proofs for the following invalid sequents:

(h) $\exists xPx \wedge \exists xQx \longrightarrow \exists x(Px \wedge Qx)$.
(i) $\forall x\exists yPxy \longrightarrow \exists y\forall xPxy$.

II.2.4. Free Variable Normal Form. In future chapters it will be useful to assume that ***LK*** proofs satisfy certain restrictions on free variables.

Definition II.2.20 (Free Variable Normal Form). Let π be an ***LK*** proof with endsequent S. A free variable in S is called a *parameter variable* of π. We say π is in *free variable normal form* if (1) no free variable is completely eliminated from any sequent in π by any rule except possibly $\forall$-right and $\exists$-left, and in these cases the eigenvariable which is eliminated is not a parameter variable, and (2) every nonparameter free variable appearing in π is used exactly once as an eigenvariable.

Thus if a proof is in free variable normal form, then any occurrence of a parameter variable persists until the endsequent, and any occurrence of a

nonparameter free variable persists until it is eliminated as an eigenvariable in $\forall$-right or $\exists$-left.

We now describe a simple procedure for transforming an ***LK*** proof π to a similar proof of the same endsequent in free variable normal form, assuming that the underlying vocabulary $\mathcal{L}$ has at least one constant symbol e. Note that the only rules other than $\forall$-right and $\exists$-left which can eliminate a free variable from a sequent are cut, $\exists$-right, and $\forall$-left. It is important that π have a tree structure in order for the procedure to work.

Transform π by repeatedly performing the following operation until the resulting proof is in free variable normal form. Select some upper-most rule in π which eliminates a free variable from a sequent which violates free variable normal form. If the rule is $\forall$-right or $\exists$-left, and the eigenvariable b which is eliminated occurs somewhere in the proof other than above this rule, then replace b by a new variable b' (which does not occur elsewhere in the proof) in every sequent above this rule. If the rule is cut, $\exists$-right, or $\forall$-left, then replace every variable eliminated by the rule by the same constant symbol e in every sequent above the rule (so now the rule does not eliminate any free variable).

II.2.5. Completeness of *LK* without Equality.

Notation. Let Φ be a set of formulas. Then $\longrightarrow \Phi$ is the set of all sequents of the form $\longrightarrow A$, where A is in Φ.

Definition II.2.21. Assume that the underlying vocabulary does not contain $=$. If Φ is a set of formulas, then an ***LK***-Φ proof is an ***LK*** proof in which sequents at the leaves may be either logical axioms or nonlogical axioms of the form $\longrightarrow A$, where A is in Φ.

Notice that a structure $\mathcal{M}$ satisfies $\longrightarrow \Phi$ iff $\mathcal{M}$ satisfies Φ. Also a sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of $\longrightarrow \Phi$ iff $\Gamma \longrightarrow \Delta$ is a logical consequence of Φ.

We would like to be able to say that a sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of a set Φ of formulas iff there is an ***LK***-Φ proof of $\Gamma \longrightarrow \Delta$. Unfortunately the soundness direction of the assertion is false. For example, using the $\forall$-right rule we can derive $\longrightarrow \forall x Px$ from $\longrightarrow Pb$, but $\longrightarrow \forall x Px$ is not a logical consequence of Pb.

We could correct the soundness statement by asserting it true for sentences, but we want to generalize this a little by introducing the notion of the universal closure of a formula or sequent.

Definition II.2.22. Suppose that A is a formula whose free variables comprise the list $a_1, \ldots, a_n$. Then the *universal closure* of A, written $\forall A$, is the sentence $\forall x_1 \ldots \forall x_n A(x_1/a_1, \ldots, x_n/a_n)$, where $x_1, \ldots, x_n$ is a list of new (bound) variables. If Φ is a set of formulas, then $\forall\Phi$ is the set of all sentences $\forall A$, for A in Φ.

Notice that if A is a sentence (i.e., it has no free variables), then $\forall A$ is the same as A.

Initially we study the case in which the underlying vocabulary does not contain $=$. To handle the case in which $=$ occurs we must introduce equality axioms. This will be done later.

