CHAPTER I

An Introduction to Proof Theory

Samuel R. Buss
Departments of Mathematics and Computer Science, University of California, San Diego
La Jolla, California 92093-0112, USA

Contents

HANDBOOK OF PROOF THEORY
Edited by S. R. Buss

Proof Theory is the area of mathematics which studies the concepts of mathematical proof and mathematical provability. Since the notion of "proof" plays a central role in mathematics as the means by which the truth or falsity of mathematical propositions is established; Proof Theory is, in principle at least, the study of the foundations of all of mathematics. Of course, the use of Proof Theory as a foundation for mathematics is of necessity somewhat circular, since Proof Theory is itself a subfield of mathematics.

There are two distinct viewpoints of what a mathematical proof is. The first view is that proofs are social conventions by which mathematicians convince one another of the truth of theorems. That is to say, a proof is expressed in natural language plus possibly symbols and figures, and is sufficient to convince an expert of the correctness of a theorem. Examples of social proofs include the kinds of proofs that are presented in conversations or published in articles. Of course, it is impossible to precisely define what constitutes a valid proof in this social sense; and, the standards for valid proofs may vary with the audience and over time. The second view of proofs is more narrow in scope: in this view, a proof consists of a string of symbols which satisfy some precisely stated set of rules and which prove a theorem, which itself must also be expressed as a string of symbols. According to this view, mathematics can be regarded as a 'game' played with strings of symbols according to some precisely defined rules. Proofs of the latter kind are called "formal" proofs to distinguish them from "social" proofs.

In practice, social proofs and formal proofs are very closely related. Firstly, a formal proof can serve as a social proof (although it may be very tedious and unintuitive) provided it is formalized in a proof system whose validity is trusted. Secondly, the standards for social proofs are sufficiently high that, in order for a proof to be socially accepted, it should be possible (in principle!) to generate a formal proof corresponding to the social proof. Indeed, this offers an explanation for the fact that there are generally accepted standards for social proofs; namely, the implicit requirement that proofs can be expressed, in principle, in a formal proof system enforces and determines the generally accepted standards for social proofs.

Proof Theory is concerned almost exclusively with the study of formal proofs: this is justified, in part, by the close connection between social and formal proofs, and it is necessitated by the fact that only formal proofs are subject to mathematical analysis. The principal tasks of Proof Theory can be summarized as follows. First, to formulate systems of logic and sets of axioms which are appropriate for formalizing mathematical proofs and to characterize what results of mathematics follow from certain axioms; or, in other words, to investigate the proof-theoretic strength of particular formal systems. Second, to study the structure of formal proofs; for instance, to find normal forms for proofs and to establish syntactic facts about proofs. This is the study of proofs as objects of independent interest. Third, to study what kind of additional information can be extracted from proofs beyond the truth of the theorem being proved. In certain cases, proofs may contain computational or constructive information. Fourth, to study how best to construct formal proofs; e.g., what kinds of proofs can be efficiently generated by computers?

The study of Proof Theory is traditionally motivated by the problem of formalizing mathematical proofs; the original formulation of first-order logic by Frege [1879] was the first successful step in this direction. Increasingly, there have been attempts to extend Mathematical Logic to be applicable to other domains; for example, intuitionistic logic deals with the formalization of constructive proofs, and logic programming is a widely used tool for artificial intelligence. In these and other domains, Proof Theory is of central importance because of the possibility of computer generation and manipulation of formal proofs.

This handbook covers the central areas of Proof Theory, especially the mathematical aspects of Proof Theory, but largely omits the philosophical aspects of proof theory. This first chapter is intended to be an overview and introduction to mathematical proof theory. It concentrates on the proof theory of *classical* logic, especially propositional logic and first-order logic. This is for two reasons: firstly, classical first-order logic is by far the most widely used framework for mathematical reasoning, and secondly, many results and techniques of classical first-order logic frequently carryover with relatively minor modifications to other logics.

This introductory chapter will deal primarily with the sequent calculus, and resolution, and to lesser extent, the Hilbert-style proof systems and the natural deduction proof system. We first examine proof systems for propositional logic, then proof systems for first-order logic. Next we consider some applications of cut elimination, which is arguably the central theorem of proof theory. Finally, we review the proof theory of some non-classical logics, including intuitionistic logic and linear logic.

1. Proof theory of propositional logic

Classical propositional logic, also called sentential logic, deals with sentences and propositions as abstract units which take on distinct True/False values. The basic syntactic units of propositional logic are *variables* which represent atomic propositions which may have value either *True* or *False*. Propositional variables are combined with Boolean functions (also called connectives): a *k-ary Boolean function* is a mapping from $\{T,F\}^k$ to $\{T,F\}$ where we use T and F to represent *True* and *False*. The most frequently used examples of Boolean functions are the connectives $\top$ and $\bot$ which are the 0-ary functions with values T and F, respectively; the binary connectives $\land$, $\lor$, $\supset$, $\leftrightarrow$ and $\oplus$ for "and", "or", "if-then", "if-and-only-if" and "parity"; and the unary connective $\neg$ for negation. Note that $\lor$ is the inclusive-or and $\oplus$ is the exclusive-or.

We shall henceforth let the set of propositional variables be $V = \{p_1, p_2, p_3, \ldots\}$; however, our theorems below hold also for uncountable sets of propositional variables. The set of formulas is inductively defined by stating that every propositional variable is a formula, and that if A and B are formulas, then $(\neg A)$, $(A \land B)$, $(A \lor B)$, $(A \supset B)$, etc., are formulas. A *truth assignment* consists of an assignment of True/False values to the propositional variables, i.e., a truth assignment is a mapping $\tau : V \to \{T,F\}$.

A truth assignment can be extended to have domain the set of all formulas in the obvious way, according to Table 1; we write $\overline{\tau}(A)$ for the truth value of the formula A induced by the truth assignment τ.

Table 1
Values of a truth assignment $\overline{\tau}$

A	B	$(\neg A)$	$(A \land B)$	$(A \lor B)$	$(A \supset B)$	$(A \leftrightarrow B)$	$(A \oplus B)$
T	T	F	T	T	T	T	F
T	F	F	F	T	F	F	T
F	T	T	F	T	T	F	T
F	F	T	F	F	T	T	F

A formula A involving only variables among $p_1, \ldots, p_k$ defines a k-ary Boolean function f_A, by letting $f_A(x_1, ..., x_k)$ equal the truth value $\overline{\tau}(A)$ where $\tau(p_i) = x_i$ for all i. A *language* is a set of connectives which may be used in the formation of L*-formulas.* A language L is *complete* if and only if every Boolean function can be defined by an L-formula. Propositional logic can be formulated with any complete (usually finite) language L — for the time being, we shall use the language $\neg$, $\land$, $\lor$ and $\supset$.

A propositional formula A is said to be a *tautology* or to be *(classically) valid* if A is assigned the value T by every truth assignment. We write $\vDash A$ to denote that A is a tautology. The formula A is *satisfiable* if there is some truth assignment that gives it value T. If Γ is a set of propositional formulas, then Γ is *satisfiable* if there is some truth assignment that simultaneously satisfies all members of Γ. We say Γ *tautologically implies* A, or $\Gamma \vDash A$, if every truth assignment which satisfies Γ also satisfies A.

One of the central problems of propositional logic is to find useful methods for recognizing tautologies; since A is a tautology if and only if $\neg A$ is not satisfiable, this is essentially the same as the problem of finding methods for recognizing satisfiable formulas. Of course, the set of tautologies is decidable, since to verify that a formula A with n distinct propositional variables is a tautology, one need merely check that the 2^n distinct truth assignments to these variables all give A the value T. This brute-force 'method of truth-tables' is not entirely satisfactory; firstly, because it can involve an exorbitant amount of computation, and secondly, because it provides no intuition as to *why* the formula is, or is not, a tautology.

For these reasons, it is often advantageous to *prove* that A is a tautology instead of using the method of truth-tables. The next three sections discuss three commonly used propositional proof systems. The so-called Frege proof systems are perhaps the most widely used and are based on modus ponens. The sequent calculus systems provide an elegant proof system which combines both the possibility of elegant proofs and the advantage of an extremely useful normal form for proofs. The resolution refutation proof systems are designed to allow for efficient computerized search for proofs. Later, we will extend these three systems to first-order logic.

1.1. Frege proof systems

The mostly commonly used propositional proof systems are based on the use of *modus ponens* as the sole rule of inference. Modus ponens is the inference rule, which allows, for arbitrary A and B, the formula B to be inferred from the two hypotheses $A \supset B$ and A; this is pictorially represented as

$$\frac{A \qquad A \supset B}{B}$$

In addition to this rule of inference, we need *logical axioms* that allow the inference of 'self-evident' tautologies from no hypotheses. There are many possible choices for sets of axioms: obviously, we wish to have a sufficiently strong set of axioms so that every tautology can be derived from the axioms by use of modus ponens. In addition, we wish to specify the axioms by a finite set of schemes.

1.1.1. Definition. A *substitution* σ is a mapping from the set of propositional variables to the set of propositional formulas. If A is a propositional formula, then the result of applying σ to A is denoted $A\sigma$ and is equal to the formula obtained by simultaneously replacing each variable appearing in A by its image under σ.

An example of a set of axiom schemes over the language $\neg$, $\wedge$, $\vee$ and $\supset$ is given in the next definition. We adopt conventions for omitting parentheses from descriptions of formulas by specifying that unary operators have the highest precedence, the connectives $\wedge$ and $\vee$ have second highest precedence, and that $\supset$ and $\leftrightarrow$ have lowest precedence. All connectives of the same precedence are to be associated from right to left; for example, $A \supset \neg B \supset C$ is a shorthand representation for the formula $(A \supset ((\neg B) \supset C))$.

1.1.2. Definition. Consider the following set of axiom schemes:

$$\begin{array}{ll}
p_1 \supset (p_2 \supset p_1) & (p_1 \supset p_2) \supset (p_1 \supset \neg p_2) \supset \neg p_1 \\
(p_1 \supset p_2) \supset (p_1 \supset (p_2 \supset p_3)) \supset (p_1 \supset p_3) & (\neg\neg p_1) \supset p_1 \\
p_1 \supset p_1 \vee p_2 & p_1 \wedge p_2 \supset p_1 \\
p_2 \supset p_1 \vee p_2 & p_1 \wedge p_2 \supset p_2 \\
(p_1 \supset p_3) \supset (p_2 \supset p_3) \supset (p_1 \vee p_2 \supset p_3) & p_1 \supset p_2 \supset p_1 \wedge p_2
\end{array}$$

The propositional proof system $\mathcal{F}$ is defined to have as its axioms every substitution instance of the above formulas and to have modus ponens as its only rule. An $\mathcal{F}$-proof of a formula A is a sequence of formulas, each of which is either an $\mathcal{F}$-axiom or is inferred by modus ponens from two earlier formulas in the proof, such that the final formula in the proof is A.

We write $\vdash_{\mathcal{F}} A$, or just $\vdash A$, to say that A has an $\mathcal{F}$-proof. We write $\Gamma \vdash_{\mathcal{F}} A$, or just $\Gamma \vdash A$, to say that A has a proof in which each formula either is deduced according the axioms or inference rule of $\mathcal{F}$ or is in Γ. In this case, we say that A is proved from the extra-logical hypotheses Γ; note that Γ may contain formulas which are not tautologies.

1.1.3. Soundness and completeness of $\mathcal{F}$. It is easy to prove that every $\mathcal{F}$-provable formula is a tautology, by noting that all axioms of $\mathcal{F}$ are valid and that modus ponens preserves the property of being valid. Similarly, whenever $\Gamma \vdash_{\mathcal{F}} A$, then Γ tautologically implies A. In other words, $\mathcal{F}$ is *(implicationally) sound*; which means that all provable formulas are valid (or, are consequences of the extra-logical hypotheses Γ).

Of course, any useful proof system ought to be sound, since the purpose of creating proofs is to establish the validity of a sentence. Remarkably, the system $\mathcal{F}$ is also *complete* in that it can prove any valid formula. Thus the semantic notion of validity and the syntactic notion of provability coincide, and a formula is valid if and only if it is provable in $\mathcal{F}$.

Theorem. *The propositional proof system $\mathcal{F}$ is complete and is implicationally complete; namely,*
(1) *If A is a tautology, then $\vdash_{\mathcal{F}} A$.*
(2) *If $\Gamma \vDash A$, then $\Gamma \vdash_{\mathcal{F}} A$.*

The philosophical significance of the completeness theorem is that a finite set of (schematic) axioms and rules of inference are sufficient to establish the validity of any tautology. In hindsight, it is not surprising that this holds, since the method of truth-tables already provides an algorithmic way of recognizing tautologies. Indeed, the proof of the completeness theorem given below, can be viewed as showing that the method of truth tables can be formalized within the system $\mathcal{F}$.

1.1.4. Proof. We first observe that part (2) of the completeness theorem can be reduced to part (1) by a two step process. Firstly, note that the compactness theorem for propositional logic states that if $\Gamma \vDash A$ then there is a finite subset Γ_0 of Γ which also tautologically implies A. A topological proof of the compactness theorem for propositional logic is sketched in 1.1.5 below. Thus Γ may, without loss of generality, be assumed to be a finite set of formulas, say $\Gamma = \{B_1, \ldots, B_k\}$. Secondly, note that $\Gamma \vDash A$ implies that $B_1 \supset B_2 \supset \cdots \supset A$ is a tautology. So, by part (1), the latter formula has an $\mathcal{F}$-proof, and by k additional modus ponens inferences, $\Gamma \vdash A$. (To simplify notation, we write $\vdash$ instead of $\vdash_{\mathcal{F}}$.)

It remains to prove part (1). We begin by establishing a series of special cases, (a)–(k), of the completeness theorem, in order to "bootstrap" the propositional system $\mathcal{F}$. We use symbols ϕ, ψ, χ for arbitrary formulas and Π to represent any set of formulas.

(a) $\vdash \phi \supset \phi$.
Proof: Combine the three axioms $(\phi \supset \phi \supset \phi) \supset (\phi \supset (\phi \supset \phi) \supset \phi) \supset (\phi \supset \phi)$, $\phi \supset (\phi \supset \phi)$ and $\phi \supset (\phi \supset \phi) \supset \phi$ with two uses of modus ponens.

(b) Deduction Theorem: $\Gamma, \phi \vdash \psi$ if and only if $\Gamma \vdash \phi \supset \psi$.
Proof: The reverse implication is trivial. To prove the forward implication, suppose $C_1, C_2, \ldots, C_k$ is an $\mathcal{F}$-proof of ψ from Γ, ϕ. This means that C_k is ψ and that each

C_i is ϕ, is in Γ, is an axiom, or is inferred by modus ponens. It is straightforward to prove, by induction on i, that $\Gamma \vdash \phi \supset C_i$ for each C_i.

(c) $\phi \supset \psi \vdash \neg\psi \supset \neg\phi$.
Proof: By the deduction theorem, it suffices to prove that $\phi \supset \psi, \neg\psi \vdash \neg\phi$. To prove this, use the two axioms $\neg\psi \supset (\phi \supset \neg\psi)$ and $(\phi \supset \psi) \supset (\phi \supset \neg\psi) \supset \neg\phi$ and three uses of modus ponens.

(d) $\phi, \neg\phi \vdash \psi$.
Proof: From the axiom $\phi \supset (\neg\psi \supset \phi)$, we have $\phi \vdash \neg\psi \supset \phi$. Thus, by (c) we get $\phi \vdash \neg\phi \supset \neg\neg\psi$, and by (b), $\phi, \neg\phi \vdash \neg\neg\psi$. Finally modus ponens with the axiom $\neg\neg\psi \supset \psi$ gives the desired result.

(e) $\neg\phi \vdash \phi \supset \psi$ and $\psi \vdash \phi \supset \psi$.
Proof: This former follows from (d) and the deduction theorem, and the latter follows from the axiom $\psi \supset (\phi \supset \psi)$.

(f) $\phi, \neg\psi \vdash \neg(\phi \supset \psi)$.
Proof: It suffices to prove $\phi \vdash \neg\psi \supset \neg(\phi \supset \psi)$. Thus, by (c) and the deduction theorem, it suffices to prove $\phi, \phi \supset \psi \vdash \psi$. The latter assertion is immediate from modus ponens.

(g) $\phi, \psi \vdash \phi \land \psi$.
Proof: Two uses of modus ponens with the axiom $\phi \supset \psi \supset (\phi \land \psi)$.

(h) $\neg\phi \vdash \neg(\phi \land \psi)$ and $\neg\psi \vdash \neg(\phi \land \psi)$.
Proof: For the first part, it suffices to show $\vdash \neg\phi \supset \neg(\phi \land \psi)$, and thus, by (c), it suffices to show $\vdash (\phi \land \psi) \supset \phi$, which is an axiom. The proof that $\neg\psi \vdash \neg(\phi \land \psi)$ is similar.

(i) $\phi \vdash \phi \lor \psi$ and $\psi \vdash \phi \lor \psi$.
Proof: $\phi \supset (\phi \lor \psi)$ and $\psi \supset (\phi \lor \psi)$ are axioms.

(j) $\neg\phi, \neg\psi \vdash \neg(\phi \lor \psi)$.
Proof: It suffices to prove $\neg\phi \vdash \neg\psi \supset \neg(\phi \lor \psi)$, and this, by (c), follows from $\neg\phi \vdash (\phi \lor \psi) \supset \psi$. For this, we combine
(i) $\neg\phi \vdash \phi \supset \psi$, by (e),
(ii) $\vdash \psi \supset \psi$, by (a),
(iii) $\vdash (\phi \supset \psi) \supset (\psi \supset \psi) \supset ((\phi \lor \psi) \supset \psi)$, an axiom,
with two uses of modus ponens.

(k) $\phi \vdash \neg\neg\phi$.
Proof: By (d), $\phi, \neg\phi \vdash \neg\neg\phi$, and obviously, $\phi, \neg\neg\phi \vdash \neg\neg\phi$. So $\phi \vdash \neg\neg\phi$ follows from the next lemma.

1.1.4.1. Lemma. *If* $\Gamma, \phi \vdash \psi$ *and* $\Gamma, \neg\phi \vdash \psi$*, then* $\Gamma \vdash \psi$.

Proof. By (b) and (c), the two hypotheses imply that $\Gamma \vdash \neg\psi \supset \neg\phi$ and $\Gamma \vdash \neg\psi \supset \neg\neg\phi$. These plus the two axioms $(\neg\psi \supset \neg\phi) \supset (\neg\psi \supset \neg\neg\phi) \supset \neg\neg\psi$ and $\neg\neg\psi \supset \psi$ give $\Gamma \vdash \psi$. $\square$

1.1.4.2. Lemma. *Let the formula A involve only the propositional variables among $p_1, \ldots, p_n$. For $1 \le i \le n$, suppose that B_i is either p_i or $\neg p_i$. Then, either*

$$B_1, \ldots, B_n \vdash A \qquad \text{or} \qquad B_1, \ldots, B_n \vdash \neg A.$$

Proof. Define τ to be a truth assignment that makes each B_i true. By the soundness theorem, A (respectively, $\neg A$), can be proved from the hypotheses $B_1, \ldots, B_n$ only if $\overline{\tau}(A) = T$ (respectively $\overline{\tau}(A) = F$). Lemma 1.1.4.2 asserts that the converse holds too.

The lemma is proved by induction on the complexity of A. In the base case, A is just p_i: this case is trivial to prove since B_i is either p_i or $\neg p_i$. Now suppose A is a formula $A_1 \vee A_2$. If $\sigma(A) = T$, then we must have $\tau(A_i) = T$ for some $i \in \{1, 2\}$; the induction hypothesis implies that $B_1, \ldots, B_n \vdash A_i$ and thus, by (i) above, $B_1, \ldots, B_n \vdash A$. On the other hand, if $\tau(A) = F$, then $\tau(A_1) = \tau(A_2) = F$, so the induction hypothesis implies that $B_1, \ldots, B_n \vdash \neg A_i$ for both $i = 1$ and $i = 2$. From this, (j) implies that $B_1, \ldots, B_n \vdash \neg A$. The cases where A has outermost connective $\wedge$, $\supset$ or $\neg$ are proved similarly. $\square$.

We are now ready to complete the proof of the Completeness Theorem 1.1.3. Suppose A is a tautology. We claim that Lemma 1.1.4.2 can be strengthened to have

$$B_1, \ldots, B_k \vdash A$$

where, as before each B_i is either p_i or $\neg p_i$, but now $0 \le k \le n$ is permitted. We prove this by induction on $k = n, n-1, \ldots, 1, 0$. For $k = n$, this is just Lemma 1.1.4.2. For the induction step, note that $B_1, \ldots, B_k \vdash A$ follows from $B_1, \ldots, B_k, p_{k+1} \vdash A$ and $B_1, \ldots, B_k, \neg p_{k+1} \vdash A$ by Lemma 1.1.4.1. When $k = 0$, we have that $\vdash A$, which proves the Completeness Theorem.
Q.E.D. Theorem 1.1.3

1.1.5. It still remains to prove the compactness theorem for propositional logic. This theorem states:

Compactness Theorem. *Let Γ be a set of propositional formulas.*
(1) *Γ is satisfiable if and only if every finite subset of Γ is satisfiable.*
(2) *$\Gamma \vDash A$ if and only if there is a finite subset Γ_0 of Γ such that $\Gamma_0 \vDash A$.*

Since $\Gamma \vDash A$ is equivalent to $\Gamma \cup \{\neg A\}$ being unsatisfiable, (2) is implied by (1). It is fairly easy to prove the compactness theorem directly, and most introductory books in mathematical logic present such a proof. Here, we shall instead, give a proof based on the Tychonoff theorem; obviously this connection to topology is the reason for the name 'compactness theorem.'

Proof. Let V be the set of propositional variables used in Γ; the sets Γ and V need not necessarily be countable. Let 2^V denote the set of truth assignments on V and endow 2^V with the product topology by viewing it as the product of $|V|$ copies of the two element space with the discrete topology. That is to say, the subbasis elements of 2^V are the sets $B_{p,i} = \{\tau : \tau(p) = i\}$ for $p \in V$ and $i \in \{T, F\}$. Note that these subbasis elements are both open and closed. Recall that the Tychonoff theorem states that an arbitrary product of compact spaces is compact; in particular, 2^V is compact. (See Munkres [1975] for background material on topology.)

For $\phi \in \Gamma$, define $D_\phi = \{\tau \in 2^V : \tau \vDash \phi\}$. Since ϕ only involves finitely many variables, each D_ϕ is both open and closed. Now Γ is satisfiable if and only if $\cap_{\phi\in\Gamma} D_\phi$ is non-empty. By the compactness of 2^V, the latter condition is equivalent to the sets $\cap_{\phi\in\Gamma_0} D_\phi$ being non-empty for all finite $\Gamma_0 \subset \Gamma$. This, in turn is equivalent to each finite subset Γ_0 of Γ being satisfiable. □

The compactness theorem for first-order logic is more difficult; a purely model-theoretic proof can be given with ultrafilters (see, e.g., Eklof [1977]). We include a proof-theoretic proof of the compactness theorem for first-order logic for countable languages in section 2.3.7 below.

1.1.6. Remarks. There are of course a large number of possible ways to give sound and complete proof systems for propositional logic. The particular proof system $\mathcal{F}$ used above is adapted from Kleene [1952]. A more detailed proof of the completeness theorem for $\mathcal{F}$ and for related systems can be found in the textbook of Mendelson [1987]. The system $\mathcal{F}$ is an example of a class of proof systems called *Frege proof systems*: a Frege proof system is any proof system in which all axioms and rules are schematic and which is implicationally sound and implicationally complete. Most of the commonly used proof systems similar to $\mathcal{F}$ are based on modus ponens as the only rule of inference; however, some (non-Frege) systems also incorporate a version of the deduction theorem as a rule of inference. In these systems, if B has been inferred from A, then the formula $A \supset B$ may also be inferred. An example of such a system is the propositional fragment of the natural deduction proof system described in section 2.4.8 below.

Other rules of inference that are commonly allowed in propositional proof systems include the *substitution rule* which allows any instance of ϕ to be inferred from ϕ, and the *extension rule* which permits the introduction of abbreviations for long formulas. These two systems *appear* to be more powerful than Frege systems in that they seem to allow substantially shorter proofs of certain tautologies. However, whether they actually are significantly more powerful than Frege systems is an open problem. This issues are discussed more fully by Pudlák in Chapter VIII.

There are several currently active areas of research in the proof theory of propositional logic. Of course, the central open problem is the P versus NP question of whether there exists a polynomial time method of recognizing tautologies. Research on the proof theory of propositional logic can be, roughly speaking, separated into three problem areas. Firstly, the problem of "proof-search" is the question of what are the best algorithmic methods for searching for propositional proofs. The proof-search problem is important for artificial intelligence, for automated theorem proving and for logic programming. The most common propositional proof systems used for proof-search algorithms are variations of the resolution system discussed in 1.3 below. A second, related research area is the question of proof lengths. In this area, the central questions concern the minimum lengths of proofs needed for tautologies in particular proof systems. This topic is treated in more depth in Chapter VIII in this volume.

A third research area concerns the investigation of fragments of the propositional proof system $\mathcal{F}$. For example, propositional intuitionist logic is the logic which is axiomatized by the system $\mathcal{F}$ without the axiom scheme $\neg\neg A \supset A$. Another important example is linear logic. Brief discussions of these two logics can be found in section 3.

1.2. The propositional sequent calculus

The sequent calculus, first introduced by Gentzen [1935] as an extension of his earlier natural deduction proof systems, is arguably the most elegant and flexible system for writing proofs. In this section, the propositional sequent calculus for classical logic is developed; the extension to first-order logic is treated in 2.3 below.

1.2.1. Sequents and Cedents. In the Hilbert-style systems, each line in a proof is a formula; however, in sequent calculus proofs, each line in a proof is a *sequent*: a sequent is written in the form

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

where the symbol $\longrightarrow$ is a new symbol called the sequent arrow (not to be confused with the implication symbol $\supset$) and where each A_i and B_j is a formula. The intuitive meaning of the sequent is that the conjunction of the A_i's implies the disjunction of the B_j's. Thus, a sequent is equivalent in meaning to the formula

$$\bigwedge_{i=1}^{k} A_i \supset \bigvee_{j=1}^{\ell} B_j.$$

The symbols $\bigwedge$ and $\bigvee$ represent conjunctions and disjunctions, respectively, of multiple formulas. We adopt the convention that an empty conjunction (say, when $k = 0$ above) has value "True", and that an empty disjunction (say, when $\ell = 0$ above) has value "False". Thus the sequent $\longrightarrow A$ has the same meaning as the formula A, and the *empty sequent* $\longrightarrow$ is false. A sequent is defined to be valid or a tautology if and only if its corresponding formula is.

The sequence of formulas $A_1, \ldots, A_k$ is called the *antecedent* of the sequent displayed above; $B_1, \ldots, B_\ell$ is called its *succedent*. They are both referred to as *cedents*.

1.2.2. Inferences and proofs. We now define the propositional sequent calculus proof system PK. A sequent calculus proof consists of a rooted tree (or sometimes a directed acyclic graph) in which the nodes are sequents. The root of the tree, written at the bottom, is called the *endsequent* and is the sequent proved by the proof. The leaves, at the top of the tree, are called *initial sequents* or *axioms*. Usually, the only initial sequents allowed are the logical axioms of the form $A \longrightarrow A$, where we further require that A be atomic.

Other than the initial sequents, each sequent in a PK-proof must be inferred by one of the rules of inference given below. A rule of inference is denoted by a figure $\frac{S_1}{S}$ or $\frac{S_1 \quad S_2}{S}$ indicating that the sequent S may be inferred from S_1 or from the pair S_1 and S_2. The conclusion, S, is called the *lower sequent* of the inference; each hypotheses is an *upper sequent* of the inference. The valid rules of inference for PK are as follows; they are essentially schematic, in that A and B denote arbitrary formulas and Γ, Δ, etc. denote arbitrary cedents.