Theorem II.2.23 (Derivational Soundness and Completeness of ***LK***). *Assume that the underlying vocabulary does not contain* $=$. *Let* Φ *be a set of formulas and let* $\Gamma \longrightarrow \Delta$ *be a sequent. Then there is an* ***LK***-Φ *proof of* $\Gamma \longrightarrow \Delta$ *iff* $\Gamma \longrightarrow \Delta$ *is a logical consequence of* $\forall\Phi$. *The soundness* (*only if*) *direction holds also when the underlying vocabulary contains* $=$.

Proof of Soundness. Let π be a ***LK***-Φ proof of $\Gamma \longrightarrow \Delta$. We must show that $\Gamma \longrightarrow \Delta$ is a logical consequence of $\forall\Phi$. We want to prove this by induction on the number of sequents in the proof π, but in fact we need a stronger induction hypothesis, to the effect that the "closure" of $\Gamma \longrightarrow \Delta$ is a logical consequence of $\forall\Phi$. So we first have to define the closure of a sequent.

Thus we define the closure $\forall S$ of a sequent S to be the closure of its associated formula A_S (Definition II.2.16). Note that if $S =_{\text{syn}} \Gamma \longrightarrow \Delta$, then $\forall S$ is *not* equivalent to $\forall\Gamma \longrightarrow \forall\Delta$ in general.

We now prove by induction on the number of sequents in π, that if π is an ***LK***-Φ proof of a sequent S, then $\forall S$ is a logical consequence of $\forall\Phi$. Since $\forall S \models S$, it follows that S itself is a logical consequence of $\forall\Phi$, and so Soundness follows.

For the base case, the sequent S is either a logical axiom, which is valid and hence a consequence of $\forall\Phi$, or it is a nonlogical axiom $\longrightarrow A$, where A is a formula in Φ. In the latter case, $\forall S$ is equivalent to $\forall A$, which of course is a logical consequence of $\forall\Phi$.

For the induction step, it is sufficient to check that for each rule of ***LK***, the closure of the bottom sequent is a logical consequence of the closure(s) of the sequent(s) on top. With two exceptions, this statement is true when the word "closure" is omitted, and adding back the word "closure" does not change the argument much. The two exceptions are the rules $\forall$-right and $\exists$-left. For these, the bottom is *not* a logical consequence of the top in general, but an easy argument shows that the closures of the top and bottom are equivalent. □

The proof of completeness is more difficult and more interesting than the proof of soundness. The following lemma lies at the heart of this proof.

Lemma II.2.24 (Completeness). *Assume that the underlying vocabulary does not contain* $=$. *If* $\Gamma \longrightarrow \Delta$ *is a sequent and* Φ *is a* (*possibly infinite*) *set of formulas such that* $\Gamma \longrightarrow \Delta$ *is a logical consequence of* Φ, *then there is a finite subset* $\{C_1, \ldots, C_n\}$ *of* Φ *such that the sequent*

$$C_1, \ldots, C_n, \Gamma \longrightarrow \Delta$$

has an ***LK*** *proof* π *which does not use the cut rule.*

Note that a form of the Compactness Theorem for predicate calculus sentences without equality follows from the above lemma. See Theorem II.4.2 for a more general form of compactness.

Proof of Derivational Completeness. Let Φ be a set of formulas such that $\Gamma \longrightarrow \Delta$ is a logical consequence of $\forall\Phi$. By the completeness lemma, there is a finite subset $\{C_1, \ldots, C_n\}$ of Φ such that

$$\forall C_1, \ldots, \forall C_n, \Gamma \longrightarrow \Delta$$

has a cut-free ***LK*** proof π. Note that for each $i, 1 \leq i \leq n$, the sequent $\longrightarrow \forall C_i$ has an ***LK***-Φ proof from the nonlogical axiom $\longrightarrow C_i$ by repeated use of the rule $\forall$-right. Now the proof π can be extended, using these proofs of the sequents

$$\longrightarrow \forall C_1, \quad \ldots, \quad \longrightarrow \forall C_n$$

and repeated use of the cut rule, to form an ***LK***-Φ proof $\Gamma \longrightarrow \Delta$. □