Weak Structural Rules

$$\textit{Exchange:left}\ \frac{\Gamma, A, B, \Pi \longrightarrow \Delta}{\Gamma, B, A, \Pi \longrightarrow \Delta} \qquad \textit{Exchange:right}\ \frac{\Gamma \longrightarrow \Delta, A, B, \Lambda}{\Gamma \longrightarrow \Delta, B, A, \Lambda}$$

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

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

The weak structural rules are also referred to as just *weak* inference rules. The rest of the rules are called *strong* inference rules. The *structural* rules consist of the weak structural rules and the cut rule.

The Cut Rule

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

The Propositional Rules[1]

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

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

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

$$\supset\textit{:left}\ \frac{\Gamma \longrightarrow \Delta, A \qquad B, \Gamma \longrightarrow \Delta}{A \supset B, \Gamma \longrightarrow \Delta} \qquad \supset\textit{:right}\ \frac{A, \Gamma \longrightarrow \Delta, B}{\Gamma \longrightarrow \Delta, A \supset B}$$

[1] We have stated the $\wedge$*:left* and the $\vee$*:right* rules differently than the traditional form. The traditional definitions use the following two $\vee$*:right* rules of inference

The above completes the definition of PK. We write $PK \vdash \Gamma \longrightarrow \Delta$ to denote that the sequent $\Gamma \longrightarrow \Delta$ has a PK-proof. When A is a formula, we write $PK \vdash A$ to mean that $PK \vdash \longrightarrow A$.

The cut rule plays a special role in the sequent calculus, since, as is shown in section 1.2.8, the system PK is complete even without the cut rule; however, the use of the cut rule can significantly shorten proofs. A proof is said to be *cut-free* if does not contain any cut inferences.

1.2.3. Ancestors, descendents and the subformula property. All of the inferences of PK, with the exception of the cut rule, have a *principal formula* which is, by definition, the formula occurring in the lower sequent of the inference which is not in the cedents Γ or Δ (or Π or Λ). The exchange inferences have two principal formulas. Every inference, except weakenings, has one or more *auxiliary formulas* which are the formulas A and B, occurring in the upper sequent(s) of the inference. The formulas which occur in the cedents Γ, Δ, Π or Λ are called *side formulas* of the inference. The two auxiliary formulas of a cut inference are called the *cut formulas*.

We now define the notions of descendents and ancestors of formulas occurring in a sequent calculus proof. First we define *immediate descendents* as follows: If C is a side formula in an upper sequent of an inference, say C is the i-th subformula of a cedent Γ, Π, Δ or Λ, then C's only immediate descendent is the corresponding occurrence of the same formula in the same position in the same cedent in the lower sequent of the inference. If C is an auxiliary formula of any inference except an exchange or cut inference, then the principal formula of the inference is the immediate descendent of C. For an exchange inference, the immediate descendent of the A or B in the upper sequent is the A or B, respectively, in the lower sequent. The cut formulas of a cut inference do not have immediate descendents. We say that C is an *immediate ancestor* of D if and only if D is an immediate descendent of C. Note that the only formulas in a proof that do not have immediate ancestors are the formulas in initial sequents and the principal formulas of weakening inferences.

The *ancestor* relation is defined to be the reflexive, transitive closure of the immediate ancestor relation; thus, C is an ancestor of D if and only if there is a chain of zero or more immediate ancestors from D to C. A *direct ancestor* of D is an ancestor C of D such that C is the same formula as D. The concepts of *descendent* and *direct descendent* are defined similarly as the converses of the ancestor and direct ancestor relations.

A simple, but important, observation is that if C is an ancestor of D, then C is a subformula of D. This immediately gives the following subformula property:

$$\frac{\Gamma \longrightarrow \Delta, A}{\Gamma \longrightarrow \Delta, A \lor B} \qquad \text{and} \qquad \frac{\Gamma \longrightarrow \Delta, A}{\Gamma \longrightarrow \Delta, B \lor A}$$

and two dual rules of inference for $\land$*:left*. Our method has the advantage of reducing the number of rules of inference, and also simplifying somewhat the upper bounds on cut-free proof length we obtain below.

1.2.4. Proposition. *(The Subformula Property) If P is a cut-free PK-proof, then every formula occurring in P is a subformula of a formula in the endsequent of P.*

1.2.5. Lengths of proofs. There are a number of ways to measure the length of a sequent calculus proof P; most notably, one can measure either the number of symbols or the number of sequents occurring in P. Furthermore, one can require P to be tree-like or to be dag-like; in the case of dag-like proofs no sequent needs to be derived, or counted, twice. ('Dag' abbreviates 'directed acyclic graph', another name for such proofs is 'sequence-like'.)

For this chapter, we adopt the following conventions for measuring lengths of sequent calculus proofs: proofs are always presumed to be tree-like, unless we explicitly state otherwise, and we let $||P||$ denote the number of **strong** inferences in a tree-like proof P. The value $||P||$ is polynomially related to the number of sequents in P. If P has n sequents, then, of course, $||P|| < n$. On the other hand, it is not hard to prove that for any tree-like proof P of a sequent $\Gamma \longrightarrow \Delta$, there is a (still tree-like) proof of an endsequent $\Gamma' \longrightarrow \Delta'$ with at most $||P||^2$ sequents and with $\Gamma' \subseteq \Gamma$ and $\Delta' \subseteq \Delta$. The reason we use $||P||$ instead of merely counting the actual number of sequents in P, is that using $||P||$ often makes bounds on proof size significantly more elegant to state and prove.

Occasionally, we use $||P||_{dag}$ to denote the number of strong inferences in a dag-like proof P.

1.2.6. Soundness Theorem. *The propositional sequent calculus PK is sound. That is to say, any PK-provable sequent or formula is a tautology.*

The soundness theorem is proved by observing that the rules of inference of *PK* preserve the property of sequents being tautologies.

The implicational form of the soundness theorem also holds. If $\mathfrak{S}$ is a set of sequents, let an $\mathfrak{S}$-proof be any sequent calculus proof in which sequents from $\mathfrak{S}$ are permitted as initial sequents (in addition to the logical axioms). The implicational soundness theorem states that if a sequent $\Gamma \longrightarrow \Delta$ has an $\mathfrak{S}$-proof, then $\Gamma \longrightarrow \Delta$ is made true by every truth assignment which satisfies $\mathfrak{S}$.

1.2.7. The inversion theorem. The inversion theorem is a kind of inverse to the implicational soundness theorem, since it says that, for any inference except weakening inferences, if the conclusion of the inference is valid, then so are all of its hypotheses.

Theorem. *Let I be a propositional inference, a cut inference, an exchange inference or a contraction inference. If I's lower sequent is valid, then so are are all of I's upper sequents. Likewise, if I's lower sequent is true under a truth assignment τ, then so are are all of I's upper sequents.*

The inversion theorem is easily proved by checking the eight propositional inference rules; it is obvious for exchange and contraction inferences.

Note that the inversion theorem can fail for weakening inferences. Most authors define the $\land$*:left* and $\lor$*:right* rules of inference differently than we defined them for PK, and the inversion theorem can fail for these alternative formulations (see the footnote on page 11).

1.2.8. The completeness theorem. The completeness theorem for PK states that every valid sequent (tautology) can be proved in the propositional sequent calculus. This, together with the soundness theorem, shows that the PK-provable sequents are precisely the valid sequents.

Theorem. *If* $\Gamma \longrightarrow \Delta$ *is a tautology, then it has a PK-proof in which no cuts appear.*

1.2.9. In order to prove Theorem 1.2.8 we prove the following stronger lemma which includes bounds on the size of the PK-proof.

Lemma. *Let* $\Gamma \longrightarrow \Delta$ *be a valid sequent in which there are* m *occurrences of logical connectives. Then there is a tree-like, cut free PK-proof* P *of* $\Gamma \longrightarrow \Delta$ *containing fewer than* 2^m *strong inferences.*

Proof. The proof is by induction on m. In the base case, $m = 0$, the sequent $\Gamma \longrightarrow \Delta$ contains no logical connectives and thus every formula in the sequent is a propositional variable. Since the sequent is valid, there must be some variable, p, which occurs both in Γ and in Δ. Thus $\Gamma \longrightarrow \Delta$ can be proved with zero strong inferences from the initial sequent $p \longrightarrow p$.

The induction step, $m > 0$, is handled by cases according to which connectives are used as outermost connectives of formulas in the cedents Γ and Δ. First suppose there is a formula of the form $(\neg A)$ in Γ. Letting Γ' be the cedent obtained from Γ by removing occurrences of $\neg A$, we can infer $\Gamma \longrightarrow \Delta$ by:

$$\cfrac{\cfrac{\Gamma' \longrightarrow \Delta, A}{\neg A, \Gamma' \longrightarrow \Delta}}{\Gamma \longrightarrow \Delta}$$

where the double line indicates a series of weak inferences. By the inversion theorem, $\Gamma' \longrightarrow \Delta, A$ is valid, and hence, since it has at most $m-1$ logical connectives, the induction hypothesis implies that it has a cut-free proof with fewer than 2^{m-1} strong inferences. This gives $\Gamma \longrightarrow \Delta$ a cut-free proof with fewer than $2^{m-1}+1 \leq 2^m$ strong inferences. The case where a formula $(\neg A)$ occurs in Δ is handled similarly.

Second, consider the case where a formula of the form $A \land B$ appears in Γ. Then, letting Γ' be the cedent Γ minus the formula $A \land B$, we can infer $\Gamma \longrightarrow \Delta$ by:

$$\cfrac{\cfrac{A, B, \Gamma' \longrightarrow \Delta}{A \land B, \Gamma' \longrightarrow \Delta}}{\Gamma \longrightarrow \Delta}$$

By the inversion theorem and the induction hypothesis, $A, B, \Gamma' \longrightarrow \Delta$ has a cut-free proof with fewer than 2^{m-1} strong inferences. Thus $\Gamma \longrightarrow \Delta$ has a cut-free proof with fewer than 2^m strong inferences. Third, suppose there is a formula of the $A \land B$

appearing in the succedent Δ. Letting Δ' be the the succedent Δ minus the formula $A \wedge B$, we can infer

$$\cfrac{\cfrac{\Gamma \longrightarrow \Delta', A \qquad \Gamma \longrightarrow \Delta', B}{\Gamma \longrightarrow \Delta', A \wedge B}}{\Gamma \longrightarrow \Delta}$$

By the inversion theorem, both of upper sequents above are valid. Furthermore, they each have fewer than m logical connectives, so by the induction hypothesis, they have cut-free proofs with fewer than 2^{m-1} strong inferences. This gives the sequent $\Gamma \longrightarrow \Delta$ a cut-free proof with fewer than 2^m strong inferences.

The remaining cases are when a formula in the sequent $\Gamma \longrightarrow \Delta$ has outermost connective $\vee$ or $\supset$. These are handled with the inversion theorem and the induction hypothesis similarly to the above cases. $\square$

1.2.10. The bounds on the proof size in Lemma 1.2.9 can be improved somewhat by counting only the occurrences of distinct subformulas in $\Gamma \longrightarrow \Delta$. To make this precise, we need to define the concepts of positively and negatively occurring subformulas. Given a formula A, an occurrence of a subformula B of A, and a occurrence of a logical connective α in A, we say that B is negatively bound by α if either (1) α is a negation sign, $\neg$, and B is in its scope, or (2) α is an implication sign, $\supset$, and B is a subformula of its first argument. Then, B is said to occur *negatively* (respectively, *positively*) in A if B is negatively bound by an odd (respectively, even) number of connectives in A. A subformula occurring in a sequent $\Gamma \longrightarrow \Delta$ is said to be positively occurring if it occurs positively in Δ or negatively in Γ; otherwise, it occurs negatively in the sequent.

Lemma. *Let $\Gamma \longrightarrow \Delta$ be a valid sequent. Let m' equal the number of distinct subformulas occurring positively in the sequent and m'' equal the number of distinct subformulas occurring negatively in the sequent. Let $m = m' + m''$. Then there is a tree-like, cut free PK-proof P containing fewer than 2^m strong inferences.*

Proof. (Sketch) Recall that the proof of Lemma 1.2.9 built a proof from the bottom-up, by choosing a formula in the endsequent to eliminate (i.e., to be inferred) and thereby reducing the total number of logical connectives and then appealing to the induction hypothesis. The construction for the proof of the present lemma is exactly the same, except that now care must be taken to reduce the total number of distinct positively or negatively occurring subformulas, instead of just reducing the total number of connectives. This is easily accomplished by always choosing a formula from the endsequent which contains a maximal number of connectives and which is therefore not a proper subformula of any other subformula in the endsequent. $\square$

1.2.11. The cut elimination theorem states that if a sequent has a *PK*-proof, then it has a cut-free proof. This is an immediate consequence of the soundness and completeness theorems, since any *PK*-provable sequent must be valid, by the soundness theorem, and hence has a cut-free proof, by the completeness theorem.

This is a rather slick method of proving the cut elimination theorem, but unfortunately, does not shed any light on how a given PK-proof can be constructively transformed into a cut-free proof. In section 2.3.7 below, we shall give a step-by-step procedure for converting first-order sequent calculus proofs into cut-free proofs; the same methods work also for propositional sequent calculus proofs. We shall not, however, describe this constructive proof transformation procedure here; instead, we will only state, without proof, the following upper bound on the increase in proof length which can occur when a proof is transformed into a cut-free proof. (A proof can be given using the methods of Sections 2.4.2 and 2.4.3.)

Cut-Elimination Theorem. *Suppose P be a (possibly dag-like) PK-proof of $\Gamma \longrightarrow \Delta$. Then $\Gamma \longrightarrow \Delta$ has a cut-free, tree-like PK-proof with less than or equal to $2^{||P||_{dag}}$ strong inferences.*

1.2.12. Free-cut elimination. Let $\mathfrak{S}$ be a set of sequents and, as above, define an $\mathfrak{S}$-proof to be a sequent calculus proof which may contain sequents from $\mathfrak{S}$ as initial sequents, in addition to the logical sequents. If I is a cut inference occurring in an $\mathfrak{S}$-proof P, then we say I's cut formulas are *directly descended from* $\mathfrak{S}$ if they have at least one direct ancestor which occurs as a formula in an initial sequent which is in $\mathfrak{S}$. A cut I is said to be *free* if neither of I's auxiliary formulas is directly descended from $\mathfrak{S}$. A proof is *free-cut free* if and only if it contains no free cuts. (See Definition 2.4.4.1 for a better definition of free cuts.) The cut elimination theorem can be generalized to show that the free-cut free fragment of the sequent calculus is implicationally complete:

Free-cut Elimination Theorem. *Let S be a sequent and $\mathfrak{S}$ a set of sequents. If $\mathfrak{S} \vDash \mathfrak{S}$, then there is a free-cut free $\mathfrak{S}$-proof of S.*

We shall not prove this theorem here; instead, we prove the generalization of this for first-order logic in 2.4.4 below. This theorem is essentially due to Takeuti [1987] based on the cut elimination method of Gentzen [1935].

1.2.13. Some remarks. We have developed the sequent calculus only for classical propositional logic; however, one of the advantages of the sequent calculus is its flexibility in being adapted for non-classical logics. For instance, propositional intuitionistic logic can be formalized by a sequent calculus PJ which is defined exactly like PK except that succedents in the lower sequents of *strong* inferences are restricted to contain at most one formula. As another example, minimal logic is formalized like PJ, except with the restriction that every succedent contain exactly one formula. Linear logic, relevant logic, modal logics and others can also be formulated elegantly with the sequent calculus.

1.2.14. The Tait calculus. Tait [1968] gave a proof system similar in spirit to the sequent calculus. Tait's proof system incorporates a number of simplifications with regard to the sequent calculus; namely, it uses sets of formulas in place of sequents, it allows only propositional formulas to be negated, and there are no weak structural rules at all.

Since the Tait calculus is often used for analyzing fragments of arithmetic, especially in the framework of infinitary logic, we briefly describe it here. Formulas are built up from atomic formulas p, from negated atomic formulas $\neg p$, and with the connectives $\land$ and $\lor$. A negated atomic formula is usually denoted $\overline{p}$; and the negation operator is extended to all formulas by defining $\overline{\overline{p}}$ to be just p, and inductively defining $\overline{A \lor B}$ and $\overline{A \land B}$ to be $\overline{A} \land \overline{B}$ and $\overline{A} \lor \overline{B}$, respectively.

Frequently, the Tait calculus is used for infinitary logic. In this case, formulas are defined so that whenever Γ is a set of formulas, then so are $\bigvee \Gamma$ and $\bigwedge \Gamma$. The intended meaning of these formulas is the disjunction, or conjunction, respectively, of all the formulas in Γ.

Each line in a Tait calculus proof is a set Γ of formulas with the intended meaning of Γ being the disjunction of the formulas in Γ. A Tait calculus proof can be tree-like or dag-like. The initial sets, or logical axioms, of a proof are sets of the form $\Gamma \cup \{p, \overline{p}\}$. In the infinitary setting, there are three rules of inference; namely,

$$\frac{\Gamma \cup \{A_j\}}{\Gamma \cup \{\bigvee_{i \in I} A_i\}} \qquad \text{where } j \in I,$$

$$\frac{\Gamma \cup \{A_j : j \in I\}}{\Gamma \cup \{\bigwedge_{j \in I} A_j\}} \qquad \text{(there are } |I| \text{ many hypotheses), and}$$

$$\frac{\Gamma \cup \{A\} \qquad \Gamma \cup \{\overline{A}\}}{\Gamma} \qquad \text{the cut rule.}$$

In the finitary setting, the same rules of inference may also be used. It is evident that the Tait calculus is practically isomorphic to the sequent calculus. This is because a sequent $\Gamma \longrightarrow \Delta$ may be transformed into the equivalent set of formulas containing the formulas from Δ and the negations of the formulas from Γ. The exchange and contraction rules are superfluous once one works with sets of formulas, the weakening rule of the sequent calculus is replaced by allowing axioms to contain extra side formulas (this, in essence, means that weakenings are pushed up to the initial sequents of the proof). The strong rules of inference for the sequent calculus translate, by this means, to the rules of the Tait calculus.

Recall that we adopted the convention that the length of a sequent calculus proof is equal to the number of strong inferences in the proof. When we work with tree-like proofs, this corresponds exactly to the number of inferences in the corresponding Tait-style proof.

The cut elimination theorem for the (finitary) sequent calculus immediately implies the cut elimination theorem for the Tait calculus for finitary logic; this is commonly called the *normalization theorem* for Tait-style systems. For general infinitary logics, the cut elimination/normalization theorems may not hold; however, Lopez-Escobar [1965] has shown that the cut elimination theorem does hold for infinitary logic with formulas of countably infinite length. Also, Chapters III and IV of this volume discuss cut elimination in some infinitary logics corresponding to theories of arithmetic.

1.3. Propositional resolution refutations

The Hilbert-style and sequent calculus proof systems described earlier are quite powerful; however, they have the disadvantage that it has so far proved to be very difficult to implement computerized procedures to search for propositional Hilbert-style or sequent calculus proofs. Typically, a computerized procedure for proof search will start with a formula A for which a proof is desired, and will then construct possible proofs of A by working backwards from the conclusion A towards initial axioms. When cut-free proofs are being constructed this is fairly straightforward, but cut-free proofs may be much longer than necessary and may even be too long to be feasibly constructed by a computer. General, non-cut-free, proofs may be quite short; however, the difficulty with proof search arises from the need to determine what formulas make suitable cut formulas. For example, when trying to construct a proof of $\Gamma \longrightarrow \Delta$ that ends with a cut inference; one has to consider *all* formulas C and try to construct proofs of the sequents $\Gamma \longrightarrow \Delta, C$ and $C, \Gamma \longrightarrow \Delta$. In practice, it has been impossible to choose cut formulas C effectively enough to effectively generate general proofs. A similar difficulty arises in trying to construct Hilbert-style proofs which must end with a modus ponens inference.

Thus to have a propositional proof system which would be amenable to computerized proof search, it is desirable to have a proof system in which (1) proof search is efficient and does not require too many 'arbitrary' choices, and (2) proof lengths are not excessively long. Of course, the latter requirement is intended to reflect the amount of available computer memory and time; thus proofs of many millions of steps might well be acceptable. Indeed, for computerized proof search, having an easy-to-find proof which is millions of steps long may well be preferable to having a hard-to-find proof which has only hundreds of steps.

The principal propositional proof system which meets the above requirements is based on resolution. As we shall see, the expressive power and implicational power of resolution is weaker than that of the full propositional logic; in particular, resolution is, in essence, restricted to formulas in conjunctive normal form. However, resolution has the advantage of being amenable to efficient proof search.

Propositional and first-order resolution were introduced in the influential work of Robinson [1965b] and Davis and Putnam [1960]. Propositional resolution proof systems are discussed immediately below. A large part of the importance of propositional resolution lies in the fact that it leads to efficient proof methods in first-order logic: first-order resolution is discussed in section 2.6 below.

1.3.1. Definition. A *literal* is defined to be either a propositional variable p_i or the negation of a propositional variable $\neg p_i$. The literal $\neg p_i$ is also denoted $\overline{p_i}$; and if x is the literal $\overline{p_i}$, then $\overline{x}$ denotes the unnegated literal p_i. The literal $\overline{x}$ is called the *complement* of x. A *positive* literal is one which is an unnegated variable; a *negative* literal is one which is a negated variable.

A *clause* C is a finite set of literals. The intended meaning of C is the disjunction of its members; thus, for σ a truth assignment, $\sigma(C)$ equals *True* if and only if $\sigma(x) = \mathit{True}$ for some $x \in C$. Note that the empty clause, $\emptyset$, always has value *False*. Since clauses that contain both x and $\overline{x}$ are always true, it is often assumed w.l.o.g. that no clause contains both x and $\overline{x}$. A clause is defined to be *positive* (respectively, *negative*) if it contains only positive (resp., only negative) literals. The empty clause is the only clause which is both positive and negative. A clause which is neither positive nor negative is said to be *mixed*.

A non-empty set Γ of clauses is used to represent the conjunction of its members. Thus $\sigma(\Gamma)$ is *True* if and only if $\sigma(C)$ is *True* for all $C \in \Gamma$. Obviously, the meaning of Γ is the same as the conjunctive normal form formula consisting of the conjunction of the disjunctions of the clauses in Γ. A set of clauses is said to be *satisfiable* if there is at least one truth assignment that makes it true.

Resolution proofs are used to prove that a set Γ of clauses is unsatisfiable: this is done by using the resolution rule (defined below) to derive the empty clause from Γ. Since the empty clause is unsatisfiable this will be sufficient to show that Γ is unsatisfiable.

1.3.2. Definition. Suppose that C and D are clauses and that $x \in C$ and $\overline{x} \in D$ are literals. The *resolution rule* applied to C and D is the inference

$$\frac{C \qquad\qquad D}{(C \setminus \{x\}) \cup (D \setminus \{\overline{x}\})}$$

The conclusion $(C \setminus \{x\}) \cup (D \setminus \{\overline{x}\})$ is called the *resolvent* of C and D (with respect to x).[2]

Since the resolvent of C and D is satisfied by any truth assignment that satisfies both C and D, the resolution rule is *sound* in the following sense: if Γ is satisfiable and B is the resolvent of two clauses in Γ, then $\Gamma \cup \{B\}$ is satisfiable. Since the empty clause is not satisfiable, this yields the following definition of the resolution proof system.

Definition. A *resolution refutation* of Γ is a sequence $C_1, C_2, \ldots, C_k$ of clauses such that each C_i is either in Γ or is inferred from earlier member of the sequence by the resolution rule, and such that C_k is the empty clause.

1.3.3. Resolution is defined to be a refutation procedure which refutes the satisfiability of a set of clauses, but it also functions as a proof procedure for proving the validity of propositional formulas; namely, to prove a formula A, one forms a set Γ_A of clauses such that A is a tautology if and only if Γ_A is unsatisfiable. Then

[2]Note that x is uniquely determined by C and D if we adopt the (optional) convention that clauses never contain any literal and its complement.

a resolution proof of A is, by definition, a resolution refutation of Γ_A. There are two principal means of forming Γ_A from A. The first method is to express the negation of A in conjunctive normal form and let Γ_A be the set of clauses which express that conjunctive normal form formula. The principal drawback of this definition of Γ_A is that the conjunctive normal form of $\neg A$ may be exponentially longer than A, and Γ_A may have exponentially many clauses.

The second method is the method of *extension*, introduced by Tsejtin [1968], which involves introducing new propositional variables in addition to the variables occurring in the formula A. The new propositional variables are called *extension variables* and allow each distinct subformula B in A to be assigned a literal x_B by the following definition: (1) for propositional variables appearing in A, x_p is p; (2) for negated subformulas $\neg B$, $x_{\neg B}$ is $\overline{x_B}$; (3) for any other subformula B, x_B is a new propositional variable. We then define Γ_A to contain the following clauses: (1) the singleton clause $\{\overline{x_A}\}$, (2) for each subformula of the form $B \wedge C$ in A, the clauses

$$\{\overline{x_{B\wedge C}}, x_B\}, \quad \{\overline{x_{B\wedge C}}, x_C\} \quad \text{and} \quad \{\overline{x_B}, \overline{x_C}, x_{B\wedge C}\};$$

and (3) for each subformula of the form $B \vee C$ in A, the clauses

$$\{x_{B\vee C}, \overline{x_B}\}, \quad \{x_{B\vee C}, \overline{x_C}\} \quad \text{and} \quad \{x_B, x_C, \overline{x_{B\vee C}}\}.$$

If there are additional logical connectives in A then similar sets of clauses can be used. It is easy to check that the set of three clauses for $B \wedge C$ (respectively, $B \vee C$) is satisfied by exactly those truth assignments that assign $x_{B\wedge C}$ (respectively, $x_{B\vee C}$) the same truth value as $x_B \wedge x_C$ (resp., $x_B \vee x_C$). Thus, a truth assignment satisfies Γ_A only if it falsifies A and, conversely, if a truth assignment falsifies A then there is a unique assignment of truth values to the extension variables of A which lets it satisfy Γ_A. The primary advantage of forming Γ_A by the method of extension is that Γ_A is linear-size in the size of A. The disadvantage, of course, is that the introduction of additional variables changes the meaning and structure of A.

1.3.4. Completeness Theorem for Propositional Resolution. *If Γ is an unsatisfiable set of clauses, then there is a resolution refutation of Γ.*

Proof. We shall briefly sketch two proofs of this theorem. The first, and usual, proof is based on the Davis-Putnam procedure. First note, that by the Compactness Theorem we may assume w.l.o.g. that Γ is finite. Therefore, we may use induction on the number of distinct variables appearing in Γ. The base case, where no variables appear in Γ is trivial, since Γ must contain just the empty clause. For the induction step, let p be a fixed variable in Γ and define Γ' to be the set of clauses containing the following clauses:

a. For all clauses C and D in Γ such that $p \in C$ and $\overline{p} \in D$, the resolvent of C and D w.r.t. p is in Γ', and

b. Every clause C in Γ which contains neither p nor $\overline{p}$ is in Γ'.

Assuming, without loss of generality, that no clause in Γ contained both p and $\overline{p}$, it is clear that the variable p does not occur in Γ'. Now, it is not hard to show that Γ' is satisfiable if and only if Γ is, from whence the theorem follows by the induction hypothesis.

The second proof reduces resolution to the free-cut free sequent calculus. For this, if C is a clause, let Δ_C be the cedent containing the variables which occur positively in C and Π_C be the variables which occur negatively in C. Then the sequent $\Pi_C \longrightarrow \Delta_C$ is a sequent with no non-logical symbols which is identical in meaning to C. For example, if $C = \{p_1, \overline{p_2}, p_3\}$, then the associated sequent is $p_2 \longrightarrow p_1, p_3$. Clearly, if C and D are clauses with a resolvent E, then the sequent $\Pi_E \longrightarrow \Delta_E$ is obtained from $\Pi_C \longrightarrow \Delta_C$ and $\Pi_D \longrightarrow \Delta_D$ with a single cut on the resolution variable. Now suppose Γ is unsatisfiable. By the completeness theorem for the free-cut free sequent calculus, there is a free-cut free proof of the empty sequent from the sequents $\Pi_C \longrightarrow \Delta_C$ with $C \in \Gamma$. Since the proof is free-cut free and there are no non-logical symbols appearing in any initial sequents, every cut formula in the proof must be atomic. Therefore no non-logical symbol appear anywhere in the proof and, by identifying the sequents in the free-cut free proof with clauses and replacing each cut inference with the corresponding resolution inference, a resolution refutation of the empty clause is obtained. □

1.3.5. Restricted forms of resolution. One of the principal advantages of resolution is that it is easier for computers to search for resolution refutations than to search for arbitrary Hilbert-style or sequent calculus proofs. The reason for this is that resolution proofs are less powerful and more restricted than Hilbert-style and sequent calculus proofs and, in particular, there are fewer options on how to form resolution proofs. This explains the paradoxical situation that a less-powerful proof system can be preferable to a more powerful system. Thus it makes sense to consider further restrictions on resolution which may reduce the proof search space even more. Of course there is a tradeoff involved in using more restricted forms of resolution, since one may find that although restricted proofs are easier to search for, they are a lot less plentiful. Often, however, the ease of proof search is more important than the existence of short proofs; in fact, it is sometimes even preferable to use a proof system which is not complete, provided its proofs are easy to find.

Although we do not discuss this until section 2.6, the second main advantage of resolution is that propositional refutations can be 'lifted' to first-order refutations of first-order formulas. It is important that the restricted forms of resolution discussed next also apply to first-order resolution refutations.

One example of a restricted form of resolution is implicit in the first proof of the Completeness Theorem 1.3.4 based on the Davis-Putnam procedure; namely, for any ordering of the variables $p_1, \ldots, p_m$, it can be required that a resolution refutation has first resolutions with respect to p_1, then resolutions with respect to p_2, etc., concluding with resolutions with respect to p_m. This particular strategy is not particularly useful since it does not reduce the search space sufficiently. We consider next several strategies that have been somewhat more useful.

1.3.5.1. Subsumption. A clause C is said to *subsume* a clause D if and only if $C \subseteq D$. The subsumption principle states that if two clauses C and D have been derived such that C subsumes D, then D should be discarded and not used further in the refutation. The subsumption principle is supported by the following theorem:

Theorem. *If Γ is unsatisfiable and if $C \subset D$, then $\Gamma' = (\Gamma \setminus \{D\}) \cup \{C\}$ is also unsatisfiable. Furthermore, Γ' has a resolution refutation which is no longer than the shortest refutation of Γ.*

1.3.5.2. Positive resolution and hyperresolution. Robinson [1965a] introduced positive resolution and hyperresolution. A *positive* resolution inference is one in which one of the hypotheses is a positive clause. The completeness of positive resolution is shown by:

Theorem. (Robinson [1965a]) *If Γ is unsatisfiable, then Γ has a refutation containing only positive resolution inferences.*

Proof. (Sketch) It will suffice to show that it is impossible for an unsatisfiable Γ to be closed under positive resolution inferences and not contain the empty clause. Let A be the set of positive clauses in Γ; A must be non-empty since Γ is unsatisfiable. Pick a truth assignment τ that satisfies all clauses in A and assigns the minimum possible number of "true" values. Pick a clause L in $\Gamma \setminus A$ which is falsified by τ and has the minimum number of negative literals; and let $\overline{p}$ be one of the negative literals in L. Note that L exists since Γ is unsatisfiable and that $\tau(p) = \mathit{True}$. Pick a clause $J \in A$ that contains p and has the rest of its members assigned false by τ; such a clause J exists by the choice of τ. Considering the resolvent of J and L, we obtain a contradiction. $\square$

Positive resolution is nice in that it restricts the kinds of resolution refutations that need to be attempted; however, it is particularly important as the basis for *hyperresolution*. The basic idea behind hyperresolution is that multiple positive resolution inferences can be combined into a single inference with a positive conclusion. To justify hyperresolution, note that if R is a positive resolution refutation then the inferences in R can be uniquely partitioned into subproofs of the form

$$
\dfrac{A_n \qquad \dfrac{\vdots}{B_n}}{A_{n+1}}
\qquad
\begin{array}{c}
\dfrac{A_3 \qquad \dfrac{A_2 \qquad \dfrac{A_1 \quad B_1}{B_2}}{B_3}}{B_4} \\
\cdot^{\cdot^{\cdot}}
\end{array}
$$

where each of the clauses $A_1, \ldots, A_{n+1}$ are positive (and hence the clauses $B_1, \ldots, B_n$ are not positive). These $n+1$ positive resolution inferences are combined into the single *hyperresolution* inference

$$\frac{A_1 \quad A_2 \quad A_3 \quad \cdots \quad A_n \quad B_1}{A_{n+1}}$$

(This construction is the definition of hyperresolution inferences.)

It follows immediately from the above theorem that hyperresolution is complete. The importance of hyperresolution lies in the fact that one can search for refutations containing only positive resolutions and that as clauses are derived, only the positive clauses need to be saved for possible future use as hypotheses.

Negative resolution is defined similarly to positive resolution and is likewise complete.

1.3.5.3. Semantic resolution. Semantic resolution, independently introduced by Slagle [1967] and Luckham [1970], can be viewed as a generalization of positive resolution. For semantic resolution, one uses a fixed truth assignment (interpretation) τ to restrict the permissible resolution inferences. A resolution inference is said to be τ-supported if one of its hypotheses is given value *False* by τ. Note that at most one hypothesis can have value *False*, since the hypotheses contain complementary occurrences of the resolvent variable.

A resolution refutation is said to be τ-supported if each of its resolution inferences are τ-supported. If τ_F is the truth assignment which assigns every variable the value *False*, then a τ_F-supported resolution refutation is definitionally the same as a positive resolution refutation. Conversely, if Γ is a set of clauses and if τ is any truth assignment, then one can form a set Γ' by complementing every variable in Γ which has τ-value *True*: clearly, a τ-supported resolution refutation of Γ is isomorphic to a positive resolution refutation of Γ'. Thus, Theorem 1.3.5.2 is equivalent to the following Completeness Theorem for semantic resolution:

Theorem. *For any τ and Γ, Γ is unsatisfiable if and only if Γ has a τ-supported resolution refutation.*

It is possible to define semantic-hyperresolution in terms of semantic resolution, just as hyperresolution was defined in terms of positive resolution.

1.3.5.4. Set-of-support resolution. Wos, Robinson and Carson [1965] introduced *set-of-support* resolution as another principle for guiding a search for resolution refutations. Formally, set of support is defined as follows: if Γ is a set of clauses and if $\Pi \subset \Gamma$ and $\Gamma \setminus \Pi$ is satisfiable, then Π is a set of support for Γ; a refutation R of Γ is said to be *supported by* Π if every inference in R is derived (possibly indirectly) from at least one clause in Π. (An alternative, almost equivalent, definition would be to require that no two members of $\Gamma \setminus \Pi$ are resolved together.) The intuitive idea behind set of support resolution is that when trying to refute Γ, one should concentrate on trying to derive a contradiction from the part Π of Γ which is not known to be consistent. For example, $\Gamma \setminus \Pi$ might be a database of facts which is presumed to be consistent, and Π a clause which we are trying to refute.

Theorem. *If Γ is unsatisfiable and Π is a set of support for Γ, then Γ has a refutation supported by Π.*

This theorem is immediate from Theorem 1.3.5.3. Let τ be any truth assignment which satisfies $\Gamma \setminus \Pi$, then a τ-supported refutation is also supported by Π.

The main advantage of set of support resolution over semantic resolution is that it does not require knowing or using a satisfying assignment for $\Gamma \setminus \Pi$.

1.3.5.5. Unit and input resolution. A *unit* clause is defined to be a clause containing a single literal; a *unit resolution inference* is an inference in at least one of the hypotheses is a unit clause. As a general rule, it is desirable to perform unit resolutions whenever possible. If Γ contains a unit clause $\{x\}$, then by combining unit resolutions with the subsumption principle, one can remove from Γ every clause which contains x and also every occurrence of $\overline{x}$ from the rest of the clauses in Γ. (The situation is a little more difficult when working in first-order logic, however.) This completely eliminates the literal x and reduces the number of and sizes of clauses to consider.

A unit resolution refutation is a refutation which contains only unit resolutions. Unfortunately, unit resolution is not complete: for example, an unsatisfiable set Γ with no unit clauses cannot have a unit resolution refutation.

An *input* resolution refutation of Γ is defined to be a refutation of Γ in which every resolution inference has at least one of its hypotheses in Γ. Obviously, a minimal length input refutation will be tree-like. Input resolution is also not complete; in fact, it can refute exactly the same sets as unit resolution:

Theorem. (Chang [1970]) *A set of clauses has a unit refutation if and only if it has a input refutation.*

1.3.5.6. Linear resolution. Linear resolution is a generalization of input resolution which has the advantage of being complete: a *linear resolution refutation* of Γ is a refutation $A_1, A_2, \ldots, A_{n-1}, A_n = \emptyset$ such that each A_i is either in Γ or is obtained by resolution from A_{i-1} and A_j for some $j < i-1$. Thus a linear refutation has the same linear structure as an input resolution, but is allowed to reuse intermediate clauses which are not in Γ.

Theorem. (Loveland [1970] and Luckham [1970]) *If Γ is unsatisfiable, then Γ has a linear resolution refutation.*

Linear and input resolution both lend themselves well to depth-first proof search strategies. Linear resolution is still complete when used in conjunction with set-of-support resolution.

Further reading. We have only covered some of the basic strategies for proposition resolution proof search. The original paper of Robinson [1965b] still provides an excellent introduction to resolution; this and many other foundational papers on this topic have been reprinted in Siekmann and Wrightson [1983]. In addition, the textbooks by Chang and Lee [1973], Loveland [1978], and Wos et al. [1992] give a more complete description of various forms of resolution than we have given above.

Horn clauses. A *Horn clause* is a clause which contains at most one positive literal. Thus a Horn clause must be of the form $\{p, \overline{q_1}, \ldots, \overline{q_n}\}$ or of the form $\{\overline{q_1}, \ldots, \overline{q_n}\}$ with $n \geq 0$. If a Horn clause is rewritten as sequents of atomic variables, it will have at most one variable in the antecedent; typically, Horn clauses are written in reverse-sequent format so, for example, the two Horn clauses above would be written as implications

$$p \Leftarrow q_1, \ldots, q_n$$

and

$$\Leftarrow q_1, \ldots, q_n.$$

In this reverse-sequent notation, the antecedent is written after the $\Leftarrow$, and the commas are interpreted as conjunctions ($\land$'s). Horn clauses are of particular interest both because they are expressive enough to handle many situations and because deciding the satisfiability of sets of Horn clauses is more feasible than deciding the satisfiability of arbitrary sets of clauses. For these reasons, many logic programming environments such as PROLOG are based partly on Horn clause logic.

In propositional logic, it is an easy matter to decide the satisfiability of a set of Horn clauses; the most straightforward method is to restrict oneself to positive unit resolution. A *positive unit* inference is a resolution inference in which one of the hypotheses is a unit clause containing a positive literal only. A positive unit refutation is a refutation containing only positive unit resolution inferences.

Theorem. *A set of Horn clauses is unsatisfiable if and only if it has a positive unit resolution refutation.*

Proof. Let Γ be an unsatisfiable set of Horn clauses. Γ must contain at least one positive unit clause $\{p\}$, since otherwise the truth assignment that assigned *False* to all variables would satisfy Γ. By resolving $\{p\}$ against all clauses containing $\overline{p}$, and then discarding all clauses which contain p or $\overline{p}$, one obtains a smaller unsatisfiable set of Horn clauses. Iterating this yields the desired positive unit refutation. □

Positive unit resolutions are quite adequate in propositional logic, however, they do not lift well to applications in first-order logic and logic programming. For this, a more useful method of search for refutations is based on combining semantic resolution, linear resolution and set-of-support resolution:

1.3.5.7. Theorem. Henschen and Wos [1974]. *Suppose Γ is an unsatisfiable set of Horn clauses with $\Pi \subseteq \Gamma$ a set of support for Γ, and suppose that every clause in $\Gamma \setminus \Pi$ contains a positive literal. Then Γ has a refutation which is simultaneously a negative resolution refutation and a linear refutation and which is supported by Π.*

Note that the condition that every clause in $\Gamma \setminus \Pi$ contains a positive literal means that the truth assignment τ that assigns *True* to every variable satisfies $\Gamma \setminus \Pi$. Thus a negative resolution refutation is the same as a τ-supported refutation and hence is supported by Π.

The theorem is fairly straightforward to prove, and we leave the details to the reader. However, note that since every clause in $\Gamma \setminus \Pi$ is presumed to contain a positive literal, it is impossible to get rid of all positive literals only by resolving against clauses in $\Gamma \setminus \Pi$. Therefore, Π must contain a negative clause C such that there is a linear derivation that begins with C, always resolves against clauses in $\Gamma \setminus \Pi$ yielding negative clauses only, and ending with the empty clause. The resolution refutations of Theorem 1.3.5.7, or rather the lifting of these to Horn clauses described in section 2.6.5, can be combined with restrictions on the order in which literals are resolved to give what is commonly called SLD-resolution.

2. Proof theory of first-order logic

2.1. Syntax and semantics

2.1.1. Syntax of first-order logic. First-order logic is a substantial extension of propositional logic, and allows reasoning about individuals using functions and predicates that act on individuals. The symbols allowed in first-order formulas include the propositional connectives, quantifiers, variables, function symbols, constant symbols and relation symbols. We take $\neg$, $\land$, $\lor$ and $\supset$ as the allowed propositional connectives. There is an infinite set of variable symbols; we use $x, y, z, \ldots$ and $a, b, c, \ldots$ as metasymbols for variables. The quantifiers are the existential quantifiers, $(\exists x)$, and the universal quantifiers, $(\forall x)$, which mean "there exists x" and "for all x". A given first-order language contains a set of function symbols of specified arities, denoted by metasymbols $f, g, h, \ldots$ and a set of relation symbols of specified arities, denoted by metasymbols $P, Q, R, \ldots$. Function symbols of arity zero are called *constant symbols*. Sometimes the first-order language contains a distinguished two-place relation symbol $=$ for equality.

The formulas of first-order logic are defined as follows. Firstly, *terms* are built up from function symbols, constant symbols and variables. Thus, any variable x is a term, and if $t_1, \ldots, t_k$ are terms and f is k-ary, then $f(t_1, \ldots, t_k)$ is a term. Second, *atomic formulas* are defined to be of the form $P(t_1, \ldots, t_k)$ for P a k-ary relation symbol. Finally, *formulas* are inductively to be built up from atomic formulas and logical connectives; namely, any atomic formula is a formula, and if A and B are formulas, then so are $(\neg A)$, $(A \land B)$, $(A \lor B)$, $(A \supset B)$, $(\forall x)A$ and $(\exists x)A$. To avoid writing too many parentheses, we adopt the conventions on omitting parentheses in propositional logic with the additional convention that quantifiers bind more tightly than binary propositional connectives. In addition, binary predicate symbols, such as $=$ and $<$, are frequently written in infix notation.

Consider the formula $(x = 0 \lor (\forall x)(x \neq f(x)))$. (We use $x \neq f(x)$ to abbreviate $(\neg x = f(x))$.) This formula uses the variable x in two different ways: on one hand, it asserts something about an object x; and on the other hand, it (re)uses the variable x to state a general property about all objects. Obviously it would be less confusing to write instead $(x = 0 \lor (\forall y)(y \neq f(y)))$; however, there is nothing wrong, formally speaking, with using x in two ways in the same formula. To keep track of this, we need to define free and bound occurrences of variables. An *occurrence* of a variable x in a formula A is defined to be any place that the symbol x occurs in A, except in quantifer symbols (Qx). (We write (Qx) to mean either $(\forall x)$ or $(\exists x)$.) If $(Qx)(\cdots)$ is a subformula of A, then the *scope* of this occurrence of (Qx) is defined to be the subformula denoted $(\cdots)$. An occurrence of x in A is said to be *bound* if and only if it is in the scope of a quantifier (Qx); otherwise the occurrence of x is called *free.* If x is a bound occurrence, it is *bound by* the rightmost quantifier (Qx) which it is in the scope of. A formula in which no variables appear freely is called a *sentence.*

The intuitive idea of free occurrences of variables in A is that A says something about the free variables. If t is a term, we define the *substitution of t for x in A*, denoted $A(t/x)$ to be the formula obtained from A by replacing each free occurrence of x in A by the term t. To avoid unwanted effects, we generally want t to be *freely substitutable* for x, which means that no free variable in t becomes bound in A as a result of this substitution; formally defined, this means that no free occurrence of x in A occurs in the scope of a quantifier (Qy) with y a variable occurring in t. The *simultaneous substitution of $t_1, \ldots, t_k$ for $x_1, \ldots, x_k$ in A*, denoted $A(t_1/x_1, \ldots, t_k/x_k)$, is defined similarly in the obvious way.

To simplify notation, we adopt some conventions for denoting substitution. Firstly, if we write $A(x)$ and $A(t)$ in the same context, this indicates that $A = A(x)$ is a formula, and that $A(t)$ is $A(t/x)$. Secondly, if we write $A(s)$ and $A(t)$ in the same context, this is to mean that A is formula, x is some variable, and that $A(s)$ is $A(s/x)$ and $A(t)$ is $A(t/x)$.

The sequent calculus, discussed in 2.3, has different conventions on variables than the Hilbert-style systems discussed in 2.2; most notably, it has distinct classes of symbols for free variables and bound variables. See section 2.3.1 for the discussion of the usage of variables in the sequent calculus.

2.1.2. Semantics of first-order logic. In this section, we define the semantics, or 'meaning', of first-order formulas. Since this is really model theory instead of proof theory, and since the semantics of first-order logic is well-covered in any introductory textbook on mathematical logic, we give only a very concise description of the notation and conventions used in this chapter.

In order to ascribe a truth value to a formula, it is necessary to give an interpretation of the non-logical symbols appearing in it; namely, we must specify a *domain* or *universe* of objects, and we must assign meanings to each variable occurring freely and to each function symbol and relation symbol appearing in the formula. A *structure* $\mathcal{M}$ (also called an *interpretation*), for a given language L, consists of the following:

(1) A non-empty universe M of objects, intended to be the universe of objects over which variables and terms range;

(2) For each k-ary function f of the language, an interpretation $f^{\mathcal{M}} : M^k \mapsto M$; and

(3) For each k-ary relation symbol P of the language, an interpretation $P^{\mathcal{M}} \subseteq M^k$ containing all k-tuples for which P is intended to hold. If the first-order language contains the symbol for equality, then $=^{\mathcal{M}}$ must be the true equality predicate on M.

We shall next define the true/false value of a sentence in a structure. This is possible since the structure specifies the meaning of all the function symbols and relation symbols in the formula, and the quantifiers and propositional connectives take their usual meanings. For A a sentence and $\mathcal{M}$ a structure, we write $\mathcal{M} \vDash \mathcal{A}$ to denote A being true in the structure $\mathcal{M}$. In this case, we say that $\mathcal{M}$ is a *model* of A, or that A is *satisfied* by $\mathcal{M}$. Often, we wish to talk about the meaning of a formula A in which free variables occur. For this, we need not only a structure, but also an *object assignment*, which is a mapping σ from the set of variables (at least the ones free in A) to the universe M. The object assignment σ gives meanings to the freely occurring variables, and it is straightforward to define the property of A being true in a given structure $\mathcal{M}$ under a given object assignment σ, denoted $\mathcal{M} \vDash A[\sigma]$.

To give the formal definition of $\mathcal{M} \vDash A[\sigma]$, we first need to define the interpretation of terms, i.e, we need to formally define the manner in which arbitrary terms represent objects in the universe M. To this end, we define $t^{\mathcal{M}}[\sigma]$ by induction on the complexity of t. For x a variable, $x^{\mathcal{M}}[\sigma]$ is just $\sigma(x)$. For t a term of the form $f(t_1, \ldots, t_k)$, we define $t^{\mathcal{M}}[\sigma]$ to equal $f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_k^{\mathcal{M}}[\sigma])$. If t is a *closed term*, i.e., contains no variables, then $t^{\mathcal{M}}[\sigma]$ is independent of σ and is denoted by just $t^{\mathcal{M}}$.

If σ is an object assignment and $m \in M$, then $\sigma(m/x)$ is the object assignment which is identical to σ except that it maps x to m. We are now ready to define the truth of A in $\mathcal{M}$ with respect to σ, by induction on the complexity of A. For A an atomic formula $P(t_1, \ldots, t_k)$, then $\mathcal{M} \vDash A[\sigma]$ holds if and only if the k-tuple $(t_1^{\mathcal{M}}[\sigma], \ldots, t_k^{\mathcal{M}}[\sigma])$ is in $P^{\mathcal{M}}$. Also, $\mathcal{M} \vDash \neg A[\sigma]$ holds if and only $\mathcal{M} \nvDash A[\sigma]$, where $\nvDash$ is the negation of $\vDash$. Likewise, the value of $\mathcal{M} \vDash A \odot B[\sigma]$ with $\odot$ one of the binary propositional connectives depends only on the truth values of $\mathcal{M} \vDash A[\sigma]$ and $\mathcal{M} \vDash B[\sigma]$, in the obvious way according to Table 1 above. If A is $(Qx)B$ with Q denoting either $\exists$ or $\forall$, then $\mathcal{M} \vDash A[\sigma]$ holds if and only if the property $\mathcal{M} \vDash B[\sigma(m/x)]$ holds for some (respectively, for all) $m \in M$.

We say that a formula A is *valid in* $\mathcal{M}$, if $\mathcal{M} \vDash A[\sigma]$ holds for all appropriate object assignments. A formula is defined to be *valid* if and only if it is valid in all structures. If Γ is a set of formulas, we say Γ is valid in $\mathcal{M}$, written $\mathcal{M} \vDash \Gamma$, if and only if every formula in Γ is valid in M. When $\mathcal{M} \vDash \Gamma$, we say that Γ is *satisfied* by $\mathcal{M}$. The set Γ is *satisfiable* if and only if it is satisfied by some model.

For Γ a set of formulas and A a formula, we define Γ *logically implies* A, $\Gamma \vDash A$, to hold if A is valid in every structure in which Γ is valid. If Γ is the empty set, we write just $\vDash A$, which means of course that A is valid.[3]

[3]This definition of $\Gamma \vDash A$ has the slightly unexpected result that free variables appearing in Γ are treated as if they are universally quantified. For example, according to our definition, we have

2.1.3. For languages which contain the equality symbol, $=$, the interpretation of $=^{\mathcal{M}}$ in any structure $\mathcal{M}$ must be true equality. Of course, this restriction influences the definition of $\Gamma \vDash A$ in languages which contain equality. Let $\vDash'$ indicate logical implication defined with respect to all structures, including structures in which equality is not interpreted by true equality. It is an elementary, but important, fact that $\vDash$ can be defined in terms of $\vDash'$. Namely, let EQ be the set of equality axioms defined in section 2.2.1 below; then $\Gamma \vDash A$ holds if and only if $\Gamma, EQ \vDash' A$ holds.

2.1.4. Definition. A first-order *theory* is a set T of sentences closed under logical implication; i.e., if $T \vDash A$ then $A \in T$. An *axiomatization* of T is a set Γ of sentences such that T is precisely the set of sentences logically implied by Γ.

2.2. Hilbert-style proof systems

2.2.1. A proof system. We give here an example of a proof system $\mathcal{F}_{FO}$ for first-order logic, which is an extension of $\mathcal{F}$ to first-order logic. In addition to the axioms of $\mathcal{F}$ and the rule modus ponens, there are two axiom schemes:

$$A(t) \supset (\exists x)A(x) \qquad \text{and} \qquad (\forall x)A(x) \supset A(t)$$

where A may be any formula and t any term. There are also two quantifier rules of inference:

$$\frac{C \supset A(x)}{C \supset (\forall x)A(x)} \qquad \text{and} \qquad \frac{A(x) \supset C}{(\exists x)A(x) \supset C}$$

where, in both inferences, x may not appear freely in C.

If the first-order language under consideration contains the distinguished equality sign $(=)$, then the *equality axioms* must also be included; namely,

$$(\forall x)(x = x)$$
$$(\forall \vec{x})(\forall \vec{y})(x_1 = y_1 \land \cdots \land x_k = y_k \supset f(\vec{x}) = f(\vec{y}))$$
$$(\forall \vec{x})(\forall \vec{y})(x_1 = y_1 \land \cdots \land x_k = y_k \land P(\vec{x}) \supset P(\vec{y}))$$

where f and P are arbitrary function and predicate symbols and k is their arity.

We leave it to the reader to check that the symmetry and transitivity of equality follow from these axioms, since in the third equality axiom, P may be the equality sign. We write $\mathcal{F}_{\rm FO} \vdash A$ to denote A having an $\mathcal{F}_{\rm FO}$-proof.

$A(x) \vDash (\forall x)A(x)$, and thus $A(x) \vDash A(y)$. An alternative definition of logical implication is often used (but not in this chapter!) in which free variables occurring in Γ are treated as syntactically as constants; under this definition $A(x) \vDash (\forall x)A(x)$ does not hold in general.

The advantage our choice for the definition of $\vDash$, is that it yields a simpler and more elegant proof-theory. This choice does not involve any loss of expressive power since, instead of free variables in Γ, one can use new constant symbols, and thus obtain the same effect as the alternative definition of $\vDash$. In any event, the two definitions of $\vDash$ coincide when Γ is a set of sentences.

2.2.2. The Soundness Theorem.

(1) *If* $\mathcal{F}_{\mathrm{FO}} \vdash A$*, then* $\vDash A$*.*

(2) *Let* Γ *be a set of sentences. If there is an* $\mathcal{F}_{\mathrm{FO}}$*-proof of* A *using sentences from* Γ *as additional axioms, then* $\Gamma \vDash A$*.*

The Soundness Theorem states that $\mathcal{F}_{\mathrm{FO}}$ proves only valid formulas. Both parts of the soundness theorem are readily proved by induction on the number of lines in a proof.

2.2.3. The Completeness Theorem.

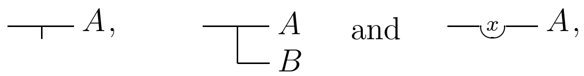
(1) *If* $\vDash A$*, then* $\mathcal{F}_{\mathrm{FO}} \vdash A$*.*

(2) *Let* Γ *be a set of formulas. If* $\Gamma \vDash A$*, then there is an* $\mathcal{F}_{\mathrm{FO}}$*-proof of* A *using sentences from* Γ *as additional axioms.*

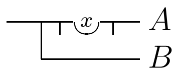
The completeness theorem and soundness theorem together show that $\mathcal{F}_{\mathrm{FO}}$ is an adequate system for formalizing first-order logic. The completeness theorem was proved originally by Gödel [1930]; the now-standard textbook proof is due to Henkin [1949]. We shall not give a proof for the completeness of $\mathcal{F}_{\mathrm{FO}}$ here; instead, we shall prove the completeness of the cut-free fragment of LK in 2.3.7 below. It is straightforward to see that $\mathcal{F}_{\mathrm{FO}}$ can simulate the LK proof system, and this gives an indirect proof of the completeness of $\mathcal{F}_{\mathrm{FO}}$.

2.2.4. A set Γ of sentences is *consistent* if and only if there is no sentence A such that both A and $\neg A$ are provable from Γ and, equivalently, if and only if there is some formula A such that there is no proof of A from Γ. The soundness and completeness theorems immediately imply that Γ is consistent if and only if Γ is satisfiable.