Proof of the Completeness Lemma. We loosely follow the proof of the Cut-free Completeness Theorem, pp. 33–36 of Buss [27]. (Warning: our definition of logical consequence differs from Buss's when the formulas in the hypotheses have free variables.) We will only prove it for the case in which the underlying first-order vocabulary $\mathcal{L}$ has a countable set (including the case of a finite set) of function and predicate symbols; i.e., the function symbols form a list $f_1, f_2, \ldots$ and the predicate symbols form a list $P_1, P_2, \ldots$. This may not seem like much of a restriction, but for example in developing the model theory of the real numbers, it is sometimes useful to introduce a distinct constant symbol e_c for every real number c; and there are uncountably many real numbers. The completeness theorem and lemma hold for the uncountable case, but we shall not prove them for this case.

For the countable case, we may assign a distinct binary string to each function symbol, predicate symbol, variable, etc., and hence assign a unique binary string to each formula and term. This allows us to enumerate all the $\mathcal{L}$-formulas in a list $A_1, A_2, \ldots$ and enumerate all the $\mathcal{L}$-terms (which contain only free variables $a, b, c, \ldots$) in a list $t_1, t_2, \ldots$. The free variables available to build the formulas and terms in these lists must include all the free variables which appear in Φ, together with a countably infinite set $\{c_0, c_1, \ldots\}$ of new free variables which do not occur in any of the formulas in Φ. (These new free variables are needed for the cases $\exists$-left and $\forall$-right in the argument below.) Further we may assume that every formula occurs infinitely often in the list of formulas, and every term occurs infinitely often in the list of terms. Finally we may enumerate all pairs $\langle A_i, t_j \rangle$, using any method of enumerating all pairs of natural numbers.

We are trying to find an ***LK*** proof of some sequent of the form

$$C_1, \ldots, C_n, \Gamma \longrightarrow \Delta$$

for some n. Starting with $\Gamma \longrightarrow \Delta$ at the bottom, we work upward by applying the rules in reverse, much as in the proof of the propositional completeness theorem for ***PK***. However now we will add formulas C_i to the antecedent from time to time. Also unlike the ***PK*** case we have no inversion principle to work with (specifically for the rules $\forall$-left and $\exists$-right). Thus it may happen that our proof-building procedure may not terminate. In this case we will show how to define a structure which shows that $\Gamma \longrightarrow \Delta$ is not a logical consequence of Φ.

We construct our cut-free proof tree π in stages. Initially π consists of just the sequent $\Gamma \longrightarrow \Delta$. At each stage we modify π by possibly adding a formula from Φ to the antecedent of every sequent in π, and by adding subtrees to some of the leaves.

NOTATION. A sequent in π is said to be *active* provided it is at a leaf and cannot be immediately derived from a logical axiom (i.e., no formula occurs in both its antecedent and succedent, the logical constant $\top$ does not occur in its succedent, and $\bot$ does not occur in its antecedent).

Each stage uses one pair in our enumeration of all pairs $\langle A_i, t_j \rangle$. Here is the procedure for the next stage, in general.

Let $\langle A_i, t_j \rangle$ be the next pair in the enumeration. We call A_i the *active* formula for this stage.

Step 1. If A_i is in Φ, then replace every sequent $\Gamma' \longrightarrow \Delta'$ in π with the sequent $\Gamma', A_i \longrightarrow \Delta'$.

Step 2. If A_i is atomic, do nothing and proceed to the next stage. Otherwise, modify π at the active sequents which contain A_i by applying the appropriate introduction rule in reverse, much as in the proof of propositional completeness (Theorem II.1.8). (It suffices to pick any one occurrence of A_i in each active sequent.) For example, if A_i is of the form $B \vee C$, then every active sequent in π of the form $\Gamma', B \vee C, \Gamma'' \longrightarrow \Delta'$ is replaced by the derivation

$$\dfrac{\Gamma', B, \Gamma'' \longrightarrow \Delta' \qquad \Gamma', C, \Gamma'' \longrightarrow \Delta'}{\Gamma', B \vee C, \Gamma'' \longrightarrow \Delta'}$$

Here the double line represents a derivation involving the rule $\vee$-left, together with exchanges to move the formulas B, C to the left end of the antecedent and move $B \vee C$ back to the right. The treatment is similar when $B \vee C$ occurs in the succedent, only the rule $\vee$-right is used.