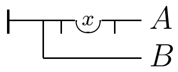
2.2.5. Historical remarks. Frege [1879] gave the first full formulation of first-order logic. Frege used a pictorial representation of propositional connectives and quantifiers; one remnant of his notation that is still in use is "$\vdash A$", which was Frege's notation for "A is asserted to be true". Frege used the pictorial notation $—\!\!— A$ to express a proposition A; whereas he used the notation $\vdash\!\!—\!\!— A$ to express the assertion that the proposition A is true (or has been proved). Negation, implication, and universal quantification were represented by Frege with pictures such as



which represent the propositions $\neg A$, $B \supset A$, and $(\forall x)A$, respectively. These constructions can be iterated to form arbitrary first-order expression; that is to say, in the pictures above, A and B may be replaced by arbitrary pictorial propositions. The addition of a vertical line to the left end of a proposition indicates that the proposition has been established to be true (e.g., proved). Thus, for instance



and



denote the proposition $B \supset (\exists x)A$ and the assertion that this proposition has been established, respectively.

Frege also developed extensions to his first-order logic to include functionals and predicates; however, these extensions were later discovered by Russell to introduce set-theoretic paradoxes.

Peano [1889] introduced, independently of Frege, a fragment of first-order logic for reasoning about integers. Based on the work of Frege and Peano, Whitehead and Russell [1910] gave a consistent framework for formalizing mathematics based on first-order logic, using universal and existential quantifiers denoted (x) and $(\exists x)$.

The 'Hilbert-style' system given above is apparently so-named because it is closely related to a system used by Hilbert and Ackermann [1928], which is based on earlier lectures of Hilbert. The later work of Hilbert and Bernays [1934-39], however, used the ϵ-calculus instead of a 'Hilbert-style' system. The ϵ-calculus contains no quantifiers, but in their place uses symbols $\epsilon_x(A(x))$ which are intended to denote an object x such that $A(x)$ holds, if there is such an object. Note that other free variables and other uses of ϵ symbols may appear in $A(x)$. The ϵ-symbol can be used to express quantifiers by the informal equivalences $(\exists x)A(x) \Leftrightarrow A(\epsilon_x(A(x)))$ and $(\forall x)A(x) \Leftrightarrow A(\epsilon_x(\neg A(x)))$.

The Hilbert-style system we used above is essentially that of Kleene [1952].

2.3. The first-order sequent calculus

In this section, the propositional sequent calculus, introduced in section 1.2, is enlarged to a proof system for first-order logic.

2.3.1. Free and bound variables. The first-order sequent calculus has two classes of variables, called *free variables* and *bound variables*. There are infinitely many variables of each type; free variables are denoted by the metavariables $a, b, c, \ldots$, and bound variables by the metavariables $z, y, x, \ldots$. The essential idea is that free variables may not be quantified, while bound variables may not occur freely in formulas. The syntactic distinction between free and bound variables necessitates a change to the definitions of terms and formulas. Firstly, *terms* are now defined as being built up from free variables and function symbols; whereas, the *semiterms* are defined as being built up from free and bound variables and function symbols. Secondly, only bound variables may ever be quantified. The set of *formulas* is now redefined with the additional requirement that only free variables may occur freely in formulas. *Semiformulas* are like formulas, except that bound variables may occur freely in semiformulas. We henceforth use $r, s, t, \ldots$ as metavariables for terms, and $A, B, C, \ldots$ as metavariables for formulas.

Note that in general, a subformula of a formula will be a semiformula instead of a formula.

One advantage of these conventions on free and bound variables, is that it avoids some of the difficulties involved in defining $A(t)$, since it is always the case that the term t will be freely substitutable for b in a formula $A(b)$. Also, without these conventions, the cut elimination theorem proved below would have to be reformulated slightly. For example, it is not hard to see that $P(x,y) \longrightarrow (\exists y)(\exists x)P(y,x)$ would not have a cut-free proof (this example is from Feferman [1968]).

2.3.2. Definition. The first-order sequent calculus LK is defined as an extension of the propositional system PK. LK contains all the rules of inference of PK plus the following additional rules of inference:

The Quantifier Rules

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

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

In quantifier rules, A may be an arbitrary formula, t an arbitrary term, and the free variable b of the $\forall$*:right* and $\exists$*:left* inferences is called the *eigenvariable* of the inference and must not appear in Γ, Δ. The propositional rules and the quantifier rules are collectively called *logical rules.*

Most of the syntactic definitions of PK carry over to LK. For example, the notions of (direct) descendents and ancestors are identically defined; and proof lengths are still measured in terms of the number of strong inferences in the proof. The quantifier inferences are, of course, considered strong rules of inferences.

If S is a sequent $\Gamma \longrightarrow \Delta$, then we let A_S be the formula $(\bigwedge \Gamma) \supset (\bigvee \Delta)$. Taking S to have the same meaning as A_S, all the definitions of 'validity' and 'logical implication' of section 2.1.2 apply also to sequents. Let the free variables of A_S be $\vec{b}$, so $A_S = A_S(\vec{b})$. We let $\forall S$ denote the *universal closure*, $(\forall \vec{x})A_S(\vec{x})$, of the formula A_S.

2.3.3. LK is defined to allow only initial sequents of the form $A \longrightarrow A$ with A atomic. However, it is often convenient to allow other initial sequents; so if $\mathfrak{S}$ is a set of sequents, we define $LK_{\mathfrak{S}}$ to be the proof system defined like LK, but allowing initial sequents to be from $\mathfrak{S}$ too.

An important example is the theory LK_e for first-order logic with equality. LK_e is LK with the addition of the following initial sequents for equality:

$$\longrightarrow s = s$$

$$s_1 = t_1, \ldots, s_k = t_k \longrightarrow f(\vec{s}) = f(\vec{t})$$

$$s_1 = t_1, \ldots, s_k = t_k, P(\vec{s}) \longrightarrow P(\vec{t})$$

We say that a set $\mathfrak{S}$ is *closed under substitution*, if whenever $\Gamma(a) \longrightarrow \Delta(a)$ is in $\mathfrak{S}$ and t is a term, then $\Gamma(t) \longrightarrow \Delta(t)$ is also in $\mathfrak{S}$.

2.3.4. When P is a proof, we write $P(a)$ and $P(t)$ to indicate that $P(t)$ is the result of replacing every free occurrence of a in formulas of P with t.

Theorem. *Let $\mathfrak{S}$ be a set of sequents which is closed under substitution. If $P(b)$ is an $LK_{\mathfrak{S}}$-proof, and if neither b nor any variable in t is used as an eigenvariable in $P(b)$, then $P(t)$ is a valid $LK_{\mathfrak{S}}$-proof.*

2.3.5. Definition. A free variable in the endsequent of a proof is called a *parameter variable* of the proof. A proof P is said to be *in free variable normal form* provided that (1) no parameter variable is used as an eigenvariable, and (2) every other free variable appearing in P is used exactly once as an eigenvariable and appears in P only in sequents above the inference for which it is used as an eigenvariable.

In this chapter, we consider only tree-like proofs and thus any proof may be put in free variable normal form by merely renaming variables.

2.3.6. Soundness Theorem. *Let $\Gamma \longrightarrow \Delta$ be an arbitrary sequent.*

(1) *If $\Gamma \longrightarrow \Delta$ has an LK-proof, then $\Gamma \longrightarrow \Delta$ is valid.*

(2) *Let $\mathfrak{S}$ be a set of sequents. If $\Gamma \longrightarrow \Delta$ has an $LK_{\mathfrak{S}}$-proof, then $\mathfrak{S} \vDash \Gamma \longrightarrow \Delta$.*

The soundness theorem is readily proved by induction on the number of inferences in a proof.

2.3.7. Completeness of cut-free LK. We next prove the completeness of the cut-free fragment of LK and, more generally, the completeness of $LK_{\mathfrak{S}}$. Since our proofs of completeness also give a proof of the (countable) compactness theorem, we include the compactness theorem as part (2) of the completeness theorem.

Cut-free Completeness Theorem. *Let $\Gamma \longrightarrow \Delta$ be a sequent in a first-order language L which does not contain equality.*

(1) *If $\Gamma \longrightarrow \Delta$ is valid, then it has a cut-free LK-proof.*

(2) *Let Π be a set of sentences. If Π logically implies $\Gamma \longrightarrow \Delta$, then there are $C_1, \ldots, C_k \in \Pi$ such that $C_1, \ldots, C_k, \Gamma \longrightarrow \Delta$ has a cut-free LK-proof.*

Corollary. *Let $\mathfrak{S}$ be a set of sequents. If $\mathfrak{S}$ logically implies $\Gamma \longrightarrow \Delta$, then $\Gamma \longrightarrow \Delta$ has an $LK_{\mathfrak{S}}$-proof.*

Note that in the corollary, the $LK_{\mathfrak{S}}$-proof may not be cut-free. An important special case of the corollary is that LK_e is complete. Although, in general, the cut-free fragment of $LK_{\mathfrak{S}}$ may not be complete; we shall see later that it is always possible to get $LK_{\mathfrak{S}}$ proofs which contain no 'free cuts'.

The corollary is an immediate consequence of the cut-free completeness theorem. This is because, if $\mathfrak{S} \vDash \Gamma \longrightarrow \Delta$, then part (2) of the theorem implies that there are $S_1, \ldots, S_k \in \mathfrak{S}$ so that $\forall S_1, \ldots, \forall S_k, \Gamma \longrightarrow \Delta$ has an LK-proof. But clearly each $\longrightarrow \forall S_i$ has an $LK_{\mathfrak{S}}$-proof, so with k further cut inferences, $\Gamma \longrightarrow \Delta$ also has an $LK_{\mathfrak{S}}$-proof.

Proof. Part (2) of the cut-free completeness theorem clearly implies part (1); we shall prove (2) only for the case where Π is countable (and hence the language L is, w.l.o.g., countable). Assume Π logically implies $\Gamma \longrightarrow \Delta$. The general idea of the argument is to try to build an LK-proof of $\Gamma \longrightarrow \Delta$ from the bottom up, working backwards from $\Gamma \longrightarrow \Delta$ to initial sequents. This is, of course, analogous to the procedure used to prove Theorem 1.2.8; but because of the presence of quantifiers, the process of searching for a proof of $\Gamma \longrightarrow \Delta$ is no longer finite. Thus, we shall have to show the proof-search process eventually terminates — this will be done by showing that if the proof-search does not yield a proof, then $\Gamma \longrightarrow \Delta$ is not valid.

Since the language is countable, we may enumerate all L-formulas as $A_1, A_2, A_3, \ldots$ so that every L-formula occurs infinitely often in the enumeration. Likewise, we enumerate all L-terms as $t_1, t_2, t_3, \ldots$ with each term repeated infinitely often. We shall attempt to construct a cut-free proof P of $\Gamma \longrightarrow \Delta$. The construction of P will proceed in stages: initially, P consists of just the sequent $\Gamma \longrightarrow \Delta$; at each stage, P will be modified (we keep the same name P for the new partially constructed P). A sequent in P is said to be *active* provided it is a leaf sequent of P and there is no formula that occurs in both its antecedent and its succedent. Note that a non-active leaf sequent is derivable with a cut-free proof.

We enumerate all pairs $\langle A_i, t_j \rangle$ by a diagonal enumeration; each stage of the construction of P considers one such pair. Initially, P is the single sequent $\Gamma \longrightarrow \Delta$. At each stage we do the following:

Loop: Let $\langle A_i, t_j \rangle$ be the next pair in the enumeration.

Step (1): If A_i is in Π, then replace every sequent $\Gamma' \longrightarrow \Delta'$ in P with the sequent $\Gamma', A_i \longrightarrow \Delta'$.

Step (2): If A_i is atomic, do nothing and proceed to the next stage. Otherwise, we will modify P at the active sequents which contain A_i by doing one of the following:

Case (2a): If A_i is $\neg B$, then every active sequent in P which contains A_i, say of form $\Gamma', \neg B, \Gamma'' \longrightarrow \Delta'$, is replaced by the derivation

$$\frac{\Gamma', \neg B, \Gamma'' \longrightarrow \Delta', B}{\overline{\Gamma', \neg B, \Gamma'' \longrightarrow \Delta'}}$$

and similarly, every active sequent in P of the form $\Gamma' \longrightarrow \Delta', \neg B, \Delta''$ is replaced by the derivation

$$\frac{B, \Gamma' \longrightarrow \Delta', \neg B, \Delta''}{\overline{\Gamma' \longrightarrow \Delta', \neg B, \Delta''}}$$

Case (2b): If A_i is of the form $B \lor C$, then every active sequent in P of the form $\Gamma', B \lor C, \Gamma'' \longrightarrow \Delta'$, is replaced by the derivation

$$\frac{B, \Gamma', B \lor C, \Gamma'' \longrightarrow \Delta' \qquad C, \Gamma', B \lor C, \Gamma'' \longrightarrow \Delta'}{\Gamma', B \lor C, \Gamma'' \longrightarrow \Delta'}$$

And, every active sequent in P of the form $\Gamma' \longrightarrow \Delta', B \lor C, \Delta''$ is replaced by the derivation

$$\frac{\Gamma' \longrightarrow \Delta', B \lor C, \Delta'', B, C}{\Gamma' \longrightarrow \Delta', B \lor C, \Delta''}$$

Cases (2c)-(2d): The cases where A_i has outermost connective $\supset$ or $\land$ are similar (dual) to case (2b), and are omitted here.

Case (2e): If A_i is of the form $(\exists x)B(x)$, then every active sequent in P of the form $\Gamma', (\exists x)B(x), \Gamma'' \longrightarrow \Delta'$ is replaced by the derivation

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

where c is a new free variable, not used in P yet. In addition, any active sequent of the form $\Gamma' \longrightarrow \Delta', (\exists x)B(x), \Delta''$ is replaced by the derivation

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

Note that this, and the dual $\forall$*:left* case, are the only cases where t_j is used. These two cases are also the only two cases where it is really necessary to keep the formula A_i in the new active sequent; we have however, always kept A_i since it makes our arguments a little simpler.

Case (2f): The case where A_i begins with a universal quantifier is dual to case (2e).

Step (3): If there are no active sequents remaining in P, exit from the loop; otherwise, continue with the next loop iteration.

End loop.

If the algorithm constructing P ever halts, then P gives a cut-free proof of $C_1, \ldots, C_k, \Gamma \longrightarrow \Delta$ for some $C_1, \ldots, C_k \in \Pi$. This is since each (non-active) initial sequent has a formula A which appears in both its antecedent and succedent and since, by induction on the complexity of A, every sequent $A \longrightarrow A$ is derivable with a cut-free proof.

It remains to show that if the above construction of P never halts, then the sequent $\Gamma \longrightarrow \Delta$ is not logically implied by Π. So suppose the above construction of P never halts and consider the result of applying the entire infinite construction process. From the details of the construction of P, P will be an infinite tree (except in the exceptional case where $\Gamma \longrightarrow \Delta$ contains only atomic formulas and Π is empty, in which case P is a single sequent). If Π is empty, then each node in the infinite tree P will be a sequent; however, in the general case, each node in the infinite tree will be a generalized sequent of the form $\Gamma', \Pi \longrightarrow \Delta'$ with an infinite number of formulas in its antecedent. (At each stage of the construction of P, the sequents contain only finitely many formulas, but in the limit the antecedents contain every formula from Π since these are introduced by step (1).) P is a finitely branching tree, so by König's lemma, there is at least one infinite branch π in P starting at the roots and proceeding up through the tree (except, in the exceptional case, π is to contain just the endsequent). We use π to construct a structure $\mathcal{M}$ and object assignment σ, for which $\Gamma \longrightarrow \Delta$ is not true. The universe of $\mathcal{M}$ is equal to the set of L-terms. The object assignment σ just maps a variable a to itself. The interpretation of a function symbol is defined so that $f^{\mathcal{M}}(r_1, \ldots, r_k)$ is the term $f(r_1, \ldots, r_k)$. Finally, the interpretation of a predicate symbol P is defined by letting $P^{\mathcal{M}}(r_1, \ldots, r_k)$ hold if and only the formula $P(r_1, \ldots, r_k)$ appears in an antecedent of a sequent contained in the branch π.

To finish the proof of the theorem, it suffices to show that every formula A occurring in an antecedent (respectively, a succedent) along π is true (respectively, false) in $\mathcal{M}$ with respect to σ; since this implies that $\Gamma \longrightarrow \Delta$ is not valid. This claim is proved by induction on the complexity of A. For A atomic, it is true by definition. Consider the case where A is of the form $(\exists x)B(x)$. If A appears in an antecedent, then so does a formula of the form $B(c)$; by the induction hypothesis, $B(c)$ is true in $\mathcal{M}$ w.r.t. σ, so hence A is. If A appears in an succedent, then, for every term t, $B(t)$ eventually appears in an succedent; hence every $B(t)$ is false in $\mathcal{M}$ w.r.t. σ, which implies A is also false. Note that A cannot appear in both an antecedent and a succedent along π, since the nodes in π were never active initial sequents. The rest of cases, for different outermost connectives of A, are similar and we omit their proofs.

2.3.8. There are number of *semantic tableau* proof systems, independently due to Beth [1956], Hintikka [1955], Kanger [1957] and Schütte [1965], which are very similar to the first-order cut-free sequent calculus. The proof above, of the completeness of the cut-free sequent calculus, is based on the proofs of the completeness of semantic tableau systems given by these four authors. The original proof, due to Gentzen, was based on the completeness of LK with cut and on a process of eliminating cuts from a proof similar to the construction of the next section.

2.4. Cut elimination. The cut-free completeness theorem, as established above, has a couple of drawbacks. Firstly, we have proved cut-free completeness only for pure LK and it is desirable to also establish a version of cut-free completeness (called 'free-cut free' completeness) for LK_e and for more general systems $LK_{\mathfrak{S}}$. Secondly, the proof is completely non-constructive and gives no bounds on the sizes of cut-free proofs. Of course, the undecidability of validity in first-order logic implies that the size of proofs (cut-free or otherwise) cannot be recursively bounded in terms of the formula being proved; instead, we wish to give an upper bound on the size of a cut-free LK-proof in terms of the size of a general LK-proof.

Accordingly, we shall give a constructive proof of the cut elimination theorem — this proof will give an effective procedure for converting a general LK-proof into a cut-free LK-proof. As part of our analysis, we compute an upper bound on the size of the cut-free proof constructed by this procedure. With some modification, the procedure can also be used to construct free-cut free proofs in LK_e or $LK_{\mathfrak{S}}$; this will be discussed in section 2.4.4.

It is worth noting that the cut elimination theorem proved next, together with the Completeness Theorem 2.2.3 for the Hilbert-style calculus, implies the Cut-free Completeness Theorem 2.3.7. This is because LK can easily simulate Hilbert-style proofs and is thus complete; and then, since any valid sequent has an LK-proof, it also has a cut-free LK-proof.

2.4.1. Definition. The *depth*, $dp(A)$, of a formula A is defined to equal the height of the tree representation of the formula; that is to say:

- $dp(A) = 0$, for A atomic,
- $dp(A \wedge B) = dp(A \vee B) = dp(A \supset B) = 1 + \max\{dp(A), dp(B)\}$,
- $dp(\neg A) = dp((\exists x)A) = dp((\forall x)A) = 1 + dp(A)$.

The *depth* of a cut inference is defined to equal the depth of its cut formula.

Definition. The *superexponentiation* function 2^x_i, for $i, x \geq 0$, is defined inductively by $2^x_0 = x$ and $2^x_{i+1} = 2^{2^x_i}$. Thus 2^x_i is the number which is expressed in exponential notation as a stack of i many 2's with an x at the top.

2.4.2. Cut-Elimination Theorem. *Let P be an LK-proof and suppose every cut formula in P has depth less than or equal to d. Then there is a cut-free LK-proof P^* with the same endsequent as P, with size*

$$||P^*|| \; < \; 2^{||P||}_{2d+2}.$$

Most proofs of this theorem are based on the original proof of Gentzen [1935] which involved making local changes to a proof to reduce the depth of cuts, the number of cuts, or the so-called rank of a cut. We present here a somewhat differently structured proof in which the depth or number of cuts is reduced by making *global* changes to a proof. We feel that our approach has the advantage of making the overall cut elimination process clearer and more intuitive.

The main step in proving the cut elimination theorem will be to establish the following lemma:

2.4.2.1. Lemma. *Let P be an LK-proof with final inference a cut of depth d such that every other cut in P has depth strictly less than d. Then there is an LK-proof P^* with the same endsequent as P with all cuts in P^* of depth less than d and with $||P^*|| < ||P||^2$.*

Proof. The proof P ends with a cut inference

$$\dfrac{\overset{\vdots\, Q}{\Gamma \longrightarrow \Delta, A} \qquad \overset{\vdots\, R}{A, \Gamma \longrightarrow \Delta}}{\Gamma \longrightarrow \Delta}$$

where the depth of the cut formula A equals d and where all cuts in the subproofs Q and R have depth strictly less than d. The lemma is proved by cases, based on the outermost logical connective of the cut formula A. We can assume w.l.o.g. that both Q and R contain at least one strong inference; since otherwise, we must have A in Γ or in Δ, or have a formula which occurs in both Γ and Δ, and in the former case, the sequent $\Gamma \longrightarrow \Delta$ is obtainable by weak inferences from one of the upper sequents and the cut can therefore be eliminated, and in the second case, $\Gamma \longrightarrow \Delta$ can be inferred with no cut inference at all. The proof P is also assumed to be in free variable normal form.

Case (a): Suppose A is a formula of the form $\neg B$. We shall form new proofs Q^* and R^* of the sequents $B, \Gamma \longrightarrow \Delta$ and $\Gamma \longrightarrow \Delta, B$, which can then be combined with a cut inference of depth $d-1$ to give the proof P^* of $\Gamma \longrightarrow \Delta$. To form Q^*, first form Q' by replacing every sequent $\Pi \longrightarrow \Lambda$ in Q with the sequent $\Pi, B \longrightarrow \Lambda^-$, where Λ^- is obtained from Λ by removing all direct ancestors of the cut formula A. Of course, Q' is not a valid proof; for example, a $\neg$*:right* inference in Q of the form

$$\dfrac{B, \Pi \longrightarrow \Lambda}{\Pi \longrightarrow \Lambda, \neg B}$$

could become in Q'

$$\dfrac{B, \Pi, B \longrightarrow \Lambda^-}{\Pi, B \longrightarrow \Lambda^-}$$

This is not, strictly speaking, a valid inference; but of course, it can be modified to be valid by inserting some exchanges and a contraction. In this fashion, it is straightforward to modify Q' so that it becomes a valid proof Q^* by removing some $\neg$*:left* inferences and inserting some weak inferences. We leave it to the reader to check that Q^* can be validly formed in this way: it should be noted that we are using the assumption that initial sequents $C \longrightarrow C$ must have C atomic. The proof, R^*, of $\Gamma \longrightarrow \Delta, B$ is formed in a similar fashion from R. Obviously, no new cuts are introduced by this process and, since we do not count weak inferences, $||Q^*|| \le ||Q||$ and $||R^*|| \le ||R||$; thus P^* has only cuts of depth $< d$ and has $||P^*|| \le ||P||$.

Case (b): Now suppose the cut formula A is of the form $B \vee C$. We define Q' as a tree of sequents, with root labeled $\Gamma \longrightarrow \Delta, B, C$, by replacing every sequent $\Pi \longrightarrow \Lambda$ in Q with the sequent $\Pi \longrightarrow \Lambda^-, B, C$, where Λ^- is Λ minus all occurrences of direct ancestors of the cut formula. By removing some formerly $\vee$*:right* inferences from Q' and by adding some weak inferences, Q' can be transformed into a valid proof Q^*. Now construct R_B from R by replacing every occurrence in R of $B \vee C$ as a direct ancestor of the cut formula with just the formula B. One way that R_B can fail to be a valid proof is that an $\vee$*:left* inference

$$\frac{B, \Pi \longrightarrow \Lambda \qquad C, \Pi \longrightarrow \Lambda}{B \vee C, \Pi \longrightarrow \Lambda}$$

may become just

$$\frac{B, \Pi \longrightarrow \Lambda \qquad C, \Pi \longrightarrow \Lambda}{B, \Pi \longrightarrow \Lambda}$$

in R_B. This is no longer a valid inference, but it can be fixed up by discarding the inference and its upper right hypothesis, including discarding the entire subproof of the upper right hypothesis. The only other changes needed to make R_B valid are the addition of weak inferences, and in this way, a valid proof R_B of $B, \Gamma \longrightarrow \Delta$ is formed. A similar process forms a valid proof R_C of $C, \Gamma \longrightarrow \Delta$. The proof P^* can now be defined to be

$$\dfrac{\dfrac{\dfrac{\vdots\, Q^*}{\Gamma \longrightarrow \Delta, B, C} \qquad \dfrac{\vdots\, R_C}{C, \Gamma \longrightarrow \Delta}}{\Gamma \longrightarrow \Delta, B} \qquad \dfrac{\vdots\, R_B}{B, \Gamma \longrightarrow \Delta}}{\Gamma \longrightarrow \Delta}$$

The processes of forming Q^*, R_B, and R_C did not introduce any new cuts or any new strong inferences. Thus we clearly have that every cut in P^* has depth $< d$, and that $||P^*|| \le ||Q|| + 2||R|| + 2$. Since $||P|| = ||Q|| + ||R|| + 1$ and $||Q||, ||R|| \ge 1$, this suffices to prove the lemma for this case.

Cases (c),(d): The cases where A has outermost connective $\wedge$ or $\supset$ are very similar to the previous case, and are omitted.

Case (e): Now suppose A is of the form $(\exists x)B(x)$. First consider how the formula $(\exists x)B(x)$ can be introduced in the subproof Q. Since it is not atomic, it cannot be introduced in an initial sequent; thus, it can only be introduced by weakenings and by $\exists$*:right* inferences. Suppose that there are $k \ge 0$ many $\exists$*:right* inferences in Q which have their principal formula a direct ancestor of the cut formula. These can be enumerated as

$$\frac{\Pi_i \longrightarrow \Lambda_i, B(t_i)}{\Pi_i \longrightarrow \Lambda_i, (\exists x)B(x)}$$

with $1 \le i \le k$. Similarly, we locate all the $\exists$*:left* inferences in R which have principal formula a direct ancestor of the cut formula and enumerate these as

$$\frac{B(a_i), \Pi'_i \longrightarrow \Lambda'_i}{(\exists x)B(x), \Pi'_i \longrightarrow \Lambda'_i}$$

for $1 \le i \le \ell$.

For each $i \le k$, we form a proof R_i of the sequent $B(t_i), \Gamma \longrightarrow \Delta$ by replacing all ℓ of the variables a_j with the term t_i everywhere in R, replacing every direct ancestor of the cut formula $(\exists x)B(x)$ in R with $B(t_i)$, and then removing the ℓ $\exists$*:left* inferences. It is easy to see that this yields a valid proof; note that the fact that P is in free variable normal form ensures that replacing the a_j's with t_i will not impact the eigenvariable conditions for inferences in R.

Now, form Q' from Q by replacing each sequent $\Pi \longrightarrow \Lambda$ in Q with the sequent $\Pi, \Gamma \longrightarrow \Delta, \Lambda^-$, where Λ^- is Λ minus all direct ancestors of the cut formula A.

Clearly, Q' ends with the sequent $\Gamma,\Gamma \longrightarrow \Delta,\Delta$; however, it is not a valid proof. To fix it up to be a valid proof, we need to do the following. First, an initial sequent in Q' will be of the form $A,\Gamma \longrightarrow \Delta, A$; this can be validly derived by using the initial sequent $A \longrightarrow A$ followed by weakenings and exchanges. Second, for $1 \le i \le k$, the i-th $\exists$*:right* inference enumerated above, will be of the form

$$\frac{\Pi_i, \Gamma \longrightarrow \Delta, \Lambda_i, B(t_i)}{\Pi_i, \Gamma \longrightarrow \Delta, \Lambda_i}$$

in Q. This can be replaced by the following inferences

$$\begin{array}{c} \qquad\qquad\qquad\qquad\qquad\qquad \vdots\, R_i \\ \Pi_i, \Gamma \longrightarrow \Delta, \Lambda_i, B(t_i) \qquad B(t_i), \Gamma \longrightarrow \Delta \\ \hline\hline \Pi_i, \Gamma \longrightarrow \Delta, \Lambda_i \end{array}$$

Note that this has replaced the $\exists$*:right* inference of Q with a cut of depth $d-1$ and some weak inferences.

No further changes are needed to Q' to make it a valid proof. In particular, the eigenvariable conditions still hold since no free variable in $\Gamma \longrightarrow \Delta$ is used as an eigenvariable in Q. By adding some exchanges and contractions to the end of this proof, the desired proof P^* of $\Gamma \longrightarrow \Delta$ is obtained. It is clear that every cut in P^* has depth $< d$. From inspection of the construction of P^*, the size of P^* can be bounded by

$$||P^*|| \ \le \ ||Q|| \cdot (||R|| + 1) \ < \ ||P||^2.$$

Case (f): The case where A is of the form $(\forall x)B(x)$ is completely dual to case (e), and is omitted here.

Case (g): Finally, consider the case where A is atomic. Form R' from R by replacing every sequent $\Pi \longrightarrow \Lambda$ in R with the sequent $\Pi^-, \Gamma \longrightarrow \Delta, \Lambda$, where Π^- is Π minus all occurrences of direct ancestors of A. R' will end with the sequent $\Gamma,\Gamma \longrightarrow \Delta,\Delta$ and will be valid as a proof, except for its initial sequents. The initial sequents $B \longrightarrow B$ in R, with B not a direct ancestor of the cut formula A, become $B,\Gamma \longrightarrow \Delta, B$ in R'; these are readily inferred from the initial sequent $B \longrightarrow B$ with only weak inferences. On the other hand, the other initial sequents $A \longrightarrow A$ in R become $\Gamma \longrightarrow \Delta, A$ which is just the endsequent of Q. The desired proof P^* of $\Gamma \longrightarrow \Delta$ is thus formed from R' by adding some weak inferences and adding some copies of the subproof Q to the leaves of R', and by adding some exchanges and contractions to the end of R'.

Since Q and R have only cuts of degree $< d$ (i.e., have no cuts, since $d = 0$), P^* likewise has only cuts of degree $< d$. Also, since the number of initial sequents in R' is bounded by $||R|| + 1$, the size of P^* can be bounded by

$$||P^*|| \le ||R|| + ||Q|| \cdot (||R|| + 1) < (||Q|| + 1)(||R|| + 1) < ||P||^2$$

That completes the proof of Lemma 2.4.2.1.

Lemma 2.4.2.1 shows how to replace a single cut by lower depth cut inferences. By iterating this construction, it is possible to remove all cuts of the maximum depth d in a proof. This is stated as Lemma 2.4.2.2: the Cut-Elimination Theorem is an immediate consequence of this lemma.

2.4.2.2. Lemma. *If P is an LK-proof with all cuts of depth at most d, there is an LK-proof P^* with the same endsequent which has all cuts of depth strictly less than d and with size $||P^*|| < 2^{2^{||P||}}$.*

Proof. Lemma 2.4.2.2 will be proved by induction on the number of depth d cuts in P. The base case, where there are no depth d cuts is trivial, of course, since $||P|| < 2^{2^{||P||}}$. For the induction step, it suffices to prove the lemma in the case where P ends with a cut inference

$$\dfrac{\begin{array}{c}\vdots\, Q\\ \Gamma \longrightarrow \Delta, A\end{array} \qquad \begin{array}{c}\vdots\, R\\ A, \Gamma \longrightarrow \Delta\end{array}}{\Gamma \longrightarrow \Delta}$$

where the cut formula A is of depth d.

First suppose that one of the subproofs, say R, does not have any strong inferences; i.e., $||R|| = 0$. Therefore, R must either contain the axiom $A \longrightarrow A$, or must have direct ancestors of the cut formula A introduced only by weakenings. In the former case, A must appear in Δ, and the desired proof P^* can be obtained from Q by adding some exchanges and a contraction to the end of Q. In the second case, P^* can be obtained from R by removing all the *Weakening:left* inferences that introduce direct ancestors of the cut formula A (and possibly removing some exchanges and contractions involving these A's). A similar argument works for the case $||Q|| = 0$. In both cases, $||P^*|| < ||P|| < 2^{2^{||P||}}$.

Second, suppose that $||Q||$ and $||P||$ are both nonzero. By the induction hypothesis, there are proofs Q^* and R^* of the same endsequents, with all cuts of depth less than d, and with

$$||Q^*|| < 2^{2^{||Q||}} \qquad \text{and} \qquad ||R^*|| < 2^{2^{||R||}}$$

Applying Lemma 2.4.2.1 to the proof

$$\dfrac{\begin{array}{c}\vdots\, Q^*\\ \Gamma \longrightarrow \Delta, A\end{array} \qquad \begin{array}{c}\vdots\, R^*\\ A, \Gamma \longrightarrow \Delta\end{array}}{\Gamma \longrightarrow \Delta}$$

gives a proof P^* of $\Gamma \longrightarrow \Delta$ with all cuts of depth $< d$, so that

$$||P^*|| \;<\; \left(||Q^*|| + ||R^*|| + 1\right)^2 \;\le\; \left(2^{2^{||Q||}} + 2^{2^{||R||}} - 1\right)^2 \;<\; 2^{2^{||Q||+||R||+1}} \;=\; 2^{2^{||P||}}.$$

The final inequality holds since $||Q||, ||R|| \ge 1$.
Q.E.D. Lemma 2.4.2.2 and the Cut Elimination Theorem.

2.4.3. A general bound on cut elimination. The upper bound $2_{2d+2}^{||P||}$ in the Cut Elimination Theorem is based not only on the size of P, but also on the maximum depth of the cut formulas in P. However, there is a general method that allows the bound to be expressed in terms of just $||P||$. This is based on the following:

Proposition. *Suppose P is an LK-proof of the sequent $\Gamma \longrightarrow \Delta$. Then there is a cut-free proof P^* of the same sequent with size $||P^*|| < 2_{2||P||}^{||P||}$.*

Proof. The proposition is obviously true if P has no cuts, so suppose it contains at least one. We need some definitions: Two semiformulas are said to be *term variants* if they have identical logical structure and thus one can be transformed into the other by changing only its semiterms. Obviously the 'term variant' relation is an equivalence relation. For any equivalence class of term variants, there is a formula skeleton $R(\tau_1, \ldots, \tau_k)$ such that the equivalence class contains exactly the semiformulas of the form $R(t_1, \ldots, t_k)$ where $t_1, \ldots, t_k$ are semiterms.

Let $c_1, c_2, c_3, \ldots$ be an infinite sequence of new free variables. For $R(\cdots)$ a formula skeleton for a term variant class, we pick a new predicate symbol P_R. Given a member $R(t_1, \ldots, t_k)$ of the term variant class, we can form the atomic semiformula $P_R(t_1, \ldots, t_k)$ and its bound variables correspond to the bound variables which are freely occurring in $R(\vec{t})$.

A formula A is defined to be *active* in P if some strong inference in P has a term variant of A as a principal formula. Suppose A is non-atomic and not active in P; let $R(\cdots)$ give the term variant class of A. Consider the following transformation of P: for each term variant $R(t_1, \ldots, t_k)$ which appears as a subformula of a formula in P, replace it with $P_R(t_1, \ldots, t_k)$. It can be checked that this transformation yields a valid proof, which contains exactly the same kinds of inferences as P.

By repeatedly applying this transformation, a valid proof P' is obtained in which every subformula either is atomic or is a term variant of a principal formula of a strong inference in P'. Since there are at most $||P||$ many strong inferences in P', including at least one cut inference, and since initial sequents contain only atomic formulas, this implies that every formula in P' has depth less than $||P||$. (To prove the last assertion, note that every formula in P' either is atomic or has an atomic ancestor.) Therefore, by the cut elimination theorem, there is a cut-free proof P'' with the same endsequent of size $||P''|| < 2_{2||P||}^{||P||}$.

This proof P'' has the desired size, but it may no longer be a proof of $\Gamma \longrightarrow \Delta$, since it may contain subformulas of the form $P_R(t_1, \ldots, t_k)$. However, we may (iteratively) replace all such subformulas in P'' with the formula $R(t_1, \ldots, t_k)$. This yields the desired proof of $\Gamma \longrightarrow \Delta$. $\square$

2.4.4. Free-cut elimination. We next investigate the possibility of eliminating cuts in $LK_{\mathfrak{S}}$-proofs; that is to say, in proofs in which initial sequents may come from $\mathfrak{S}$. The set $\mathfrak{S}$ is presumed to be a fixed set of sequents closed under substitution. An important example of such an $\mathfrak{S}$ is the set of equality axioms of LK_e; however, $\mathfrak{S}$ can also be the axioms of any first-order theory.

The Cut Elimination Theorem 2.4.2 applied only to LK-proofs; on the other hand, the proof of Corollary 2.3.7 gave a partial cut elimination theorem for $LK_{\mathfrak{S}}$-proofs. These results can be significantly improved by introducing a notion of 'free' cuts and giving a construction eliminating free cuts from $LK_{\mathfrak{S}}$-proofs by a method similar to the proof of Theorem 2.4.2.

2.4.4.1. Definition. Let P be an $LK_{\mathfrak{S}}$-proof. A formula occurring in P is *anchored* (by an $\mathfrak{S}$-sequent) if it is a direct descendent of a formula occurring in an initial sequent in $\mathfrak{S}$. A cut inference in P is *anchored* if either:

(i) the cut formula is not atomic and at least one of the two occurrences of the cut formula in the upper sequents is anchored, or

(ii) the cut formula is atomic and both of the occurrences of the cut formula in the upper sequents are anchored.

A cut inference which is not anchored is said to be *free*. The proof P is *free-cut free* if it contains no free cuts.

2.4.4.2. An occurrence of a formula in a proof P is said to be *only weakly introduced* in P if it does not have a direct ancestor which appears in an initial sequent or which is a principal formula of a strong inference. It is often convenient to assume that a proof P satisfies the condition that no cut formula is only weakly introduced; that is to say, that every cut inference satisfies the condition that neither occurrence of its cut formula is only weakly introduced in P.

Note that if P is an $LK_{\mathfrak{S}}$-proof, then P can be assumed w.l.o.g. to satisfy this extra condition without an increase in the size of the proof or in the depth of cuts in the proof. However, conforming to this convention might increase the number and depth of free cuts in the proof P. To see this suppose that P contains the cut inference with one of its cut formulas weakly introduced; for instance, suppose that an inference

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

has the A occurring in the right upper sequent $A, \Gamma \longrightarrow \Delta$ only weakly introduced. Now, it is possible that the subproof of $\Gamma \longrightarrow \Delta, A$ causes a formula B in Γ or Δ to be anchored, and that the corresponding B is not anchored in the subproof of $A, \Gamma \longrightarrow \Delta$. The obvious way to eliminate this cut is to remove all direct ancestors of the A in the right upper sequent, thereby getting a proof of $\Gamma \longrightarrow \Delta$, and then discard the left upper sequent and its proof. This, of course, causes B to be only weakly introduced in the new subproof of $\Gamma \longrightarrow \Delta$. In other words, the occurrence of B in the sequent $\Gamma \longrightarrow \Delta$ becomes unanchored. Then, if a direct descendent of B is later used as a cut formula, the elimination of the cut on A could have changed the cut from an anchored cut into a free cut.

Our proof of the free-cut elimination theorem will use induction on the maximum depth of free cuts appearing in a proof. For this induction to work, it is important that anchored cuts do not change into free cuts, introducing new free cuts of higher depth. To avoid this, we will assume that no cut formulas in the proof are only weakly introduced.

2.4.5. Free-cut Elimination Theorem. *Let $\mathfrak{S}$ be a set of sequents closed under substitution.*

(1) *If $LK_{\mathfrak{S}} \vdash \Gamma \longrightarrow \Delta$, then there is a free-cut free $LK_{\mathfrak{S}}$-proof of $\Gamma \longrightarrow \Delta$.*

(2) *Let P be an $LK_{\mathfrak{S}}$-proof satisfying condition* 2.4.4.2 *and suppose every anchored cut formula in P has depth less than or equal to d. Then there is a free-cut free $LK_{\mathfrak{S}}$-proof P^* with the same endsequent as P, with size*

$$||P^*|| \ < \ 2^{||P||}_{2d+2}.$$

The Free-cut Elimination Theorem is essentially due to Gentzen; Takeuti [1987] states a related construction for use with induction rules, which we describe in section 2.4.6.

Proof. Obviously, it suffices to prove part (2) of Theorem 2.4.5. For this proof, we shall give a procedure (Lemma 2.4.5.1) for removing one maximum depth free cut. It is not possible to use the procedure from the proof of Lemma 2.4.2.1 without modification, because it may remove $\mathfrak{S}$-sequents from the proof and thereby change some anchored cuts into free cuts (and thereby increase the maximum depth of free cuts). This is unacceptable since our proof uses induction on the maximum depth of free cuts in P. This undesirable situation can happen in case (b) of the proof of Lemma 2.4.2.1 where the outermost connective of the cut formula A is $\vee$, since at various points, subproofs ending with $C, \Pi \longrightarrow \Lambda$ or with $B, \Pi \longrightarrow \Lambda$ are just discarded in R'_B and in R'_C. Subformulas in Π and Λ may become unanchored by this process. This can happen also in the similar cases (c) and (d). In addition, it could also happen in cases (e), (f) and (g) if the cut formula occurring in the right upper sequent is only weakly introduced, since in these cases the subproof on the left is completely discarded; however, since condition 2.4.4.2 holds, this never occurs. Therefore, we need the following analogue of Lemma 2.4.2.1:

2.4.5.1. Lemma. *Let P be an $LK_{\mathfrak{S}}$-proof with final inference a free cut of depth d such that every other free cut in P has depth strictly less than d. Then there is an $LK_{\mathfrak{S}}$-proof P^* with the same endsequent as P with all free cuts in P^* of depth less than d and with $||P^*|| < ||P||^2$. Furthermore, every formula occurring in the endsequent of P which was anchored by an $\mathfrak{S}$-sequent is still anchored in the proof P^*, and every formula in the endsequent of P^* which is only weakly introduced in P^* was already only weakly introduced in P.*

Proof. We indicate how to modify the proof of Lemma 2.4.2.1. Assume that the proof P ends with a free cut inference

$$\dfrac{\begin{array}{c}\vdots\, Q \\ \Gamma \longrightarrow \Delta, A\end{array} \qquad \begin{array}{c}\vdots\, R \\ A, \Gamma \longrightarrow \Delta\end{array}}{\Gamma \longrightarrow \Delta}$$

If the cut formula A is nonatomic, then Q and R both contain at least one strong inference with principal formula equal to A. As before the proof splits into cases depending on the outermost connective of A. When A has outermost connective $\neg$, $\exists$ or $\forall$ then the argument used in cases (a), (e) and (f) of Lemma 2.4.2.1 still works: we leave it to the reader to check that, in these cases, any formulas in the sequent $\Gamma \longrightarrow \Delta$ which were anchored or were not only weakly introduced in P still have these properties in P^*.

For the case where A is of the form $B \vee C$, a different construction is needed. In this case, form a proof Q^* with endsequent $\Gamma \longrightarrow \Delta, B, C$ by the construction used in case (b) of the proof of Lemma 2.4.2.1. Also form R' from R be replacing every sequent $\Pi \longrightarrow \Lambda$ in R with the sequent $\Pi^-, \Gamma \longrightarrow \Delta, \Lambda$ where Π^- is Π minus all direct ancestors of the cut formula $B \vee C$. An $\vee$*:left* inference in R with principal formula $B \vee C$ will no longer be a valid inference since, in R' it will become

$$\frac{B, \Pi^-, \Gamma \longrightarrow \Delta, \Lambda \qquad C, \Pi^-, \Gamma \longrightarrow \Delta, \Lambda}{\Pi^-, \Gamma \longrightarrow \Delta, \Lambda}$$

This can be transformed into a valid proof by replacing it with

$$\dfrac{\dfrac{\dfrac{\vdots\, Q}{\Gamma \longrightarrow \Delta, B, C} \qquad \dfrac{\vdots}{C, \Pi^-, \Gamma \longrightarrow \Delta, \Lambda}}{\Pi^-, \Gamma \longrightarrow \Delta, \Lambda, B} \qquad \dfrac{\vdots}{B, \Pi^-, \Gamma \longrightarrow \Delta, \Lambda}}{\Pi^-, \Gamma \longrightarrow \Delta, \Lambda}$$

Fixing up R' in this way, plus adding some weak inferences yields a valid proof R^* of $\Gamma, \Gamma \longrightarrow \Delta, \Delta$. Appending some exchange and contraction inferences yields the desired proof P^*. It is readily checked that $||P^*|| \leq ||R|| \cdot (||Q|| + 1) < ||P||^2$.

The final case to consider is the case where A is atomic. Since the cut is not anchored, the cut formula will either not be anchored in Q or not be anchored in R. In the latter case, since the cut formula is not only weakly introduced, it must have a direct ancestor in an initial sequent $A \longrightarrow A$ of R. Therefore, the argument from case (g) of the proof of Lemma 2.4.2.1 still works. In the former case, where the cut formula is not anchored in Q, the dual argument works.

That completes the proof of Lemma 2.4.5.1. The Free-cut Elimination Theorem follows immediately from this lemma in the same way that the Cut-Elimination Theorem followed from Lemma 2.4.2.1.
Q.E.D. Free-cut Elimination Theorem.

2.4.6. Free-cut elimination with induction rules. A very useful application of free-cut elimination is for analyzing subtheories of arithmetic in which induction is restricted to certain classes of formulas. For these fragments of arithmetic, the free-cut elimination theorem becomes easier to use if induction rules are used in place of induction axioms. The most common formulation of induction rules is as inferences of the form:

$$\frac{A(b), \Gamma \longrightarrow \Delta, A(b+1)}{A(0), \Gamma \longrightarrow \Delta, A(t)}$$

where b is an eigenvariable and may appear only as indicated, and t is an arbitrary term. It is easily checked that, because of the presence of the side formulas Γ and Δ, the induction rule for A is equivalent to the induction axiom for A:

$$A(0) \wedge (\forall x)(A(x) \supset A(x+1)) \supset (\forall x)A(x).$$

For Φ a set of formulas, Φ-IND indicates the theory with the above induction rules for all formulas $A \in \Phi$. We shall always assume that such a set Φ is closed under (term) substitution.

The classic examples of arithmetic theories axiomatized by induction are the theories $I\Sigma_n$. These are axiomatized by the six axioms of Robinson's theory Q plus induction for Σ_n formulas (see Chapter II). When $I\Sigma_n$ is formalized in the sequent calculus, it has the non-logical initial sequents expressing the axioms of Q plus the induction rule for Σ_n formulas. The initial sequents of $I\Sigma_n$ proofs are always purely existential. Along the same lines, the fragments T_2^n and S_2^n of bounded arithmetic are commonly formalized in the sequent calculus with quantifier-free non-logical initial sequents from BASIC, plus induction on Σ_n^b-formulas (see Chapter II or Buss [1986]). For T_2^n, the induction rules are as shown above; for S_2^n the following *PIND* rules are used:

$$\frac{A(\lfloor \frac{1}{2}b \rfloor), \Gamma \longrightarrow \Delta, A(b)}{A(0), \Gamma \longrightarrow \Delta, A(t)}$$

Definition. Let P be a sequent calculus proof in an arithmetic theory $\mathfrak{S} + \Phi$-IND (or $\mathfrak{S} + \Phi$-PIND). The *principal formulas* of an induction inference are the formulas denoted by $A(0)$ and $A(t)$ in the lower sequent. An occurrence of a formula in P is defined to be *anchored* if it is a direct descendent of a formula occurring in an initial sequent from $\mathfrak{S}$ or it is a direct descendent of a principal formula of an induction inference.

Using this definition of anchored formulas, the notions of *anchored* cut inference and *free* cut inference are defined identically to Definition 2.4.4.1.

Free-cut Elimination Theorem. *Let $T = \mathfrak{S} + \Phi$-IND be a theory of arithmetic, with $\mathfrak{S}$ and Φ closed under term substitution. Let $\Gamma \longrightarrow \Delta$ be a logical consequence of T. Then there is a free-cut free T-proof of $\Gamma \longrightarrow \Delta$. Furthermore, the size bounds of Theorem 2.4.5(2) apply to T-proofs.*

This theorem is proved by an argument identical to the proof of Theorem 2.4.5. The theorem also applies without modification to theories axiomatized with PIND in place of IND. It is also straightforward to modify the free cut elimination to work with sequent calculus formulations of inference rules other than induction. For example, the collection (replacement) axioms for fragments of Peano arithmetic can be equivalently formulated as sequent calculus rules. The obvious straightforward modification of the Free-cut Elimination theorem applies to theories with collection rules.

A particularly useful corollary of the Free-Cut Elimination Theorem is:

2.4.7. Corollary. *Let $\mathfrak{S}$ be a set of sequents and Φ be a set of formulas closed under term substitution and under subformulas, such that every sequent in $\mathfrak{S}$ contains only formulas from Φ. Let T be the theory $\mathfrak{S}+\Phi$-IND (or $\mathfrak{S}+\Phi$-PIND). Suppose that $\Gamma \longrightarrow \Delta$ is a consequence of T and that every formula in $\Gamma \longrightarrow \Delta$ is in Φ. Then there is a T-proof P of $\Gamma \longrightarrow \Delta$ such that every formula appearing in P is in Φ.*

To prove this corollary, just note that it holds for every free-cut free T-proof of $\Gamma \longrightarrow \Delta$; this is because every formula in the proof must be an ancestor of either a cut formula or a formula in the endsequent.

2.4.8. Natural deduction. Natural deduction proof systems were introduced by Gentzen [1935] in the same paper which introduced the first-order sequent calculus. Gentzen's motivation in defining natural deduction was (in his words) "to set up a formula system which comes as close as possible to actual reasoning."[4] The importance of natural deduction was established by the classic result of Prawitz [1965] that a version of the cut elimination holds also for natural deduction.

It is fair to say that the process of constructing natural deduction proofs is indeed 'natural' in that it corresponds closely to the human reasoning process. On the other hand, a fully constructed natural deduction proof can be very confusing to read; in particular, because of the non-local nature of natural deduction proofs, it is difficult to quickly ascertain which formulas depends on which hypotheses.

Natural deduction proof systems are particularly elegant for intuitionistic logic, especially with respect to the Curry-Howard formulas-as-types interpretation (see section 3.1.5). A good modern treatment of applications of natural deduction is given by Girard [1989].

We give a short definition of the natural deduction proof system here; however, the reader should refer to Prawitz [1965] for a full treatment and for statements of the normalization theorems. The definitions of terms and formulas and the conventions on free and bound variables are the same for natural deduction as for the sequent calculus, except that negation, $\neg$, is not a basic symbol; instead, a new atomic formula $\bot$ is used to denote absurdity (the constant *False*), and $\neg A$ abbreviates $A \supset \bot$. A natural deduction proof is a tree of formulas; any formula may appear at a leaf, as a hypothesis. Various inferences may close or *discharge* the hypotheses; in a completed proof all hypotheses must be discharged and the proof is a proof of the formula appearing at the root node at the bottom of the tree. It is best to picture the construction of a natural deduction proof as an ongoing process; at any point in this process some hypotheses may already be discharged, whereas others remain open. The valid rules of inference are given below; they are classified as *introduction* rules

[4] Gentzen [1969], p. 68

or *elimination* rules. Hypotheses discharged by an inference are shown in square brackets.

$$\wedge\text{-intro}\quad \frac{A \qquad B}{A \wedge B} \qquad\qquad \wedge\text{-elim}\quad \frac{A \wedge B}{A} \qquad \frac{A \wedge B}{B}$$

$$\vee\text{-intro}\quad \frac{A}{A \vee B} \qquad \frac{B}{A \vee B} \qquad\qquad \vee\text{-elim}\quad \frac{A \vee B \qquad \begin{matrix}[A]\\ C\end{matrix} \qquad \begin{matrix}[B]\\ C\end{matrix}}{C}$$

$$\supset\text{-intro}\quad \frac{\begin{matrix}[A]\\ B\end{matrix}}{A \supset B} \qquad\qquad \supset\text{-elim}\quad \frac{A \qquad A \supset B}{B}$$

$$\forall\text{-intro}\quad \frac{A(b)}{(\forall x)A(x)} \qquad\qquad \forall\text{-elim}\quad \frac{(\forall x)A(x)}{A(t)}$$

$$\exists\text{-intro}\quad \frac{A(t)}{(\exists x)A(x)} \qquad\qquad \exists\text{-elim}\quad \frac{(\exists x)A(x) \qquad \begin{matrix}[A(b)]\\ B\end{matrix}}{B}$$

$$\bot_I\quad \frac{\bot}{A} \qquad\qquad \bot_C\quad \frac{\begin{matrix}[A \supset \bot]\\ \bot\end{matrix}}{A}$$

In the $\forall$-intro and $\exists$-elim rules, the free variable b is an eigenvariable: this means it must not appear in any non-discharged hypothesis above the $\forall$-intro inference or in any non-discharged hypotheses other than $A(b)$ above the hypothesis B of the $\exists$-elim inference. The $\bot_I$ rule is used for both intuitionistic and classical logic; the $\bot_C$ rule is included for classical logic. If both rules for $\bot$ are omitted, then *minimal* logic is obtained.

2.5. Herbrand's theorem, interpolation and definability theorems

2.5.1. Herbrand's theorem. One of the fundamental theorems of mathematical logic is the theorem of Herbrand [1930] which allows a certain type of reduction of first-order logic to propositional logic. In its basic form it states:

Herbrand's Theorem. *Let T be a theory axiomatized by purely universal formulas. Suppose that $T \vDash (\forall\vec{x})(\exists y_1, \ldots, y_k)B(\vec{x}, \vec{y})$ with $B(\vec{x}, \vec{y})$ a quantifier-free formula. Then there is a finite sequence of terms $t_{i,j} = t_{i,j}(\vec{x})$, with $1 \le i \le r$ and $1 \le j \le k$ so that*

$$T \vdash (\forall\vec{x})\left(\bigvee_{i=1}^{r} B(\vec{x}, t_{i,1}, \ldots, t_{i,k})\right).$$

Proof. Since T is axiomatized by purely universal formulas, it may, without loss of generality, be axiomatized by quantifier-free formulas (obtained by removing the universal quantifiers). Let $\mathfrak{T}$ be the set of sequents $\longrightarrow A$ with A a (quantifier-free) axiom of T. Since $T \vDash (\forall\vec{x})(\exists\vec{y})B(\vec{x}, \vec{y})$, there is a $LK_{\mathfrak{T}}$-proof of the sequent

$\longrightarrow (\exists\vec{y})B(\vec{a},\vec{y})$. By the free-cut elimination theorem, there is a free-cut free proof P of this sequent.