If A_i is of the form $\exists x B(x)$, then every active sequent of π of the form $\Gamma', \exists x B(x), \Gamma'' \longrightarrow \Delta'$ is replaced by the derivation

$$\dfrac{\Gamma', B(c), \Gamma'' \longrightarrow \Delta'}{\Gamma', \exists x B(x), \Gamma'' \longrightarrow \Delta'}$$

where c is a new free variable, not used in π yet. (Also c may not occur in any formula in Φ, because otherwise at a later stage, *Step* 1 of the procedure

might cause the variable restriction in the $\exists$-left rule to be violated.) In addition, any active sequent of the form $\Gamma' \longrightarrow \Delta', \exists x B(x), \Delta''$ is replaced by the derivation

$$\dfrac{\Gamma' \longrightarrow \Delta', \exists x B(x), B(t_j), \Delta''}{\Gamma' \longrightarrow \Delta', \exists x B(x), \Delta''}$$

Here the term t_j is the second component in the current pair $\langle A_i, t_j \rangle$. The derivation uses the rule $\exists$-right to introduce a new copy of $\exists x B(x)$, and then the rule contraction-right to combine the two copies of $\exists x B(x)$. This and the dual $\forall$-left case are the only two cases that use the term t_j, and the only cases that use the contraction rule.

The case where A_i begins with a universal quantifier is dual to the above existential case.

Step 3. If there are no active sequents remaining in π, then exit from the algorithm. Otherwise continue to the next stage.

Exercise II.2.25. Carry out the case above in which A_i begins with a universal quantifier.

If the algorithm constructing π ever halts, then π gives a cut-free proof of $\Gamma, C_1, \ldots, C_n \longrightarrow \Delta$ for some formulas $C_1, \ldots, C_n$ in Φ. This is because the nonactive leaf sequents all can be derived from the logical axioms using weakenings and exchanges. Thus π can be extended, using exchanges, to a cut-free proof of $C_1, \ldots, C_n, \Gamma \longrightarrow \Delta$, as desired.

It remains to show that if the above algorithm constructing π never halts, then the sequent $\Gamma \longrightarrow \Delta$ is not a logical consequence of Φ. So suppose the algorithm never halts, and let π be the result of running the algorithm forever. In general, π will be an infinite tree, although in special cases π is a finite tree. In general the objects at the nodes of the tree will not be finite sequents, but because of *Step* 1 of the algorithm above, they will be of the form $\Gamma', C_1, C_2, \ldots \longrightarrow \Delta'$, where $C_1, C_2, \ldots$ is an infinite sequence of formulas containing all formulas in Φ, each repeated infinitely often (unless Φ is empty). We shall refer to these infinite pseudo-sequents as just "sequents".

If π has only finitely many nodes, then at least one leaf node must be active (and contain only atomic formulas), since otherwise the algorithm would terminate. In this case, let β be a path in π from the root extending up to this active node. If on the other hand π has infinitely many nodes, then by Lemma II.1.18 (König), there must be an infinite branch β in π starting at the root and extending up through the tree. Thus in either case, β is a branch in π starting at the root, extending up through the tree, and such that all sequents on β were once active, and hence have no formula occurring on both the left and right, no $\top$ on the right and no $\bot$ on the left.

We use this branch β to construct a structure $\mathcal{M}$ and an object assignment σ which satisfy every formula in Φ, but falsify the sequent $\Gamma \longrightarrow \Delta$ (so $\Gamma \longrightarrow \Delta$ is not a logical consequence of Φ).

Definition II.2.26 (Construction of the "Term Model" $\mathcal{M}$). The universe M of $\mathcal{M}$ is the set of all $\mathcal{L}$-terms t (which contain only "free" variables $a, b, c, \ldots$). The object assignment σ just maps every variable a to itself.