Since the $\mathfrak{T}$-sequents contain only quantifier-free formulas, all cut formulas in P are quantifier-free. Thus, any non-quantifier-free formula in P must be of the form $(\exists y_j)\cdots(\exists y_k)B(\vec{a},t_1,\ldots,t_{j-1},y_j,\ldots,y_k)$ with $1 \le j < k$. We claim that P can be modified to be a valid proof of a sequent of the form

$$\longrightarrow B(\vec{a},t_{1,1},\ldots,t_{1,k}),\ldots,B(\vec{a},t_{r,1},\ldots,t_{r,k}).$$

The general idea is to remove all $\exists$*:right* inferences in P and remove all existential quantifiers, replacing the bound variables by appropriate terms. Since there may have been contractions on existential formulas that are no longer identical after terms are substituted for variables it will also be necessary to remove contractions and add additional formulas to the sequents. To do this more formally, we know that any sequent in P is of the form $\Gamma \longrightarrow \Delta, \Delta'$ (up to order of the formulas in the sequent), where each formula in Γ and Δ is quantifier-free and where each formula in Δ' is not quantifier-free but is purely existential. We can then prove by induction on the number of lines in the free-cut free proof of $\Gamma \longrightarrow \Delta, \Delta'$ that there is an $r \ge 0$ and a cedent Δ'' of the form

$$B(\vec{a},t_{1,1},\ldots,t_{1,k}),\ldots,B(\vec{a},t_{r,1},\ldots,t_{r,k})$$

such that $\Gamma \longrightarrow \Delta, \Delta''$ is provable. We leave the rest of the details to the reader. □

We momentarily define an *instance* of a universal formula $(\forall\vec{x})A(\vec{x})$ to be any quantifier-free formula $A(\vec{t})$. It is not hard to see using cut elimination, that if a quantifier-free formula C is a consequence of a universal theory T, then it is a tautological consequence of some finite set of instances of axioms of T and of equality axioms. In the special case where T is the null theory, we have that C is a consequence of instances of equality axioms (and C is therefore called a *quasitautology*). If, in addition, C does not involve equality, C will be tautologically valid. Thus, Herbrand's theorem reduces provability in first-order logic to generation of (quasi)tautologies.

2.5.2. As stated above, Herbrand's theorem has limited applicability since it applies only to Π_2-consequences of universal theories: fortunately, however, there are several ways to extend Herbrand's theorem to more general situations. In 2.5.3 below, we explain one such generalization; but first, in this paragraph, we give a simpler method of widening the applicability of Herbrand's theorem, based on the introduction of new function symbols, which we call *Herbrand* and *Skolem* functions, that allow quantifier alternations to be reduced.

For notational convenience, we will consider only formulas in prenex form in this paragraph; however, the definitions and proposition below can be readily generalized to arbitrary formulas.

Definition. Let $(\exists x)A(x,\vec{c})$ be a formula with $\vec{c}$ all of its free variables. The *Skolem function* for $(\exists x)A$ is represented by a function symbol $f_{\exists xA}$ and has the defining axiom:

$$\textit{Sk-def}(f_{\exists xA}): \quad (\forall \vec{y})(\forall x)\,(A(x,\vec{y}) \supset A(f_{\exists xA}(\vec{y}),\vec{y}))\,.$$

Note that $\textit{Sk-def}(f_{\exists xA})$ implies $(\forall \vec{y})\,((\exists x)A(x,\vec{y}) \leftrightarrow A(f_{\exists xA}(\vec{y}),\vec{y}))$.

Definition. Let $A(\vec{c})$ be a formula in prenex form. The *Skolemization*, $A^S(\vec{c})$, of A is the formula defined inductively by:

(1) If $A(\vec{c})$ is quantifier-free, then $A^S(\vec{c})$ is $A(\vec{c})$.

(2) If $A(\vec{c})$ is $(\forall y)B(\vec{c},y)$, then $A^S(\vec{c})$ is the formula $(\forall y)B^S(\vec{c},y)$.

(3) If $A(\vec{c})$ is $(\exists y)B(\vec{c},y)$, then $A^S(\vec{c})$ is $B^S(\vec{c},f_A(\vec{c}))$, where f_A is the Skolem function for A.

It is a simple, but important fact that $A^S \vDash A$.

The *Skolemization* of a theory T is the theory $T^S = \{A^S : A \in T\}$. Note that T^S is a purely universal theory. Incidentally, the set of all *Sk-def* axioms of the Skolem functions is equivalent to a set of universal formulas; however, they are not included in theory T^S. From model-theoretic considerations, it is not difficult to see that T^S contains and is conservative over T.

We next define the concept of 'Herbrandization' which is completely dual to the notion of Skolemization:

Definition. Let $(\forall x)A(x,\vec{c})$ be a formula with $\vec{c}$ all of its free variables. The *Herbrand function* for $(\forall x)A$ is represented by a function symbol $h_{\forall xA}$ and has the defining axiom:

$$(\forall \vec{y})(\forall x)\,(\neg A(x,\vec{y}) \supset \neg A(h_{\forall xA}(\vec{y}),\vec{y}))\,.$$

Note that this implies $(\forall \vec{y})\,((\forall x)A(x,\vec{y}) \leftrightarrow A(h_{\forall xA}(\vec{y}),\vec{y}))$. The Herbrand function can also be thought of as a 'counterexample function'; in that $(\forall x)A(x)$ is false if and only if $h_{\forall xA}$ provides a value x which is a counterexample to the truth of $(\forall x)A$.

Definition. Let $A(\vec{c})$ be a formula in prenex form. The *Herbrandization*, $A^H(\vec{c})$, of A is the formula defined inductively by:

(1) If $A(\vec{c})$ is quantifier-free, then $A^H(\vec{c})$ is $A(\vec{c})$.

(2) If $A(\vec{c})$ is $(\exists y)B(\vec{c},y)$, then $A^H(\vec{c})$ is the formula $(\exists y)B^H(\vec{c},y)$.

(3) If $A(\vec{c})$ is $(\forall y)B(\vec{c},y)$, then $A^H(\vec{c})$ is $B^H(\vec{c},h_A(\vec{c}))$, where h_A is the Herbrand function for A.

It is not hard to see that $A \vDash A^H$. Note that A^H is purely existential.

Proposition. *Let T be a set of prenex formulas and A any prenex formula. Then the following are equivalent:*
(1) $T \vDash A$,
(2) $T^S \vDash A$,
(3) $T \vDash A^H$,
(4) $T^S \vDash A^H$,

This proposition is easily proved from the above definitions and remarks. The importance of the proposition lies in the fact that T^S is a universal theory and that A^H is an existential formula, and that therefore Herbrand's theorem applies to $T^S \vDash A^H$. Therefore, Herbrand's theorem can be applied to an arbitrary logical implication $T \vDash A$, at the cost of converting formulas to prenex form and introducing Herbrand and Skolem functions.

2.5.3. A generalized Herbrand's theorem. Herbrand actually proved a more general version of Theorem 2.5.1 which applied directly whenever $\vDash A$, for A a general formula, not necessarily existential. His result avoids the use of Skolem/Herbrand functions but is somewhat more difficult to state and comprehend. The theorem we state next is a generalization of Herbrand's theorem which is quite similar in spirit and power to the theorem as stated originally by Herbrand [1930].

In this section, we shall consider a first-order formula A such that $\vDash A$. Without loss of generality, we shall suppose that the propositional connectives in A are restricted to be $\land$, $\lor$ and $\neg$, and that the $\neg$ connective appears only in front of atomic subformulas of A. (The only reason for this convention is that it avoids having to keep track of whether quantifiers appear positively and negatively in A.)

Definition. Let A satisfy the above convention on negations. An $\lor$*-expansion* of A is any formula that can be obtained from A by a finite number of applications of the following operation:

(α) If B is a subformula of an $\lor$-expansion A' of A, replacing B in A' with $B \lor B$ produces another $\lor$-expansion of A.

A *strong* $\lor$*-expansion* of A is defined similarly, except that now the formula B is restricted to be a subformula with outermost connective an existential quantifier.

Definition. Let A be a formula. A *prenexification* of A is a formula obtained from A by first renaming bound variables in A so that no variable is quantified more than once in A and then using prenex operations to put the formula in prenex normal form.

Note that there will generally be more than one prenexification of A since prenex operations may be applied in different orders resulting in a different order of the quantifiers in the prenex normal form formula.

Definition. Let A be a valid first-order formula in prenex normal form, with no variable quantified twice in A. If A has $r \geq 0$ existential quantifiers, then A is of the following form with B quantifier-free:

$$(\forall x_1 \cdots x_{n_1})(\exists y_1)(\forall x_{n_1+1} \cdots x_{n_2})(\exists y_2) \cdots (\exists y_r)(\forall x_{n_r+1} \cdots x_{n_{r+1}})B(\vec{x}, \vec{y})$$

with $0 \leq n_1 \leq n_2 \leq \cdots \leq n_{r+1}$. A *witnessing substitution* for A is a sequence of terms (actually, semiterms) $t_1, \ldots t_r$ such that (1) each t_i contains arbitrary free variables but only bound variables from $x_1, \ldots, x_{n_i}$ and (2) the formula $B(\vec{x}, t_1, \ldots, t_r)$ is a quasitautology (i.e., a tautological consequence of instances of equality axioms only). In the case where B does not contain the equality sign, then (2) is equivalent to B being a tautology.

Let T be a first-order theory. A sequence of terms is said to *witness A over T* if the above conditions hold except with condition (2) replaced by the weaker condition that $T \vDash (\forall \vec{x})B(x, \vec{t})$.

Definition. A *Herbrand proof* of a first-order formula A consists of a prenexification A^* of a strong $\vee$-expansion of A plus a witnessing substitution σ for A^*.

A *Herbrand T-proof* of A consists of a prenexification A^* of a strong $\vee$-expansion of A plus a substitution which witnesses A over T.

We are now in a position to state the general form of Herbrand's theorem:

Theorem. *A first-order formula A is valid if and only if A has a Herbrand proof. More generally, if T is a universal theory, then $T \vDash A$ if and only if A has a Herbrand T-proof.*

Proof. We shall sketch a proof of only the first part of the theorem since the proof of the second part is almost identical. Of course it is immediate from the definitions that if A has a Herbrand proof, then A is valid. So suppose A is valid, and therefore has a cut-free LK-proof P. We shall modify P in stages so as to extract a Herbrand proof of P.

The first stage will involve restricting the formulas which can be combined by a contraction inference. One problem that arises in this regard is that inferences with two hypothesis, such as $\vee$*:left* and $\wedge$*:right*, contain implicit contractions on side formulas. To avoid dealing with this complication, we modify the rules of inference so that no implicit contractions occur; e.g., the $\vee$*:left* inference rule is replaced by the rule

$$\frac{A, \Gamma \longrightarrow \Delta \qquad B, \Gamma' \longrightarrow \Delta'}{A \vee B, \Gamma, \Gamma' \longrightarrow \Delta, \Delta'}$$

and the $\wedge$*:right* and $\supset$*:left* are modified analogously. It is easy to see that changing the rules of inference in this way, makes no difference to what strong inferences are needed in a proof: instead, it merely makes contractions explicit. In particular, the cut elimination theorems still hold with the new inference rules.

A contraction inference is said to be a *propositional contraction* (resp., an $\exists$-*contraction*) provided that the principal formula of the contraction is quantifier-free (resp., its outermost connective is an existential quantifier). The first step in modifying P is to form a cut-free proof P_1, also with endsequent $\longrightarrow A$ such that all contraction inferences in P_1 are propositional or $\exists$-contractions. The construction of P_1 from P is formally done by an induction process analogous to the proof of the Cut-elimination Theorem 2.4.2. Namely, define the *E-depth* of a formula similarly to the definition of *depth* in 2.4.1, except define the E-depth of a quantifier-free formula or of a formula with outermost connective an existential quantifier to be zero. Then prove, by double induction on the maximum E-depth d of contraction formulas and the number of contractions of formulas of this maximum E-depth, that P_1 can be transformed into a proof in which all contractions are on formulas of E-depth zero. The induction step consists of removing a topmost contraction inference of the maximum E-depth d. For example, suppose that the following inference is a topmost contraction with principal formula of E-depth d;

$$\dfrac{\begin{array}{c}\ddots\vdots\therefore R\\ \Gamma \longrightarrow \Delta, (\forall x)B, (\forall x)B\end{array}}{\Gamma \longrightarrow \Delta, (\forall x)B}$$

Since P_1 is w.l.o.g. in free variable normal form and since this is a topmost contraction of E-depth d, we can modify the subproof R of P_1 by removing at most two $\forall$*:right* inferences and/or changing some *Weakening:right* inferences to get a proof of $\Gamma \longrightarrow \Delta, B(a), B(a')$, where a and a' are free variables not appearing in the endsequent of R. Further replacing a' everywhere by a gives a proof of $\Gamma \longrightarrow \Delta, B(a), B(a)$: we use this get to a proof ending:

$$\dfrac{\dfrac{\begin{array}{c}\ddots\vdots\therefore\\ \Gamma \longrightarrow \Delta, B(a), B(a)\end{array}}{\Gamma \longrightarrow \Delta, B(a)}}{\Gamma \longrightarrow \Delta, (\forall x)B}$$

Thus we have reduced the E-depth of the contraction inference. A similar procedure works for contractions of E-depth d with outermost connective a propositional connective—we leave the details to the reader. Note that the construction of P_1 depends on the fact that propositional inferences and $\forall$*:right* inferences can be pushed downward in the proof. It is not generally possible to push $\exists$*:right* inferences downward in a proof without violating eigenvariable conditions.

The second step in modifying P is to convert P_1 into a cut-free proof P_2 of some strong $\vee$-expansion A' of A such that every contraction in P_2 is propositional. This is done by the simple expedient of replacing every $\exists$-contraction in P_1 with an $\vee$*:right* inference, and then making the induced changes to all descendents of the principal formula of the inference. More precisely, starting with a lowermost $\exists$-contraction in P_1, say

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

replace this with an $\vee$*:left* inference

$$\frac{\Gamma \longrightarrow \Delta, (\exists x)B, (\exists x)B}{\Gamma \longrightarrow \Delta, (\exists x)B \vee (\exists x)B}$$

and then, in order to get a syntactically correct proof, replace, as necessary, subformulas $(\exists x)B'$ of formulas in P with $(\exists x)B' \vee (\exists x)B'$ (we use the notation B' since terms in B may be different in the descendents). Iterating this process yields the desired proof P_2 of a strong $\vee$-expansion A' of A. By renaming bound variables in P_2 we can assume w.l.o.g. that no variable is quantified twice in any single sequent in P_2.

Thirdly, from P_2 we can construct a prenexification A^* of A' together with a witnessing substitution, thereby obtaining a Herbrand proof of A. To do this, we iterate the following procedure for pulling quantifiers to the front of the proved formula. Find any lowest quantifier inference in P_2 which has not already been handled: this quantifier inference corresponds to a unique quantifier, (Qx), appearing in the endsequent of the proof (and conversely, each quantifier in the endsequent of the proof corresponds to a unique quantifier inference, since all contraction formulas are quantifier-free). Use prenex operations to pull (Qx) as far to the front of the endsequent formula as possible (but not past the quantifiers that have already been moved to the front of the endsequent formula). Also, push the quantifier inference downward in the proof until it reaches the group of quantifier inferences that have already been pushed downward in the proof. It is straightforward to check that this procedure preserves the property of having a syntactically valid proof. When we are done iterating this procedure, we obtain a proof P_3 of a prenexification $\longrightarrow A^*$ of A. It remains to define a witnessing substitution for A^*, which is now easy: for each existential quantifier $(\exists y_i)$ in A^*, find the corresponding $\exists$*:right* inference

$$\frac{\Gamma \longrightarrow \Delta, B(t_i)}{\Gamma \longrightarrow \Delta, (\exists y_i)B(y_i)}$$

and let the term t_i be from this inference. That this is a witnessing substitution for A^* is easily proved by noting that by removing the $\exists$*:right* inference from P_3, a proof of $A^*_M(\vec{x}, \vec{t})$ is obtained where A^*_M is the quantifier-free portion of A^*. $\Box$

The above theorem can be used to obtain the following 'no-counterexample interpretation' which has been very useful recently in the study of bounded arithmetic (see Krajíček, Pudlák and Takeuti [1991], or section **??** of Chapter II).[5]

Corollary. *Let T be a universal theory and suppose $T \vDash (\exists x)(\forall y)A(x, y, \vec{c})$ with A a quantifier-free formula. Then there is a $k > 0$ and terms $t_1(\vec{c})$, $t_2(\vec{c}, y_1)$, $t_3(\vec{c}, y_1, y_2), \ldots, t_k(\vec{c}, y_1, \ldots y_{k-1})$ such that*

$$T \vDash (\forall y_1)[A(t_1(\vec{c}), y_1, \vec{c}) \vee (\forall y_2)[A(t_2(\vec{c}, y_1), y_2, \vec{c}) \vee \\ (\forall y_3)[A(t_3(\vec{c}, y_1, y_2), y_3, \vec{c}) \vee \cdots \vee (\forall y_k)[A(t_k(\vec{c}, y_1, \ldots, y_{k-1}), y_k, \vec{c}))] \cdots]]]$$

[5]This corollary is named after the more sophisticated no-counterexample interpretations of Kreisel [1951,1952].

To prove the corollary, note that the only strong $\lor$-expansions of A are formulas of the form $\bigvee(\exists x)(\forall y)A(x,y,\vec{c})$ and apply the previous theorem.

2.5.4. No recursive bounds on number of terms. It is interesting to ask whether it is possible to bound the value of r in Herbrand's Theorem 2.5.1. For this, consider the special case where the theory T is empty, so that we have an LK-proof P of $(\exists x_1,\ldots,x_k)B(\vec{a},\vec{x})$ where B is quantifier-free. There are two ways in which one might wish to bound the number r needed for Herbrand's theorem: as a function of the size of P, or alternatively, as a function of the size of the formula $(\exists\vec{x})B$. For the first approach, it follows immediately from Theorem 2.4.3 and the proof of Herbrand's theorem, that $r \le 2^{||P||}_{2||P||}$. For the second approach, we shall sketch a proof below that r can not be recursively bounded as a function of the formula $(\exists\vec{x})B$.

To show that r cannot be recursively bounded as a function of $(\exists\vec{x})B$, we shall prove that having a recursive bound on r would give a decision procedure for determining if a given existential formula is valid. Since it is well known that validity of existential first-order formulas is undecidable, this implies that r cannot be recursively bounded in terms of the formula size.

What we shall show is that, given a formula B as in Theorem 2.5.1 and given an $r > 0$, it is decidable whether there are terms $t_{1,1},\ldots,t_{r,k}$ which make the formula

$$\bigvee_{i=1}^{r} B(\vec{a}, t_{i,1}, \ldots, t_{i,k}) \tag{1}$$

a tautology.[6] This will suffice to show that r cannot be recursively bounded. The quantifier-free formula B is expressible as a Boolean combination $C(D_1,\ldots,D_\ell)$ where each D_i is an atomic formula and $C(\cdots)$ is a propositional formula. If the formula (1) is a tautology, it is by virtue of certain formulas $D_i(\vec{a},t_{i,1},\ldots,t_{i,k})$ being identical. That is to say there is a finite set X of equalities of the form

$$D_i(\vec{a}, t_{i,1}, \ldots, t_{i,k}) \;=\; D_{i'}(\vec{a}, t_{i',1}, \ldots, t_{i',k})$$

such that, any set of terms $t_{1,1},\ldots,t_{r,k}$ which makes all the equalities in X true will make (1) a tautology.

But now the question of whether there exist terms $t_{1,1},\ldots,t_{r,k}$ which satisfy such a finite set X of equations is easily seen to be a first-order unification problem, as described in section 2.6.1 below. This means that there is an algorithm which can either determine that no choice of terms will satisfy all the equations in X or will find a most general unifier which specifies all possible ways to satisfy the equations of X.

Since, for a fixed $r > 0$, there are only finitely many possible sets X of equalities, we have the following algorithm for determining if there are terms which make (1) a tautology: for each possible set X of equalities, check if it has a solution (i.e., a

[6] This was first proved by Herbrand [1930] by the same argument that we sketch here.

most general unifier), and if so, check if the equalities are sufficient to make (1) a tautology. □

2.5.5. Interpolation theorem

Suppose we are given two formulas A and B such that $A \supset B$ is valid. An *interpolant* for A and B is a formula C such that $A \supset C$ and $C \supset B$ are both valid. It is a surprising, and fundamental, fact that it is always possible to find an interpolant C such that C contains only non-logical symbols which occur in both A and B.

We shall assume for this section that first-order logic has been augmented to include the logical symbols $\top$ and $\bot$. For this, the sequent calculus has two new initial sequents $\longrightarrow \top$ and $\bot \longrightarrow$. We write $L(A)$ to denote the set of non-logical symbols occurring in A plus all free variables occurring in A, i.e., the constant, symbols, function symbols, predicate symbols and free variables used in A. For Π a cedent, $L(\Pi)$ is defined similarly.

Craig's Interpolation Theorem. Craig [1957a].

(a) *Let A and B be first-order formulas such that $\models A \supset B$. Then there is a formula C such that $L(C) \subseteq L(A) \cap L(B)$ and such that $\models A \supset C$ and $\models C \supset B$.*

(b) *Suppose $\Gamma_1, \Gamma_2 \longrightarrow \Delta_1, \Delta_2$ is a valid sequent. Then there is a formula C such that $L(C)$ is a subset of $L(\Gamma_1, \Delta_1) \cap L(\Gamma_2, \Delta_2)$ and such that $\Gamma_1 \longrightarrow \Delta_1, C$ and $C, \Gamma_2 \longrightarrow \Delta_2$ are both valid.*

Craig's interpolation can be proved straightforwardly from the cut elimination theorem. We shall outline some of the key points of the proof, but leave a full proof to the reader. First it is easy to see that part (a) is just a special case of (b), so it suffices to prove (b). To prove (b), we first use the cut elimination theorem to obtain a cut-free proof P of $\Gamma_1, \Gamma_2 \longrightarrow \Delta_1, \Delta_2$. We then prove by induction on the number of strong inferences in P that there is a formula C with only the desired non-logical symbols and there are proofs P_1 and P_2 of $\Gamma_1 \longrightarrow \Delta_1, C$ and $C, \Gamma_2 \longrightarrow \Delta_2$. In fact, the proofs P_1 and P_2 are also cut-free and have lengths linearly bounded by the length of P. For an example of how the proof by induction goes, let's make the simplifying assumption that there are no function symbols in our languages, and then assume that the final strong inference of P is an $\exists$*:right* inference with principal formula in Δ_2. That is to say, suppose P ends with the inference

$$\dfrac{\dfrac{\ddots\vdots\,\cdot^{\cdot^{\cdot}}}{\Gamma_1, \Gamma_2 \longrightarrow \Delta_1, \Delta_2', A(t)}}{\Gamma_1, \Gamma_2 \longrightarrow \Delta_1, \Delta_2', (\exists x)A(x)}$$

with Δ_2 is the cedent $\Delta_2', (\exists x)A(x)$. Since we are assuming there are no function symbols, t is just a free variable or a constant symbol. The induction hypothesis states that there is an interpolant $C(t)$ with an appropriate first-order language such that $\Gamma_1 \longrightarrow \Delta_1, C(t)$ and $C(t), \Gamma_2 \longrightarrow \Delta_2, A(t)$ are LK-provable. The interpolant C^* for the endsequent of P is defined as follows: if the symbol t does not appear in the sequent $\Gamma_2 \longrightarrow \Delta_2$, then C^* is $(\exists y)C(y)$; otherwise, if the symbol t does not appear in the sequent $\Gamma_1 \longrightarrow \Delta_1$, then C^* is $(\forall y)C(y)$; and if t appears in both sequents, C^* is just C. It can be checked that in all three cases, the sequents $\Gamma_1 \longrightarrow \Delta_1, C^*$ and $C^*, \Gamma_2 \longrightarrow \Delta_2, (\exists x)A(x)$ are LK-provable. Therefore, C^* is an interpolant for the endsequent of P; also, it is obvious that the language $L(C)$ of C is still appropriate for the endsequent.

Secondly, suppose P ends with the inference

$$\dfrac{\dfrac{\ddots\vdots\;\cdot^{\cdot^{\cdot}}}{\Gamma_1, \Gamma_2 \longrightarrow \Delta_1, \Delta_2', A(b)}}{\Gamma_1, \Gamma_2 \longrightarrow \Delta_1, \Delta_2', (\forall x)A(x)}$$

with the principal formula still presumed to be in Δ_2. The induction hypothesis states that there is an interpolant C with an appropriate first-order language such that $\Gamma_1 \longrightarrow \Delta_1, C(b)$ and $C(b), \Gamma_2 \longrightarrow \Delta_2, A(b)$. Since, by the eigenvariable condition, b does not occur except as indicated; we get immediately LK-proofs of the sequents $\Gamma_1 \longrightarrow \Delta_1, (\forall y)C(y)$ and $(\forall y)C(y), \Gamma_2 \longrightarrow \Delta_2, (\forall x)A(x)$. Therefore, $(\forall y)C(y)$ serves as an interpolant for the endsequent of P.

There are a number of other cases that must be considered, depending on the type of the final strong inference in P and on whether its principal formula is (an ancestor of) a formula in Δ_1, Δ_2, Γ_1 or Γ_2. These cases are given in full detail in textbooks such as Takeuti [1987] or Girard [1987b].

It remains to consider the case where there are function symbols in the language. The usual method of handling this case is to just reduce it to the case where there are no function symbols by removing function symbols in favor of predicate symbols which define the graphs of the functions. Alternatively, one can carry out directly a proof on induction on the number of strong inferences in P even when function symbols are present. This involves a more careful analysis of the 'flow' of terms in the proofs, but still gives cut-free proofs P_1 and P_2 with size linear in the size of P. We leave the details of the function-symbol case to the reader.

Other interpolation theorems. A useful strengthening of the Craig interpolation theorem is due to Lyndon [1959]. This theorem states that Craig's interpolation theorem may be strengthened by further requiring that every predicate symbol which occurs positively (resp., negatively) in C also occurs positively (resp., negatively) in both A and B. The proof of Lyndon's theorem is identical to the proof sketched above, except that now one keeps track of positive and negative occurrences of predicate symbols.

Craig [1957b] gives a generalization of the Craig interpolation theorem which applies to interpolants of cedents.

Lopez-Escobar [1965] proved that the interpolation theorem holds for some infinitary logics. Barwise [1975,§III.6] proved that the interpolation theorem holds for a wider class of infinitary logics. Lopez-Escobar's proof was proof-theoretic, based on a sequent calculus formalization of infinitary logic. Barwise's proof was model-theoretic; Feferman [1968] gives a proof-theoretic treatment of these general interpolation theorems, based on the sequent calculus.

2.5.6. Beth's definability theorem

Definition. Let P and P' be predicate symbols with the same arity. Let $\Gamma(P)$ be an arbitrary set of first-order sentences not involving P', and let $\Gamma(P')$ be the same set of sentences with every occurrence of P replaced with P'.

The set $\Gamma(P)$ is said to *explicitly define* the predicate P if there is a formula $A(\vec{c})$ such that

$$\Gamma(P) \vdash (\forall\vec{x})(A(\vec{x}) \leftrightarrow P(\vec{x})).$$

The set $\Gamma(P)$ is said to *implicitly define* the predicate P if

$$\Gamma(P) \cup \Gamma(P') \vDash (\forall\vec{x})(P(\vec{x}) \leftrightarrow P'(\vec{x})).$$

The Definability Theorem of Beth [1953] states the fundamental fact that the notions of explicit and implicit definability coincide. One way to understand the importance of this is to consider *implicit* definability of P as equivalent to being able to uniquely characterize P. Thus, Beth's theorem states, loosely speaking, that if a predicate can be uniquely characterized, then it can be explicitly defined by a formula not involving P.

One common, elementary mistake is to confuse implicit definability by a set of sentences $\Gamma(P)$ with implicit definability in a particular model. For example, consider the theory T of sentences which are true in the standard model $(\mathbb{N}, 0, S, +)$ of natural numbers with zero, successor and addition. One might attempt to implicitly define multiplication in terms of zero and addition letting $\Gamma(M)$ be the theory

$$T \cup \{(\forall x)(M(x,0)=0), (\forall x)(\forall y)(M(x,S(y)) = M(x,y)+x)\}$$

It is true that this uniquely characterizes the multiplication function $M(x,y)$ in the sense that there is only one way to expand $(\mathbb{N}, 0, S, +)$ to a model of $\Gamma(M)$; however, this is not an implicit definition of M since there are nonstandard models of T which have more than one expansion to a model of $\Gamma(M)$.