The interpretation $f^{\mathcal{M}}$ of each k-ary function symbol f is defined so that $f^{\mathcal{M}}(r_1, \ldots, r_k)$ is the term $fr_1 \ldots r_k$, where $r_1, \ldots, r_k$ are any terms (i.e., any members of the universe). The interpretation $P^{\mathcal{M}}$ of each k-ary predicate symbol P is defined by letting $P^{\mathcal{M}}(r_1, \ldots, r_k)$ hold iff the atomic formula $Pr_1 \ldots r_k$ occurs in the antecedent (left side) of some sequent in the branch β.

Exercise II.2.27. Prove by structural induction that for every term t, $t^{\mathcal{M}}[\sigma] = t$.

Claim. For every formula A, if A occurs in some antecedent in the branch β, then $\mathcal{M}$ and σ satisfy A, and if A occurs in some succedent in β, then $\mathcal{M}$ and σ falsify A.

Since the root of π is the sequent $\Gamma, C_1, C_2, \ldots \longrightarrow \Delta$, where $C_1, C_2, \ldots$ contains all formulas in Φ, it follows that $\mathcal{M}$ and σ satisfy Φ and falsify $\Gamma \longrightarrow \Delta$.

We prove the Claim by structural induction on formulas A. For the base case, if A is an atomic formula, then by the definition of $P^{\mathcal{M}}$ above, A is satisfied iff A occurs in some antecedent of β or $A = \top$. But no atomic formula can occur both in an antecedent of some node in β and in a succedent (of possibly some other node) in β, since then these formulas would persist upward in β so that some particular sequent in β would have A occurring both on the left and on the right. Thus if A occurs in some succedent of β, it is not satisfied by $\mathcal{M}$ and σ (recall that $\top$ does not occur in any succedent of β).

For the induction step, there is a different case for each of the ways of constructing a formula from simpler formulas (see Definition II.2.2). In general, if A occurs in some sequent in β, then A persists upward in every higher sequent of β until it becomes the active formula ($A =_{\text{syn}} A_i$). Each case is handled by the corresponding introduction rule used in the algorithm. For example, if A is of the form $B \vee C$ and A occurs on the left of a sequent in β, then the rule $\vee$-left is applied in reverse, so that when β is extended upward either it will have some antecedent containing B or one containing C. In the case of B, we know that $\mathcal{M}$ and σ satisfy B by the induction hypothesis, and hence they satisfy $B \vee C$. (Similarly for C.)

Now consider the interesting case in which A is $\exists x B(x)$ and A occurs in some succedent of β. (See Step 2 above to find out what happens

when A becomes active in this case.) The path β will hit a succedent with $B(t_j)$ in the succedent, and by the induction hypothesis, $\mathcal{M}$ and σ falsify $B(t_j)$. But this succedent still has a copy of $\exists x B(x)$, and in fact this copy will be in *every* succedent of β above this point. Hence *every* $\mathcal{L}$-term t will eventually be of the form t_j and so the formula $B(t)$ will occur as a succedent on β. (This is why we assumed that every term appears infinitely often in the sequence $t_1, t_2, \ldots$.) Therefore $\mathcal{M}$ and σ falsify $B(t)$ for every term t (i.e., for every element in the universe of $\mathcal{M}$). Therefore they falsify $\exists x B(x)$, as required.

This and the dual case in which A is $\forall x B(x)$ and occurs in some antecedent of β are the only subtle cases. All other cases are straightforward. □

We now wish to strengthen the derivational completeness of ***LK*** and show that cuts can be restricted so that cut formulas are in Φ. The definition of *anchored* ***PK*** proof (Definition II.1.12) can be generalized to *anchored* ***LK*** proof. We will continue to restrict our attention to the case in which all nonlogical axioms have the simple form $\longrightarrow A$, although an analog of the following theorem does hold for an arbitrary set of nonlogical axioms, provided they are closed under substitution of terms for variables.

Theorem II.2.28 (Anchored ***LK*** Completeness). *Assume that the underlying vocabulary does not contain* $=$. *Suppose that* Φ *is a set of formulas closed under substitution of terms for variables. (I.e., if* $A(b)$ *is in* Φ, *and* t *is any term not containing "bound" variables* $x, y, z, \ldots$, *then* $A(t)$ *is also in* Φ.) *Suppose that* $\Gamma \longrightarrow \Delta$ *is a sequent that is a logical consequence of* $\forall\Phi$. *Then there is an* ***LK***-Φ *proof of* $\Gamma \longrightarrow \Delta$ *in which the cut rule is restricted so that the only cut formulas are formulas in* Φ.