Beth's Definability Theorem. $\Gamma(P)$ *implicitly defines* P *if and only if it explicitly defines* P.

Proof. Beth's theorem is readily proved from the Craig interpolation theorem as follows. First note that if P is explicitly definable, then it is clearly implicitly definable. For the converse, assume that P is implicitly definable. By compactness, we may assume without loss of generality that $\Gamma(P)$ is a single sentence. Then we have that

$$\Gamma(P) \land P(\vec{c}) \vDash \Gamma(P') \supset P'(\vec{c}).$$

By the Craig Interpolation Theorem, there is a interpolant $A(\vec{c})$ for $\Gamma(P) \land P(\vec{c})$ and $\Gamma(P') \supset P'(\vec{c})$. This interpolant is the desired formula explicitly defining P. □

It is also possible to prove the Craig Interpolation Theorem from the Beth Definability Theorem. In addition, both theorems are equivalent to the model-theoretic Joint Consistency Theorem of Robinson [1956].

2.6. First-order logic and resolution refutations

The importance of the resolution proof method for propositional logic (described in section 1.3) lies in large part in the fact that it also serves as a foundation for theorem-proving in first-order logic. Recall that by introducing Herbrand and Skolem functions, theorem-proving in first-order logic can be reduced to proving Π_2-formulas of the form $(\forall\vec{x})(\exists\vec{y})A(\vec{x},\vec{y})$ with A quantifier-free (see §2.5.2). Also, by Herbrand's Theorem 2.5.1, the problem of proving $(\forall\vec{x})(\exists\vec{y})A(\vec{x},\vec{y})$ is reducible to the problem of finding terms $\vec{r}_1,\vec{r}_2,\ldots,\vec{r}_k$ so that $\bigvee_i A(\vec{x},\vec{r}_i)$ is tautologically valid. Now, given the terms $\vec{r}_1,\ldots,\vec{r}_k$, determining tautological validity is 'merely' a problem in propositional logic; and hence is amenable to theorem proving methods such as propositional resolution. Thus one hopes that if one had a good scheme for choosing terms $\vec{r}_1,\ldots,\vec{r}_k$, then one could have a reasonable method of first-order theorem proving.

This latter point is exactly the problem that was solved by Robinson [1965b]; namely, he introduced the resolution proof method and showed that by using a unification algorithm to select terms, the entire problem of which terms to use could be solved efficiently by using the "most general" possible terms. In essence, this reduces the problem of first-order theorem proving to propositional theorem proving. (Of course, this last statement is not entirely true for two reasons: firstly, there may be a very large number (not recursively bounded) of terms that are needed, and secondly, it is entirely possible that foreknowledge of what terms are sufficient, might help guide the search for a propositional proof.)

2.6.1. Unification. We shall now describe the unification algorithm for finding most general unifiers. We shall let t denote a term containing function symbols, constant symbols and variables. A *substitution*, σ, is a partial map from variables to terms; we write $x\sigma$ to denote $\sigma(x)$, and when x is not in the domain of σ, we let $x\sigma$ be x. If σ is a substitution, then $t\sigma$ denotes the result of simultaneously replacing every variable x in t with $x\sigma$. We extend σ to atomic relations by letting $R(t_1,\ldots,t_k)\sigma$ denote $R(t_1\sigma,\ldots,t_k\sigma)$. We use concatenation $\sigma\tau$ to denote the substitution which is equivalent to an application of σ followed by an application of τ.

Definition. Let $A_1,\ldots,A_k$ be atomic formulas. A *unifier* for the set $\{A_1,\ldots,A_k\}$ is a substitution σ such that $A_1\sigma = A_2\sigma = \cdots = A_k\sigma$, where $=$ represents the property of being the identical formula.

A substitution is said to be a *variable renaming* substitution, if the substitution maps variables only to terms which are variables.

A unifier σ is said to be a *most general unifier* if, for every unifier τ for the same set, there is a unifier ρ such that $\tau = \sigma\rho$. Note that up to renaming of variables, a most general unifier must be unique.

Unification Theorem. *If $\{A_1, \ldots, A_k\}$ has a unifier then it has a most general unifier.*

Proof. We shall prove the theorem by outlining an efficient algorithm for determining whether a unifier exists and, if so, finding a most general unifier. The algorithm is described as an iterative procedure which, at stage s has a set E_s of equations and a substitution σ_s. The equations in E_s are of the form $\alpha \doteq \beta$ where α and β may be formulas or terms. The meaning of this equation is that the sought-for most general unifier must be a substitution which makes α and β identical. Initially, E_0 is the set of $k-1$ equations $A_j \doteq A_{j+1}$ and σ_0 is the identity. Given E_s and σ_s, the algorithm does any one of the following operations to choose E_{s+1} and σ_{s+1}:

(1) If E_s is empty, we are done and σ_s is a most general unifier.

(2) If E_s contains an equation of the form
$$F(t_1, \ldots, t_i) \doteq F(t'_1, \ldots, t'_i),$$
then $\sigma_{s+1} = \sigma_s$ and E_{s+1} is obtained from E_s by removing this equation and adding the i equations $t_i \doteq t'_i$. Here F is permitted to be a function symbol, a constant symbol or a predicate symbol.

(3) Suppose E_s contains an equation of the form
$$x \doteq t \qquad \text{or} \qquad t \doteq x,$$
with x a variable and t a term. Firstly, if t is equal to x, then this equation is discarded, so E_{s+1} is E_s minus this equation and $\sigma_{s+1} = \sigma_s$. Secondly, if t is a non-trivial term in which x occurs, then the algorithm halts outputting that no unifier exists.[7] Thirdly, if x does not occur in t, then let $[x/t]$ denote the substitution that maps x to t and define E_{s+1} to be the set of equations $s[x/t] \doteq s'[x/t]$ such that $s \doteq s'$ is in E_s, and define $\sigma_{s+1} = \sigma_s[x/t]$.
We leave it to the reader to prove that this algorithm always halts with a most general unifier if a unifier exists. The fact that the algorithm does halt can be proved by noting that each iteration of the algorithm either reduces the number of distinct variables in the equations, or reduces the sum of the lengths of the terms occurring in the equations. □

The algorithm as given above is relatively efficient; however, the sizes of the terms involved may grow exponentially large. Indeed, there are unification problems where the size of the most general unifier is exponential in the size of the unification problem; for example, the most general unifier for $\{f(x_1, x_2, \ldots x_k), f(g(x_2, x_2), \ldots, g(x_{k+1}, x_{k+1}))\}$ maps x_1 to a term of height k with 2^k atoms.

[7] This failure condition is known as the *occurs check*.

If one is willing to use a dag (directed acyclic graph) representation for terms, then this exponential growth rate does not occur. Paterson and Wegman [1978] have given an efficient linear-time unification algorithm based on representing terms as dags.

2.6.2. Resolution and factoring inferences. We now describe the resolution inference used by Robinson [1965b] for first-order logic. The starting point is Herbrand's theorem: we assume that we are attempting to prove a first-order sentence, which without loss of generality is of the form $(\exists\vec{x})A(\vec{y})$. (This may be assumed without loss of generality since, if necessary, Herbrand functions may be introduced.) We assume, in addition, that A is in disjunctive normal form. Instead of proving this sentence, we shall instead attempt to refute the sentence $(\forall\vec{x})(\neg A(x))$. Since A is in disjunctive normal form, we may view $\neg A$ as a set Γ_A of clauses with the literals in the clauses being atomic formulas or negated atomic formulas. We extend the definition of substitutions to act on clauses in the obvious way so that $\{C_1,\ldots,C_k\}\sigma$ is defined to be $\{C_1\sigma,\ldots,C_k\sigma\}$.

When we refute $(\forall\vec{x})(\neg A)$, we are showing that there is no structure M which satisfies $(\forall\vec{x})(\neg A)$. Consider a clause C in Γ_A. The clause C is a set of atomic and negated atomic formulas, and a structure M is said to *satisfy* C provided it satisfies $(\forall\vec{x})(\bigvee_{\phi\in C}\phi(x))$. Thus, it is immediate that if M satisfies C, and if σ is a substitution, then M also satisfies $C\sigma$. From this, we see that the following version of resolution is sound in that it preserves the property of being satisfied by a model M: If B and C are clauses, if ϕ is an atomic formula and if σ and τ are substitutions, let D be the clause $(B\sigma\setminus\{\phi\})\cup(C\tau\setminus\{\overline{\phi}\})$. It is easy to verify that if B and C are satisfied in M, then so is D.

Following Robinson [1965b], we use a restricted form of this inference principle as the sole rule of inference for first-order resolution refutations:

Definition. Let B and C be clauses and suppose $P(\vec{s}_1), P(\vec{s}_2), \ldots P(\vec{s}_k)$ are atomic formulas in B and that $\neg P(\vec{t}_1), \neg P(\vec{t}_2), \ldots \neg P(\vec{t}_\ell)$ are negated atomic formulas in C. Choose a variable renaming substitution τ so that $C\tau$ has no variables in common with B. Also suppose that the $k+\ell$ formulas $P(\vec{s}_i)$ and $P(\vec{t}_i)\tau$ have a most general unifier σ. Then the clause D defined by

$$(B\sigma\setminus\{P(\vec{s}_1)\sigma\})\cup(C\tau\sigma\setminus\{\neg P(\vec{s}_1)\tau\sigma\})$$

is defined to be an *R-resolvent* of B and C.

The reason for using the renaming substitution τ is that the variables in the clauses B and C are implicitly universally quantified; thus if the same variable occurs in both B and C we allow that variable to be instantiated in B by a different term than in C when we perform the unification. Applying τ before the unification allows this to happen automatically.

One often views R-resolution as the amalgamation of two distinct operations: first, the *factoring* operation finds a most general unifier of a subset of clause, and second, the unitary resolution operation which resolves two clauses with respect to a single literal. Thus, R-resolution consists of (a) choosing subsets of the clauses B and C and factoring them, and then (b) applying resolution w.r.t. to the literal obtained by the factoring.

Completeness of R-resolution. *A set Γ of first-order clauses is unsatisfiable if and only if the empty clause can be derived from Γ by R-resolution.*

This theorem is proved by the discussion in the next paragraph.

2.6.3. Lifting ground resolution to first-order resolution. A *ground literal* is defined to be a literal in which no variables occur; a *ground clause* is a set of ground literals. We assume, with no loss of generality, that our first-order language contains a constant symbol and that therefore ground literals exist. Ground literals may independently be assigned truth values[8] and therefore play the same role that literals played in propositional logic. A *ground* resolution refutation is a propositional-style refutation involving ground clauses only, with ground literals in place of propositional literals. By the Completeness Theorem 1.3.4 for propositional resolution, a set of ground clauses is unsatisfiable if and only if it has a ground resolution refutation.

For sets of ground clauses, R-resolution is identical to propositional-style resolution. Suppose, however, that Γ is an unsatisfiable set of first-order (not necessarily ground) clauses. Since Γ is unsatisfiable there is, by Herbrand's theorem, a set of substitutions $\sigma_1, \ldots, \sigma_r$ so that each $\Gamma\sigma_r$ is a set of ground clauses and so that the set $\Pi = \bigcup_i \Gamma\sigma_i$ of clauses is propositionally unsatisfiable. Therefore there is a ground resolution refutation of Π.

To justify the completeness of R-resolution, we shall show that any ground resolution refutation of Π can be 'lifted' to an R-resolution refutation of Γ. In fact, we shall prove the following: if $C_1, C_2, \ldots, C_n = \emptyset$ is a resolution refutation of Π, then there are clauses $D_1, D_2, \ldots, D_m = \emptyset$ which form an R-resolution refutation of Γ and there are substitutions $\sigma_1, \sigma_2, \ldots, \sigma_m$ so that $D_i\sigma_i = C_i$. We define D_i and σ_i by induction on i as follows. Firstly, if $C_i \in \Pi$, then it must be equal to $D_i\sigma_i$ for some $D_i \in \Gamma$ and some substitution σ_i by the definition of Π. Secondly, if C_i is inferred from C_j and C_k, with $j, k < i$ by resolution w.r.t. the literal $P(\vec{r})$, then define E_j to be the subset of D_j which is mapped to $P(\vec{r})$ by σ_j, and define E_k similarly. Now, form the R-resolution inference which factors the subsets E_j and E_k of D_j and D_k and forms the resolvent. This resolvent is D_i and it is straightforward to show that the desired σ_i exists.

That finishes the proof of the Completeness Theorem for R-resolution. It should be noted that the method of proof shows that R-resolution refutations are the shortest possible refutations, even if arbitrary substitutions are allowed for factoring inferences. Even more importantly, the method by which ground resolution refutations were 'lifted' to R-resolution refutations preserves many of the search strategies that

[8] We are assuming that the equality sign $(=)$ is not present.

were discussed in section 1.3.5. This means that these search strategies can be used for first-order theorem proving.[9]

2.6.4. Paramodulation. The above discussion of R-resolution assumed that equality was not present in the language. In the case where equality is in the language, one must either add additional initial clauses as axioms that express the equality axioms or one must add additional inference rules. For the first approach, one could add clauses which express the equality axioms from section 2.2.1; for instance the third equality axiom can be expressed with the clause

$$\{x_1 \neq y_1, ..., x_k \neq y_k, \neg P(\vec{x}), P(\vec{y})\},$$

and the other equality axioms can similarly be expressed as clauses. More computational efficiency can be obtained with equality clauses of the form

$$\{x \neq y, \overline{A(x)}, A(y)\}$$

where $A(x)$ indicates an arbitrary literal.

For the second approach, the *paramodulation* inference is used instead of equality clauses; this inference is a little complicated to define, but goes as follows: Suppose B and C are clauses with no free variables in common and that $r = s$ is a literal in C; let t be a term appearing somewhere in B and let σ be a most general unifier of r and t (or of s and t); let B' be the clause which is obtained from $B\sigma$ by replacing occurrences of $t\sigma$ with $s\sigma$ (or with $r\sigma$, respectively) and let C' be $(C \setminus \{r = s\})\sigma$. Under these circumstances, paramodulation allows $B' \cup C'$ to be inferred from B and C. Paramodulation was introduced and shown to be complete by Robinson and Wos [1969] and Wos and Robinson [1973]: for completeness, paramodulation must be combined with R-resolution, with factoring and with application of variable renaming substitutions.

2.6.5. Horn clauses. An important special case of first-order resolution is when the clauses are restricted to be Horn clauses. The propositional refutation search strategies described in section 1.3.5.6 still apply; and, in particular, an unsatisfiable set Γ of Horn clauses always has a linear refutation supported by a negative clause in Γ. In addition, the factoring portion of R-resolution is not necessary in refutations of Γ.

A typical use of Horn clause refutations is as follows: a set Δ of Horn clauses is assumed as a 'database' of knowledge, such that every clause in Δ contains a positive literal. A query, which is an atomic formula $P(s_1, ..., s_k)$, is chosen; the object is to determine if there is an instance of $P(\vec{s})$ which is a logical consequence of Δ. In other words, the object is to determine if $(\exists\vec{x})P(\vec{s})$ is a consequence of Δ where $\vec{x}$ is the vector of variables in $P(\vec{s})$. To solve this problem, one forms the clause $\{\overline{P(\vec{s})}\}$

[9]Historically, it was the desire to find strategies for first-order theorem proving and the ability to lift results from propositional theorem proving, that motivated the research into search strategies for propositional resolution.

and lets Γ be the set $\Delta \cup \{\overline{P(\vec{s})}\}$; one then searches for a linear refutation of Γ which is supported by $\overline{P(\vec{s})}$. If successful, such a linear refutation R also yields a substitution σ, such that $\Delta \vdash P(\vec{s})\sigma$; and indeed, σ is the most general substitution such that R gives a refutation of $\Delta \cup \{\overline{P(\vec{s})\sigma}\}$. From this, what one actually has is a proof of $(\forall \vec{y})(P(\vec{s})\sigma)$ where $\vec{y}$ is the vector of free variables in the terms $P(\vec{s})\sigma$. Note that there may be more than one refutation, and that different refutations can give different substitutions σ, so there is not necessarily a unique most general unifier.

What we have described is essentially a pure form of PROLOG, which is a logic programming language based on searching for refutations of Horn clauses, usually in a depth-first search. PROLOG also contains conventions for restricting the order of the proof search procedure.

For further reading. There is an extensive literature on logic programming, automated reasoning and automated theorem proving which we cannot survey here. The paper of Robinson [1965b] still provides a good introduction to the foundations of logic programming; the textbooks of Chang and Lee [1973] and Loveland [1978] provide a more detailed treatment of the subject matter above, and the textbooks of Kowalski [1979] and Clocksin and Mellish [1981] provide good detailed introductions to logic programming and PROLOG. Chapter IX, by G. Jäger and R. Stärk, discusses the proof-theory and semantics of extensions of logic programming to non-Horn clauses.

3. Proof theory for other logics

In the final section of this chapter, we shall briefly discuss two important non-classical logics, intuitionistic logic and linear logic.

3.1. Intuitionistic logic

Intuitionistic logic is a subsystem of classical logic which historically arose out of various attempts to formulate a more constructive foundation for mathematics. For example, in intuitionistic logic, the law of the excluded middle, $A \vee \neg A$, does not hold in general; furthermore, it is not possible to intuitionistically prove $A \vee B$ unless already at least one of A or B is already intuitionistically provable. We shall discuss below primarily mathematical aspects of the intuitionistic logic, and shall omit philosophical or foundational issues: the books of Troelstra and van Dalen [1988] provide a good introduction to the latter aspects of intuitionistic logic.

The intuitionistic sequent calculus, LJ, is defined similarly to the classical sequent calculus LK, except with the following modifications:

(1) To simplify the exposition, we adopt the convention that negation ($\neg$) is not a propositional symbol. In its place, we include the absurdity symbol $\perp$ in the language; $\perp$ is a nullary propositional symbol which is intended to always have value *False*. The two $\neg$ rules of LK are replaced with the single $\perp$*:left* initial sequent, namely $\perp\longrightarrow$. Henceforth, $\neg A$ is used as an abbreviation for $A \supset \perp$.

(2) In LJ, the $\lor$*:right* rule used in the definition of LK in section 1.2.2 is replaced by the two rules

$$\frac{\Gamma \longrightarrow \Delta, A}{\Gamma \longrightarrow \Delta, A \lor B} \qquad \text{and} \qquad \frac{\Gamma \longrightarrow \Delta, A}{\Gamma \longrightarrow \Delta, B \lor A}$$

(3) Otherwise, LJ is defined like LK, except with the important proviso that at most one formula may appear in the antecedent of any sequent. In particular, this means that rules which have the principal formula to the right of the sequent arrow may not have any side formulas to the right of the sequent arrow.

3.1.1. Cut elimination. An important property of intuitionistic logic is that the cut elimination and free-cut elimination theorems still apply:

Theorem.
(1) *Let $\Gamma \longrightarrow A$ be LJ-provable. Then there is a cut-free LJ-proof of $\Gamma \longrightarrow A$.*
(2) *Let $\mathfrak{S}$ be a set of sequents closed under substitution such that each sequent in $\mathfrak{S}$ has at most one formula in its antecedent. Let $LJ_{\mathfrak{S}}$ be LJ augmented with initial sequents from $\mathfrak{S}$. If $LK_{\mathfrak{S}} \vdash \Gamma \longrightarrow A$, then there is a free-cut free $LK_{\mathfrak{S}}$-proof of $\Gamma \longrightarrow A$.*

The (free) cut elimination theorem for LJ can be proved similarly to the proof-theoretic proofs used above for classical logic.

3.1.2. Constructivism and intuitionism. Intuitionistic logic is intended to provide a 'constructive' subset of classical logic: that is to say, it is designed to not allow non-constructive methods of reasoning. An important example of the constructive aspect of intuitionistic logic is the Brouwer-Heyting-Kolmogorov (BHK) constructive interpretation of logic. In the BHK interpretation, the logical connectives $\lor$, $\exists$ and $\supset$ have non-classical, constructive meanings. In particular, in order to have a proof of (or knowledge of) a statement $(\exists x)A(x)$, it is necessary to have a particular object t and a proof of (or knowledge of) $A(t)$. Likewise, in order to have a proof of $A \lor B$ the BHK interpretation requires one to have a proof either of A or of B, along with a indication of which one it is a proof of. In addition, in order to have a proof of $A \supset B$, one must have a method of converting any proof of A into a proof B. The BHK interpretation of the connectives $\land$ and $\forall$ are similar to the classical interpretation; thus, a proof of $A \land B$ or of $(\forall x)A(x)$, consists of proofs of both A and B or of a method of constructing a proof of $A(t)$ for all objects t.

It is not difficult to see that the intuitionistic logic LJ is sound under the BHK interpretation, in that any LJ-provable formula has a proof in the sense of the BHK interpretation.

The BHK interpretation provides the philosophical and historical underpinnings of intuitionistic logic; however, it has the drawback of characterizing what constitutes a valid proof rather than characterizing the meanings of the formulas directly. It is possible to extend the BHK interpretation to give meanings to formulas directly, e.g., by using realizability; under this approach, an existential quantifier $(\exists x)$ might mean "there is a constructive procedure which produces x, and (possibly) produces evidence that x is correct". This approach has the disadvantage of requiring one to pick a notion of 'constructive procedure' in an ad-hoc fashion; nonetheless it can be very fruitful in applications such a intuitionistic theories of arithmetic where there are natural notions of 'constructive procedure'.

For pure intuitionistic logic, there is a very satisfactory model theory based on Kripke models. Kripke models provide a semantics for intuitionistic formulas which is analogous to model theory for classical logic in many ways, including a model theoretic proof of the cut-free completeness theorem, analogous to the proof of Theorem 2.3.7 given above.

For an accessible account of the BHK interpretation and Kripke model semantics for intuitionistic logic, the reader can refer to Troelstra and van Dalen [1988,vol. 1]; Chapter VI contains an thorough discussion of realizability. In this section we shall give only the following theorem of Harrop, which generalizes the statements that $A \lor B$ is intuitionistically valid if and only if either A or B is, and that $(\exists x)A(x)$ is intuitionistically valid if and only if there is a term t such that $A(t)$ is intuitionistically valid.

3.1.3. Definition. The *Harrop formulas* are inductively defined by:

(1) Every atomic formula is a Harrop formula,

(2) If B is a Harrop formula and A is an arbitrary formula, then $A \supset B$ is a Harrop formula.

(3) If B is a Harrop formula, then $(\forall x)B$ is a Harrop formula.

(4) If A and B are Harrop formulas, then so is $A \land B$.

Intuitively, a Harrop formula is a formula in which all existential quantifiers and all disjunctions are 'hidden' inside the left-hand scope of an implication.

Theorem. (Harrop [1960]) *Let Γ be a cedent containing only Harrop formulas, and let A and B be arbitrary formulas.*

(a) *If $LJ \vdash \Gamma \longrightarrow (\exists x)B(x)$, then there exists a term t such that LJ proves $\Gamma \longrightarrow B(t)$.*

(b) *If $LJ \vdash \Gamma \longrightarrow A \lor B$, then at least one of $\Gamma \longrightarrow A$ and $\Gamma \longrightarrow B$ is LJ-provable.*

We omit the proof of the theorem, which is readily provable by using induction on the length of cut-free LJ-proofs.

3.1.4. Interpretation into classical logic. The 'negative translation' provides a translation of classical logic into intuitionistic logic; an immediate corollary of the negative translation is a simple, constructive proof of the consistency of classical logic from the consistency of intuitionistic logic. There are a variety of negative translations: the first ones were independently discovered by Kolmogorov, Gödel and Gentzen.

Definition. Let A be a formula. The *negative translation*, A^-, of A is inductively defined by:

(1) If A is atomic, then A^- is $\neg\neg A$,

(2) $(\neg B)^-$ is $\neg(B^-)$.

(3) $(B \land C)^-$ is $(B^-) \land (C^-)$,

(4) $(B \supset C)^-$ is $(B^-) \supset (C^-)$,

(5) $(B \lor C)^-$ is $\neg(\neg(B^-) \land \neg(C^-))$,

(6) $(\forall x B)^-$ is $\forall x(B^-)$,

(7) $(\exists x B)^-$ is $\neg\forall x(\neg(B^-))$,

Theorem. *Let A be a formula. Then $LK \vdash A$ if and only if $LJ \vdash A^-$.*

Clearly A^- is classically equivalent to A, so if A^- is intuitionistically valid, then A is classically valid. For the converse, we use the following two lemmas which immediately imply the theorem.

Lemma. *Let A be a formula. Then LJ proves $\neg\neg A^- \longrightarrow A^-$.*

Proof. The proof is by induction on the complexity of A. We will do only one of the more difficult cases. Suppose A is $B \supset C$. We need to show that LJ proves $\neg\neg(B^- \supset C^-) \longrightarrow B^- \supset C^-$, for which, it suffices to prove $B^-, \neg\neg(B^- \supset C^-) \longrightarrow C^-$. By the induction hypothesis, LJ proves $\neg\neg C^- \longrightarrow C^-$, so it will suffice to show that $B^-, \neg\neg(B^- \supset C^-), \neg C^- \longrightarrow$ is LJ-provable. To do this it suffices to show that $B^-, B^- \supset C^- \longrightarrow C^-$ is LJ-provable, which is easy to prove. □

If Γ is a cedent $B_1, \ldots, B_k$, then Γ^- denotes the cedent $B_1^-, \ldots, B_k^-$ and $\neg\Gamma^-$ denotes the cedent $\neg B_1^-, \ldots, \neg B_k^-$.

Lemma. *Suppose that LK proves $\Gamma \longrightarrow \Delta$. Then LJ proves the sequent $\Gamma^-, \neg\Delta^- \longrightarrow$.*

Proof. The proof of the lemma is by induction on the number of inferences in an LK-proof, possibly containing cuts. We'll do only one case and leave the rest to the reader. Suppose the LK-proof ends with the inference

$$\frac{B, \Gamma \longrightarrow \Delta, C}{\Gamma \longrightarrow \Delta, B \supset C}$$

The induction hypothesis is that $B^-, \Gamma^-, \neg\Delta^-, \neg C^- \longrightarrow$ is provable in LJ. By the previous lemma, the induction hypothesis implies that $B^-, \Gamma^-, \neg\Delta^- \longrightarrow C^-$ is provable, and from this it follows that $\Gamma^-, \neg\Delta^-, \neg(B^- \supset C^-) \longrightarrow$ is also LJ-provable, which is what we needed to show. □

One consequence of the proof of the above theorem is a constructive proof that the consistency of intuitionistic logic is equivalent to the consistency of classical logic; thus intuitionistic logic does not provide a better foundations for mathematics from the point of view of sheer consistency. This is, however, of little interest to a constructivist, since the translation of classical logic into intuitionistic logic does not preserve the constructive meaning (under, e.g., the BHK interpretation) of the formula.

It is possible to extend the negative translation to theories of arithmetic and to set theory; see, e.g., Troelstra and van Dalen [1988,vol. 1].

3.1.5. Formulas as types. Section 3.1.2 discussed a connection between intuitionistic logic and constructivism. An important strengthening of this connection is the Curry-Howard formulas-as-types isomorphism, which provides a direct correspondence between intuitionistic proofs and λ-terms representing computable functions, under which intuitionistic proofs can be regarded as computable functions. This section will discuss the formulas-as-types isomorphism for intuitionistic propositional logic; our treatment is based on the development by Howard [1980]. Girard [1989] contains further material, including the formulas-as-types isomorphism for first-order intuitionistic logic.

The $\{\supset\}$-fragment.. We will begin with the fragment of propositional logic in which the only logical connective is $\supset$. We work in the sequent calculus, and the only strong rules of inference are the $\supset$*:left* and $\supset$*:right* rules. Firstly we must define the set of *types*:

Definition. The *types* are defined as follows:

(a) Any propositional variable p_i is a type.

(b) If σ and τ are types, then $(\sigma \to \tau)$ is a type.

If we identify the symbols $\supset$ and $\to$, the types are exactly the same as formulas, since $\supset$ is the only logical connective allowed. We shall henceforth make this identification of types and formulas without comment.

Secondly, we must define the terms of the λ-calculus; each term t has a unique associated type. We write t^τ to mean that t is a term of type τ.

Definition. For each type τ, there is an infinite set of variables, $x_1^\tau, x_2^\tau, x_3^\tau, \ldots$ of type τ. Note that if $\sigma \neq \tau$, then x_i^σ and x_i^τ are distinct variables.

The *terms* are inductively defined by:

(a) Any variable of type τ is a term of type τ.

(b) If $s^{\sigma\to\tau}$ and t^σ are terms of the indicated types, then (st) is a term of type τ. This term may also be denoted $(st)^\tau$ or even $(s^{\sigma\to\tau}t^\sigma)^\tau$.

(c) If t^τ is a term then $\lambda x^\sigma.t$ is a term of type $\sigma \to \tau$. This term can also be denoted $(\lambda x^\sigma.t)^{\sigma\to\tau}$ or $(\lambda x^\sigma.t^\tau)^{\sigma\to\tau}$

Traditionally, a type $\sigma \to \tau$ is viewed as a set of mappings from the objects of type σ into the objects of type τ. For intuitionistic logic, it is often useful to think of a type σ as being the set of proofs of the formula σ. Note that under the BKH-interpretation this is compatible with the traditional view of types as a set of mappings.

The λx connective serves to bind occurrences of x; thus one can define the notions of bound and free variables in terms in the obvious way. A term is *closed* provided it has no free variables. The computational content of a term is defined by letting (st) represent the composition of the terms s and t, and letting $\lambda x^\sigma.t$ represent the mapping that takes objects d of type τ to the object $t(x/d)$. Clearly this gives a constructive computational meaning to terms.

Theorem. *Let B be a formula. $LJ \vdash B$ if and only if there is a closed term of type B.*

An example of this theorem is illustrated by the closed term K defined as $\lambda x.\lambda y.x$ where x and y are of type σ and τ, respectively, and therefore the term K has type $\sigma \to (\tau \to \sigma)$, which is a valid formula. A second important example is the closed term S which is defined by $\lambda x.\lambda y.\lambda z.((xz)(yz))$ where the types of x, y and z are $\sigma \to (\tau \to \mu)$, $\sigma \to \tau$ and σ, respectively, and therefore, S has type $(\sigma \to (\tau \to \mu)) \to ((\sigma \to \tau) \to (\sigma \to \mu))$ which is a valid formula. The terms K and S are examples of combinators.

The import of this theorem is that a closed term of type B corresponds to a proof of B. We shall briefly sketch the proof of this for the sequent calculus; however, the correspondence between closed terms and natural deduction proofs is even stronger, namely, the closed terms are isomorphic to natural deduction proofs in intuitionistic logic (see Girard [1989]). To prove the theorem, we prove the following lemma:

Lemma. *LJ proves the sequent $A_1, \ldots, A_n \longrightarrow B$ if and only if there is a term t^B of the indicated type involving only the free variables $x_1^{A_1}, \ldots, x_n^{A_n}$.*

Proof. It is straightforward to prove, by induction on the complexity of t^B, that the desired LJ-proof exists. For the other direction, use induction on the length of the LJ-proof to prove that the term t^B exists. We will consider only the case where the last inference of the LJ-proof is

$$\frac{\Gamma \longrightarrow A \qquad B, \Gamma \longrightarrow C}{A \supset B, \Gamma \longrightarrow C}$$

The induction hypotheses give terms $r^A(x^\Gamma)$ and $s^C(x_1^B, x^\Gamma)$ where x^Γ actually represents a vector of variables, one for each formula in Γ. The desired term $t^C(x_2^{A\supset B}, x^\Gamma)$ is defined to be $s^C((x_2^{A\supset B} r^A(x^\Gamma)), x^\Gamma)$. $\square$

The $\{\supset, \bot\}$-fragment. The above treatment of the formulas-as-types isomorphism allowed only the connective $\supset$. The formulas-as-types paradigm can be extended to allow also the symbol $\bot$ and thereby negation, by making the following changes.

Firstly, enlarge the definition of types, by adding a clause specifying that $\emptyset$ is a type. Identifying $\emptyset$ with $\bot$ makes types identical to formulas. The type $\emptyset$ corresponds to the empty set; this is consistent with the BHK interpretation since $\bot$ has no proof. Secondly, enlarge the definition of terms by adding to the definition a new clause stating that, for every type σ, there is a term $f^{\emptyset\to\sigma}$ of type $\emptyset \to \sigma$.

Now the Theorem and Lemma of section 3.1.5 still hold verbatim for formulas with connectives $\{\supset, \bot\}$ and with the new definitions of types and terms.

The $\{\supset, \bot, \land\}$-fragment. To further expand the formulas-as-types paradigm to intuitionistic logic with connectives $\supset$, $\bot$ and $\land$, we must make the following modifications to the definitions of terms and types. Firstly, add to the definition of types, a clause stating that if σ and τ are types, then $(\sigma \times \tau)$ is a type. By identifying $\times$ with $\land$, we still have the types identified with the formulas. Consistent with the BHK-interpretation, the type $(\sigma \times \tau)$ may be viewed as the cross-product of its constituent types. Secondly, add to the definition of terms the following two clauses: (*i*) if s^σ and t^τ are terms, then $\langle s, t\rangle$ is a term of type $\sigma \times \tau$, and (*ii*) if $s^{\sigma\times\tau}$ is a term, then $(\pi_1 s)^\sigma$ and $(\pi_2 s)^\tau$ are terms. The former term uses the pairing function; and the latter terms use the projection functions.

Again, the Theorem and Lemma above still holds with these new definitions of types and terms.

The $\{\supset, \bot, \land, \lor\}$-fragment. To incorporate also the connective $\lor$ into the formulas-as-types isomorphism we expand the definitions of types and terms as follows. Firstly, add to the definition of types that if σ and τ are types, then $(\sigma + \tau)$ is a type. To identify formulas with types, the symbol $+$ is identified with $\lor$. The type $\sigma + \tau$ is viewed as the disjoint union of σ and τ, consistent with the BHK-interpretation. Secondly, add to the definitions of terms the following two clauses

- If s^σ and t^τ are terms of the indicated types, then $(\iota_1^{\sigma\to(\sigma+\tau)} s)$ and $(\iota_2^{\tau\to(\sigma+\tau)} t)$ are terms of type $\sigma + \tau$.
- For all types σ, τ and μ, there is a constant symbol $d^{(\sigma\to\mu)\to((\tau\to\mu)\to((\sigma+\tau)\to\mu))}$ of the indicated type. In other words, if $s^{\sigma\to\mu}$ and $t^{\tau\to\mu}$ are terms, then (dst) is a term of type $(\sigma + \tau) \to \mu$.

Once again, the Theorem and Corollary of section 3.1.5 holds with these enlarged definitions for types and terms.

3.2. Linear logic

Linear logic was first introduced by Girard [1987a] as a refinement of classical logic, which uses additional connectives and in which weakening and contractions are not allowed. Linear logic is best understood as a 'resource logic' in that proofs in linear logic must use each formula exactly once. In this point view, assumptions in a proof are viewed as resources; each resource must be used once and only once during the proof. Thus, loosely speaking, an implication $A \vdash B$ can be proved in linear logic only if A is *exactly* what is needed to prove B; thus if the assumption A is either too weak or too strong, it is not possible to give a linear logic proof of 'if A then B'. As an example of this, $A \vdash B$ being provable in linear logic does not generally imply that $A, A \vdash B$ is provable in linear logic; this is because $A \vdash B$ is asserting that one use of the resource A gives B, whereas $A, A \vdash B$ asserts that two uses of the resource A gives B.

We shall introduce only the propositional fragment of linear logic, since already the main features of linear logic are present in its propositional fragment. We are particularly interested in giving a intuitive understanding of linear logic; accordingly, we shall frequently make vague or even not-quite-true assertions, sometimes, but not always, preceded by phrases such as "loosely speaking" or "intuitively", etc.

Linear logic will be formalized as a sequent calculus.[10] The initial logical sequents are just $A \longrightarrow A$; the weakening and contraction structural rules are not allowed and the only structural rules are the exchange rule and the cut rule. Since contraction is not permitted, this prompts one to reformulate the rules, such as $\land$*:right* and $\lor$*:left*, which contain implicit contraction of side formulas. To begin with conjunction, linear logic has two different different conjunction symbols, denoted $\otimes$ and $\&$: the *multiplicative* connective $\otimes$ does not allow contractions on side formulas, whereas the *additive* conjunction $\&$ does. The rules of inference for the two varieties of conjunction are:

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

and

$$\&\textit{:left}\,\frac{A, \Gamma \longrightarrow \Delta}{A \& B, \Gamma \longrightarrow \Delta} \qquad \&\textit{:right}\,\frac{\Gamma \longrightarrow \Delta, A \qquad \Gamma \longrightarrow \Delta, B}{\Gamma \longrightarrow \Delta, A \& B}$$

$$\&\textit{:left}\,\frac{B, \Gamma \longrightarrow \Delta}{A \& B, \Gamma \longrightarrow \Delta}$$

The obvious question now arises of what the difference is in the meanings of the two different conjunctions $\otimes$ and $\&$. For this, one should think in terms of resource usage. Consider first the $\otimes$*:left* inference: the upper sequent contains both A and B so that resources for both A and B are available. The lower sequent contains $A \otimes B$ in their stead; the obvious intuition is that the resource (for) $A \otimes B$ is equivalent to having resources for both A and B. Thus, loosely speaking, we translate $\otimes$ as meaning "both". Now consider the $\&$*:left* inference; here the lower sequent has the principal formula $A \& B$ in place of either A or B, depending on which form of the inference is used. This means that a resource $A \& B$ is equivalent to having a resource for A or B, whichever is desired. So we can translate $\&$ as "whichever".

The translations "both" and "whichever" also make sense for the other inferences for the conjunctions. Consider the $\otimes$*:right* inference and, since we have not yet discussed the disjunctions, assume Δ_1 and Δ_2 are empty. In this case, the upper hypotheses say that from a resource Γ_1 one can obtain A, and from a resource Γ_2

[10]The customary convention for linear logic is to use "$\vdash$" as the symbol for the sequent connection; we shall continue to use the symbol "$\longrightarrow$" however.

one can obtain B. The conclusion then says that from (the disjoint union of) both these resources, one obtains both A and B. Now consider a $\&$*:right* inference

$$\frac{\Gamma \longrightarrow A \qquad \Gamma \longrightarrow B}{\Gamma \longrightarrow A\&B}$$

For the lower sequent, consider an agent who has the responsibility of providing (outputting) a resource $A\&B$ using exactly the resource Γ. To provide the resource $A\&B$, the agent must be prepared to either provide a resource A or a resource B, whichever one is needed; in either case, there is an upper sequent which says the agent can accomplish this.

Linear logic also has two different forms of disjunction: the multiplicative disjunction and the additive disjunction $\oplus$. The former connective is dual to $\otimes$ and the latter is dual to $\&$. Accordingly the rule of inference for the disjunctions are:

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

and

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

$$\oplus\text{:right}\,\frac{\Gamma \longrightarrow \Delta, B}{\Gamma \longrightarrow \Delta, A \oplus B}$$

By examining these inferences, we can see that the multiplicative disjunction, , is a a kind of 'indeterminate OR' or 'unspecific OR', in that a resource for AB consists of a either a resource either for A or for B, but without a specification of which one it is a resource for. On the other hand, the additive conjunction, $\oplus$, is a 'determinate OR' or a 'specific OR', so that a resource for $A \oplus B$ is either a resource for A or a resource for B together with a specification of which one it is a resource for. To give a picturesque example, consider a military general who has to counter an invasion which will come either on the western front *or* on the eastern front. If the connective 'or' is the multiplicative or, then the general needs sufficient resources to counter both threats, whereas if the 'or' is an additive disjunction then the general needs only enough resources to counter either threat. In the latter case, the general is told ahead of time where the invasion will occur and has the ability to redeploy resources so as to counter the actual invasion; whereas in the former case, the general must separately deploy resources sufficient to counter the attack from the west and resources sufficient to counter the attack from the east. From the rules of inference for disjunction, it is clear that the commas in the succedent of a sequent intended to be interpreted as the unspecific or, .

The linear implication symbol, $\multimap$, is defined by letting $A \multimap B$ be an abbreviation for $A^\perp B$, where $A^\perp$ is defined below.

Linear logic also has two nullary connectives, $\mathbf{1}$ and $\top$, for truth and two nullary connectives, $\mathbf{0}$ and $\perp$, for falsity. Associated to these connectives are the initial sequents $\longrightarrow \mathbf{1}$ and $\mathbf{0}, \Gamma \longrightarrow \Delta$, and the one rule of inference:

$$\frac{\Gamma \longrightarrow \Delta}{\mathbf{1}, \Gamma \longrightarrow \Delta}$$

The intuitive idea for the multiplicatives is that **1** has the null resource and that there is no resource for $\bot$. For the additive connectives, $\top$ has a 'arbitrary' resource, which means that anything is a resource for $\top$, whereas any resource for **0** is a 'wild card' resource which may be used for any purpose.

In addition, linear logic has a negation operation $A^\perp$ which is involutive in that $(A^\perp)^\perp$ is the same as formula A. For each propositional variable p, $p^\perp$ is the negation of p. The operation of the negation is inductively defined by letting $(A \otimes B)^\perp$ be $A^\perp B^\perp$, $(A\&B)^\perp$ be $A^\perp \oplus B$, $\mathbf{1}^\perp$ be $\bot$, and $\top^\perp$ be $\mathbf{0}$. This plus the involution property defines the operation $A^\perp$ for all formulas A. The two rules of inference associated with negation are

$$^\perp\textit{:right}\ \frac{A, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, A^\perp} \qquad \text{and} \qquad ^\perp\textit{:left}\ \frac{\Gamma \longrightarrow \Delta, A}{A^\perp, \Gamma \longrightarrow \Delta}$$

Definition. The multiplicative/additive fragment of propositional linear logic is denoted **MALL** and contains all the logical connectives, initial sequents and rules of inference discussed above.

The absence of weakening and contraction rules, makes the linear logic sequent calculus particularly well behaved; in particular, the cut elimination theorem for **MALL** is quite easy to prove. In fact, if P is a **MALL** proof, then there is a *shorter* **MALL** proof P^* with the same endsequent which contains no cut inferences. In addition, the number of strong inferences in P^* is bounded by the number of connectives in the endsequent. Bellin [1990] contains an in-depth discussion of **MALL** and related systems.

As an exercise, consider the distributive laws $A \otimes (B \oplus C) \longrightarrow (A \otimes B) \oplus (A \otimes B)$ and $(A \otimes B) \oplus (A \otimes B) \longrightarrow A \otimes (B \oplus C)$. The reader should check that these are valid under the intuitive interpretation of the connectives given above; and, as expected, they can be proved in **MALL**. Similar reasoning shows that & is distributive over . On the other hand, $\oplus$ is not distributive over ; this can be seen intuitively by considering the meanings of the connectives, or formally by using the cut elimination theorem.

MALL is not the full propositional linear logic. Linear logic (**LL**) has, in addition, two modalities ! and ? which allow contraction and weakening to be used in certain situations. The negation operator is extended to **LL** by defining $(!A)^\perp$ to be $?(A^\perp)$. The four rules of inference for ! are:

$$!\textit{:weakening}_0\ \frac{\Gamma \longrightarrow \Delta}{!A, \Gamma \longrightarrow \Delta} \qquad !\textit{:weakening}_1\ \frac{A, \Gamma \longrightarrow \Delta}{!A, \Gamma \longrightarrow \Delta}$$

$$!\textit{:contraction}\ \frac{!A, !A, \Gamma \longrightarrow \Delta}{!A, \Gamma \longrightarrow \Delta} \qquad \frac{!\Gamma \longrightarrow ?\Delta, A}{!\Gamma \longrightarrow ?\Delta, !A}$$

where in the last inference, $!\Gamma$ (respectively, $?\Delta$) represents a cedent containing only formulas beginning with the ! symbol (respectively, the ? symbol).

The dual rules for ? are obtained from these using the negation operation. One should intuitively think of a resource for $!A$ as consisting of zero or more resources for A. The nature of full linear logic with the modalities is quite different from that of either **MALL** or propositional classical logic; in particular, Lincoln et al. [1992] show that **LL** is undecidable.

The above development of the intuitive meanings of the connectives in linear logic has not been as rigorous as one would desire; in particular, it would be nice have a completeness theorem for linear logic based on these intuitive meanings. This has been achieved for some fragments of linear logic by Blass [1992] and Abramsky and Jagadeesan [1994], but has not yet been attained for the full propositional linear logic. In addition to the references above, more information on linear logic may be found in Troelstra [1992], although his notation is different from the standard notation we have used.

Acknowledgements. We are grateful to S. Cook, E. Gertz, C. Gutiérrez, A. Jonasson, C. Lautemann, A. Maciel, R. Parikh C. Pollett, R. Stärk, J. Torán and S. Wainer for reading and suggesting corrections to preliminary versions of this chapter. Preparation of this article was partially supported by NSF grant DMS-9503247 and by cooperative research grant INT-9600919/ME-103 of the NSF and the Czech Republic Ministry of Education.

References

S. Abramsky and R. Jagadeesan
[1994] Games and full completeness for multiplicative linear logic, *Journal of Symbolic Logic*, 59, pp. 543–574.

J. Barwise
[1975] *Admissable Sets and Structures: An Approach to Definability Theory*, Springer-Verlag, Berlin.
[1977] *Handbook of Mathematical Logic*, North-Holland, Amsterdam.

G. Bellin
[1990] *Mechanizing Proof Theory: Resource-Aware Logics and Proof-Transformations to Extract Explicit Information*, PhD thesis, Stanford University.

E. W. Beth
[1953] On Padoa's method in the theory of definition, *Indagationes Mathematicae*, 15, pp. 330–339.
[1956] Semantic entailment and formal derivability, *Indagationes Mathematicae*, 19, pp. 357–388.

A. Blass
[1992] A game semantics for linear logic, *Annals of Pure and Applied Logic*, 56, pp. 183–220.

S. R. Buss
[1986] *Bounded Arithmetic*, Bibliopolis, Napoli. Revision of 1985 Princeton University Ph.D. thesis.

C.-L. Chang
[1970] The unit proof and the input proof in theorem proving, *J. Assoc. Comput. Mach.*, 17, pp. 698–707. Reprinted in: Siekmann and Wrightson [1983,vol 2].

C.-L. Chang and R. C.-T. Lee
[1973] *Symbolic Logic and Mechanical Theorem Proving*, Academic Press, New York.

W. F. Clocksin and C. S. Mellish

[1981] *Programming in Prolog*, North-Holland, Amsterdam, 4th ed.

W. Craig

[1957a] Linear reasoning. A new form of the Herbrand-Gentzen theorem, *Journal of Symbolic Logic*, 22, pp. 250–268.

[1957b] Three uses of the Herbrand-Gentzen theorem in relating model theory and proof theory, *Journal of Symbolic Logic*, 22, pp. 269–285.

M. Davis and H. Putnam

[1960] A computing procedure for quantification theory, *J. Assoc. Comput. Mach.*, 7, pp. 201–215. Reprinted in: Siekmann and Wrightson [1983,vol 1].

P. C. Eklof

[1977] Ultraproducts for algebraists, in: Barwise [1977], pp. 105–137.

S. Feferman

[1968] Lectures on proof theory, in: *Lectures on proof theory, Proceedings of the Summer School in Logic, Leeds, 1967*, M. H. Löb, ed., Lecture Notes in Mathematics #70, Springer-Verlag, Berlin, pp. 1–107.

G. Frege

[1879] *Begriffsschrift, eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Halle. English translation: in van Heijenoort [1967], pp. 1-82.

G. Gentzen

[1935] Untersuchungen über das logische Schliessen, *Mathematische Zeitschrift*, 39, pp. 176–210, 405–431. English translation in: Gentzen [1969], pp. 68-131.

[1969] *Collected Papers of Gerhard Gentzen*, North-Holland, Amsterdam. Edited by M. E. Szabo.

J.-Y. Girard

[1987a] Linear logic, *Theoretical Computer Science*, 50, pp. 1–102.

[1987b] *Proof Theory and Logical Complexity*, vol. I, Bibliopolis, Napoli.

[1989] *Proofs and Types*, Cambridge tracts in theoretical computer science #7, Cambridge University Press. Translation and appendices by P. Taylor and Y. Lafont.

K. Gödel

[1930] Die Vollständigkeit der Axiome des logischen Funktionenkalküls, *Monatshefte für Mathematik und Physik*, 37, pp. 349–360.

R. Harrop

[1960] Concerning formulas of the types $A \to B \vee C$, $A \to (Ex)B(x)$ in intuitionistic formal systems, *Journal of Symbolic Logic*, 25, pp. 27–32.

J. van Heijenoort

[1967] *From Frege to Gödel: A sourcebook in mathematical logic, 1879-1931*, Harvard University Press.

L. Henkin

[1949] The completeness of the first-order functional calculus, *Journal of Symbolic Logic*, 14, pp. 159–166.

L. Henschen and L. Wos

[1974] Unit refutations and Horn sets, *J. Assoc. Comput. Mach.*, 21, pp. 590–605.

J. Herbrand

[1930] *Recherches sur la théorie de la démonstration*, PhD thesis, University of Paris. English translation in Herbrand [1971] and translation of chapter 5 in van Heijenoort [1967], pp. 525-581.

[1971] *Logical Writings*, D. Reidel, Dordrecht, Holland. ed. by W. Goldfarb.

D. Hilbert and W. Ackermann
[1928] *Grundzüge der theoretischen Logik*, Springer-Verlag, Berlin.

D. Hilbert and P. Bernays
[1934-39] *Grundlagen der Mathematik, I & II*, Springer, Berlin.

J. Hintikka
[1955] Form and content in quantification theory, two papers on symbolic logic, *Acta Philosophica Fennica*, 8, pp. 7–55.

W. A. Howard
[1980] The formulas-as-types notion of construction, in: *To H. B. Curry: Essays in Combinatory Logic, Lambda Calculus and Formalism*, J. P. Seldin and J. R. Hindley, eds., Academic Press, New York, pp. 479–491.

S. Kanger
[1957] *Provability in Logic*, Almqvist & Wiksell, Stockholm.

S. C. Kleene
[1952] *Introduction to Metamathematics*, Wolters-Noordhoff, Groningen and North-Holland, Amsterdam.

R. Kowalski
[1979] *Logic for Problem Solving*, North-Holland, Amsterdam.

J. Krajíček, P. Pudlák, and G. Takeuti
[1991] Bounded arithmetic and the polynomial hierarchy, *Annals of Pure and Applied Logic*, 52, pp. 143–153.

G. Kreisel
[1951] On the interpretation of non-finitist proofs–part I, *Journal of Symbolic Logic*, 16, pp. 241–267.
[1952] On the interpretation of non-finitist proofs, part II. interpretation of number theory, applications, *Journal of Symbolic Logic*, 17, pp. 43–58.

P. Lincoln, J. C. Mitchell, A. Scedrov, and N. Shankar
[1992] Decision problems for linear logic, *Annals of Pure and Applied Logic*, 56, pp. 239–311.

E. G. K. Lopez-Escobar
[1965] An interpolation theorem for denumerably long formulas, *Fundamenta Mathematicae*, 57, pp. 253–272.

D. W. Loveland
[1970] A linear format for resolution, in: *Symp. on Automatic Demonstration*, Lecture Notes in Mathematics #125, Springer-Verlag, Berlin, pp. 147–162.
[1978] *Automated Theorem Proving: A Logical Basis*, North-Holland, Amsterdam.

D. Luckham
[1970] Refinement theorems in resolution theory, in: *Symp. on Automatic Demonstration*, Lecture Notes in Mathematics #125, Springer-Verlag, Berlin, pp. 163–190.

R. C. Lyndon
[1959] An interpolation theorem in the predicate calculus, *Pacific Journal of Mathematics*, 9, pp. 129–142.

E. Mendelson
[1987] *Introduction to Mathematical Logic*, Wadsworth & Brooks/Cole, Monterey.

J. R. Munkres
[1975] *Topology: A First Course*, Prentice-Hall, Englewood Cliffs, New Jersey.

M. S. Paterson and M. N. Wegman

[1978] Linear unification, *J. Comput. System Sci.*, 16, pp. 158–167.

G. Peano

[1889] *Arithmetices Principia, noca methodo exposito*, Turin. English translation in: van Heijenoort [1967], pp. 83-97.

D. Prawitz

[1965] *Natural Deduction: A Proof-Theoretical Study*, Almqvist & Wiksell, Stockholm.

A. Robinson

[1956] A result on consistency and it application to the theory of definability, *Indagationes Mathematicae*, 18, pp. 47–58.

G. Robinson and L. Wos

[1969] Paramodulation and theorem-proving in first-order theories with equality, in: *Machine Intelligence 4*, pp. 135–150.

J. A. Robinson

[1965a] Automatic deduction with hyper-resolution, *International Journal of Computer Mathematics*, 1, pp. 227–234. Reprinted in: Siekmann and Wrightson [1983,vol 1].

[1965b] A machine-oriented logic based on the resolution principle, *J. Assoc. Comput. Mach.*, 12, pp. 23–41. Reprinted in: Siekmann and Wrightson [1983,vol 1].

K. Schütte

[1965] Ein System des verknüpfenden Schliessens, *Archiv für Mathematische Logik und Grundlagenforschung*, 2, pp. 55–67.

J. Siekmann and G. Wrightson

[1983] *Automation of Reasoning*, vol. 1&2, Springer-Verlag, Berlin.

J. R. Slagle

[1967] Automatic theorem proving with renamable and semantic resolution, *J. Assoc. Comput. Mach.*, 14, pp. 687–697. Reprinted in: Siekmann and Wrightson [1983,vol 1].

W. W. Tait

[1968] Normal derivability in classical logic, in: *The Syntax and Semantics of Infinitary Languages, Lecture Notes in Mathematics #72*, J. Barwise, ed., Springer-Verlag, Berlin, pp. 204–236.

G. Takeuti

[1987] *Proof Theory*, North-Holland, Amsterdam, 2nd ed.

A. S. Troelstra

[1992] *Lectures on Linear Logic*, Center for the Study of Logic and Information, Stanford.

A. S. Troelstra and D. van Dalen

[1988] *Constructivism in Mathematics: An Introduction*, vol. I&II, North-Holland, Amsterdam.

G. S. Tsejtin

[1968] On the complexity of derivation in propositional logic, *Studies in Constructive Mathematics and Mathematical Logic*, 2, pp. 115–125. Reprinted in: Siekmann and Wrightson [1983,vol 2].

A. N. Whitehead and B. Russell

[1910] *Principia Mathematica*, vol. 1, Cambridge University Press.

L. Wos, R. Overbeek, E. Lusk, and J. Boyle

[1992] *Automated Reasoning: Introduction and Applications*, McGraw-Hill, New York, 2nd ed.

L. Wos and G. Robinson
[1973] Maximal models and refutation completeness: Semidecision procedures in automatic theorem proving, in: *Word Problems: Decision Problems and the Burnside Problem in Group Theory*, W. W. Boone, F. B. Cannonito, and R. C. Lyndon, eds., North-Holland, Amsterdam, pp. 609–639.

L. Wos, G. Robinson, and D. F. Carson
[1965] Efficiency and completeness of the set of support stategy in theorem proving, *J. Assoc. Comput. Mach.*, 12, pp. 201–215. Reprinted in: Siekmann and Wrightson [1983,vol 1].