Note that if all formulas in Φ are sentences, then the above theorem follows easily from the Completeness Lemma, since in this case $\forall\Phi$ is the same as Φ. However if formulas in Φ have free variables, then apparently the cut rule must be applied to the closures $\forall C$ of formulas C in Φ (as opposed to C itself) in order to get an ***LK***-Φ proof of $\Gamma \longrightarrow \Delta$. It will be important later, in our proof of witnessing theorems, that cuts can be restricted to the formulas C.

Exercise II.2.29. Show how to modify the proof of the Completeness Lemma to obtain a proof of the Anchored ***LK*** Completeness Theorem. Explain the following modifications to that proof.

(a) The definition of *active sequent* on page 27 must be modified, since now we are allowing nonlogical axioms in π. Give the precise new definition.

(b) *Step* 1 of the procedure on page 27 must be modified, because now we are looking for a derivation of $\Gamma \longrightarrow \Delta$ from nonlogical axioms,

rather than a proof of $C_1, \ldots, C_n, \Gamma \longrightarrow \Delta$. Describe the modification. (We still need to bring formulas A_i of Φ somehow into the proof, and your modification will involve adding a short derivation to π.)

(c) The restriction given in Step 2 for the case in which $\exists x B(x)$ is in the antecedent, that the variable c must not occur in any formula in Φ, must be dropped. Explain why.

(d) Explain why the term model $\mathcal{M}$ and object assignment σ, described on page 29 (Definition II.2.26), satisfy $\forall\Phi$. This should follow from the Claim on page 29, and your modification of *Step* 1, which should ensure that each formula in Φ occurs in the antecedent of some sequent in every branch in π. Conclude that $\Gamma \longrightarrow \Delta$ is not a logical consequence of $\forall\Phi$ (when the procedure does not terminate).

II.3. Equality Axioms

Definition II.3.1. A *weak* $\mathcal{L}$-structure $\mathcal{M}$ is an $\mathcal{L}$-structure in which we drop the requirement that $=^{\mathcal{M}}$ is the equality relation (i.e., $=^{\mathcal{M}}$ can be any binary relation on M.)

Are there sentences $\mathcal{E}$ (axioms for equality) such that a weak structure $\mathcal{M}$ satisfies $\mathcal{E}$ iff $\mathcal{M}$ is a (proper) structure? It is easy to see that no such set $\mathcal{E}$ of axioms exists, because we can always inflate a point in a weak model to a set of equivalent points.

Nevertheless every vocabulary $\mathcal{L}$ has a standard set $\mathcal{E}_{\mathcal{L}}$ of equality axioms which satisfies the Equality Theorem below.

Definition II.3.2 (Equality Axioms of $\mathcal{L}$ ($\mathcal{E}_{\mathcal{L}}$)).

EA1. $\forall x(x = x)$ (reflexivity);

EA2. $\forall x \forall y(x = y \supset y = x)$ (symmetry);

EA3. $\forall x \forall y \forall z((x = y \wedge y = z) \supset x = z)$ (transitivity);

EA4. $\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_n(x_1 = y_1 \wedge \cdots \wedge x_n = y_n) \supset f x_1 \ldots x_n = f y_1 \ldots y_n$ for each $n \geq 1$ and each n-ary function symbol f in $\mathcal{L}$.

EA5. $\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_n(x_1 = y_1 \wedge \cdots \wedge x_n = y_n) \supset (P x_1 \ldots x_n \supset P y_1 \ldots y_n)$ for each $n \geq 1$ and each n-ary predicate symbol P in $\mathcal{L}$ other than $=$.

Axioms **EA**1, **EA**2, **EA**3 assert that $=$ is an equivalence relation. Axiom **EA**4 asserts that functions respect the equivalence classes, and Axiom **EA**5 asserts that predicates respect equivalence classes. Together the axioms assert that $=$ is a congruence relation with respect to the function and predicate symbols.

Note that the equality axioms are all valid, because of our requirement that $=$ be interpreted as equality in any (proper) structure.

THEOREM II.3.3 (Equality). *Let Φ be any set of $\mathcal{L}$-formulas. Then Φ is satisfiable iff $\Phi \cup \mathcal{E}_{\mathcal{L}}$ is satisfied by some weak $\mathcal{L}$-structure.*

COROLLARY II.3.4. *$\Phi \models A$ iff for every weak $\mathcal{L}$-structure $\mathcal{M}$ and every object assignment σ, if $\mathcal{M}$ satisfies $\Phi \cup \mathcal{E}_{\mathcal{L}}$ under σ then $\mathcal{M}$ satisfies A under σ.*

COROLLARY II.3.5. *$\forall\Phi \models A$ iff A has an **LK**-Ψ proof, where $\Psi = \Phi \cup \mathcal{E}_{\mathcal{L}}$.*

Corollary II.3.4 follows immediately from the Equality Theorem and the fact that $\Phi \models A$ iff $\Phi \cup \{\neg A\}$ is unsatisfiable. Corollary II.3.5 follows from Corollary II.3.4 and the derivational soundness and completeness of ***LK*** (page 25), where in applying that theorem we treat $=$ as just another binary relation (so we can assume $\mathcal{L}$ does not have the official equality symbol).

PROOF OF EQUALITY. The ONLY IF ($\Longrightarrow$) direction is obvious, because every structure $\mathcal{M}$ must interpret $=$ as true equality, and hence $\mathcal{M}$ satisfies the equality axioms $\mathcal{E}_{\mathcal{L}}$.

For the IF ($\Longleftarrow$) direction, suppose that $\mathcal{M}$ is a weak $\mathcal{L}$-structure with universe M, such that $\mathcal{M}$ satisfies $\Phi \cup \mathcal{E}_{\mathcal{L}}$. Our job is to construct a proper structure $\hat{\mathcal{M}}$ such that $\hat{\mathcal{M}}$ satisfies Φ. The idea is to let the elements of $\hat{\mathcal{M}}$ be the equivalence classes under the equivalence relation $=^{\mathcal{M}}$. Axioms **EA**4 and **EA**5 insure that the interpretation of each function and predicate symbol under $\mathcal{M}$ induces a corresponding function or predicate in $\hat{\mathcal{M}}$. Further each object assignment σ for $\mathcal{M}$ induces an object assignment $\hat{\sigma}$ on $\hat{\mathcal{M}}$. Then for every formula A and object assignment σ, we show by structural induction on A that $\mathcal{M} \models A[\sigma]$ iff $\hat{\mathcal{M}} \models A[\hat{\sigma}]$. □

II.3.1. Equality Axioms for *LK*. For the purpose of using an ***LK*** proof to establish $\Phi \models A$, we can replace the standard equality axioms **EA**1, ..., **EA**5 by the following quantifier-free sequent schemes, where we must include an instance of the sequent for all terms t, u, v, t_i, u_i (not involving "bound" variables $x, y, z, \ldots$).

DEFINITION II.3.6 (Equality Axioms for ***LK***).

E1. $\longrightarrow t = t$;
E2. $t = u \longrightarrow u = t$;
E3. $t = u, u = v \longrightarrow t = v$;
E4. $t_1 = u_1, \ldots, t_n = u_n \longrightarrow f t_1 \ldots t_n = f u_1 \ldots u_n$, for each f in $\mathcal{L}$;
E5. $t_1 = u_1, \ldots, t_n = u_n, P t_1 \ldots t_n \longrightarrow P u_1 \ldots u_n$, for each P in $\mathcal{L}$ (here P is not $=$).

Note that the universal closures of **E**1, ..., **E**5 are semantically equivalent to **EA**1, ..., **EA**5, and in fact using the ***LK*** rule $\forall$-right repeatedly, $\longrightarrow$ **EA***i* is easily derived in ***LK*** from **E***i* (with terms t, u, etc., taken to be distinct variables), $i = 1, \ldots, 5$. Thus Corollary II.3.5 above still holds when $\Psi = \Phi \cup \{\mathbf{E}1, \ldots, \mathbf{E}5\}$.