LOGICAL FOUNDATIONS OF PROOF COMPLEXITY

STEPHEN COOK AND PHUONG NGUYEN

September 3, 2008

Preface

"Proof Complexity" as used here has two related aspects: (i) the complexity of proofs of propositional formulas, and (ii) the study of weak (i.e. "bounded") theories of arithmetic. Aspect (i) goes back at least to Tseitin [83], who proved an exponential lower bound on the lengths of proofs in the weak system known as regular resolution. Later Cook and Reckhow [34] introduced a general definition of propositional proof system and related it to main-stream complexity theory by pointing out that such a system exists in which all tautologies have polynomial length proofs iff the two complexity classes **NP** and *co*-**NP** coincide.

Aspect (ii) goes back to Parikh [67] who introduced the theory known as $\mathbf{I}\Delta_0$, which is Peano Arithmetic with induction restricted to bounded formulas. Paris and Wilkie advanced the study of $\mathbf{I}\Delta_0$ and extensions in a series of papers (including [68, 69]) which relate them to complexity theory. Buss's seminal book [12] introduced the much-studied interleaved hierarchies $\mathbf{S}^i_2$ and $\mathbf{T}^i_2$ of theories related to the complexity classes Σ^p_i making up the polynomial hierarchy. Clote and Takeuti [27] and others introduced a host of theories related to other complexity classes.

The notion of propositional translation, which relates aspects (i) and (ii), goes back to [28], which introduced the equational theory **PV** for polynomial time functions and showed how theorems of **PV** can be translated into families of tautologies which have polynomial length proofs in the extended Frege proof system. Later (and independently) Paris and Wilkie [68] gave an elegant translation of bounded theorems in the relativized theory $\mathbf{I}\Delta_0(R)$ to polynomial length families of proofs in the weak propositional system bounded-depth Frege. Krajíček and Pudlák [56] introduced a hierarchy of proof systems $\langle \mathbf{G}_i \rangle$ for the quantified propositional calculus and showed how bounded theorems in Buss's theory $\mathbf{T}^i_2$ translate into polynomial length proofs in $\mathbf{G}_i$.

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

Our treatment is unusual in that after Chapters 2 and 3 (which present Gentzen's sequent calculus **LK** and the bounded theory $\mathbf{I}\Delta_0$) we present our theories using the two-sorted vocabulary of Zambella [85]: one sort for natural numbers and the other for binary strings (i.e. finite sets of natural numbers). Our point of view is that the objects of interest are the binary strings: they are the natural inputs to the computing devices (Turing machines and Boolean circuits) studied by complexity theorists. The numbers are there as auxiliary variables, for example to index the bits in the strings and measure their length. One reason for using this vocabulary is that the weakest complexity classes (such as $\mathbf{AC}^0$) that we study do not contain integer multiplication as a function, and since standard theories of arithmetic include multiplication as a primitive function, it is awkward to turn them into theories for these weak classes. In fact our theories are simpler than many of the usual single-sorted theories in bounded arithmetic, because there is only one primitive function $|X|$ (the length of X) for strings X, while the axioms for the number sort are just those for $\mathbf{I\Delta}_0$.

Another advantage of using the two-sorted systems is that our propositional translations are especially simple: they are based on the Paris-Wilkie method [68]. The propositional atoms in the translation of a bounded formula $\varphi(X)$ with a free string variable X simply represent the bits of X.

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

One purpose of this book is to serve as a basis for a program we call "Bounded Reverse Mathematics". This is inspired by the Friedman/Simpson program Reverse Mathematics [77], where now "Bounded" refers to bounded arithmetic. The goal is to find the weakest theory capable of proving a given theorem. The theorems in question are those of interest in computer science, and in general these can be proved in weak theories. From the complexity theory point of view, the idea is to find the smallest complexity class such that the theorem can be proved using concepts in that class. This activity not only sheds light on the role of complexity classes in proofs, it can also lead to simplified proofs. A good example is Razborov's [75] greatly simplified proof of

Hastad's Switching Lemma, which grew out of his attempt to formalize the lemma using only polynomial time concepts. His new proof lead to important new results in propositional proof complexity. Throughout the book we give examples of theorems provable in the theories we describe.

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

Two sources have been invaluable for writing this book. The first is Krajíček's monograph [55], which is an essential possession for anyone working in this field. The second source is Buss's chapters [19, 20] in *Handbook of Proof Theory*. Chapter I provides an excellent introduction to the proof theory of **LK**, and Chapter II provides a thorough introduction to the first-order theories of bounded arithmetic.

The authors would like to thank the many students and colleagues who have provided us with feedback on earlier versions of this book.

CONTENTS

Chapter 1

INTRODUCTION

This book studies logical systems which use restricted reasoning based on concepts from computational complexity. One underlying motivation is to determine the complexity of the concepts needed to prove theorems of interest in computer science. The complexity classes of interest lie mainly between the basic class $\mathbf{AC}^0$(whose members are computed by polynomial-size families of bounded-depth circuits), and the polynomial hierarchy $\mathbf{PH}$, and includes the sequence

(1) $$\mathbf{AC}^0 \subset \mathbf{TC}^0 \subseteq \mathbf{NC}^1 \subseteq \mathbf{P} \subseteq \mathbf{PH}$$

where $\mathbf{P}$ is polynomial time. We associate with each of these classes a logical theory and a propositional proof system, where the proof system can be considered a nonuniform version of the universal (or sometimes the bounded) fragment of the theory. The functions definable in the logical theory are those associated with the complexity class, and (in some cases) the lines in a polynomial size proof in the propositional system express concepts in the complexity class. This three-way association for the above classes is depicted as follows:

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

Consider, for example, the class $\mathbf{NC}^1$. The uniform version is **ALogTime**, the class of problems solvable by an alternating Turing machine in time $O(\log n)$. The definable functions in the associated theory $\mathbf{VNC}^1$ are the $\mathbf{NC}^1$ functions, i.e., those functions whose bit graphs are $\mathbf{NC}^1$ relations. A problem in nonuniform $\mathbf{NC}^1$ is defined by a polynomial-size family of log-depth Boolean circuits, or equivalently a polynomial-size family of propositional formulas. The corresponding propositional proof systems are called **Frege** systems, and are described in standard logic textbooks: a **Frege** proof of a tautology A consists of a sequence of propositional formulas ending in A, where each formula is either an axiom or follows from earlier formulas by a rule of inference. Universal theorems of $\mathbf{VNC}^1$ translate into polynomial-size families of **Frege** proofs. Finally, $\mathbf{VNC}^1$ proves the soundness of **Frege** systems, but not of any more powerful propositional proof system.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Most of the theories presented in this book, including those in (2), have the same "second-order" underlying language $\mathcal{L}_A^2$, introduced by

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

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

Chapter 2 provides a sufficient background in first-order logic for the rest of the book, including Gentzen's proof system **LK**. An unusual feature is our treatment of anchored (or "free-cut-free") **LK**-proofs. The completeness of these restricted systems is proved directly by a simple term-model construction as opposed to the usual syntactic cut-elimination method. The second form of the Herbrand Theorem proved here has many applications in later chapters for witnessing theorems.

Chapter 3 presents the necessary background on Peano Arithmetic and its subsystems, including the bounded theory $\mathbf{I}\Delta_0$. The functions definable in $\mathbf{I}\Delta_0$ are precisely those in the complexity class known as **LTH** (the Linear Time Hierarchy). An important theorem needed for this result is that the predicate $y = 2^x$ is definable in the language of arithmetic using a bounded formula (Subsection 3C.3). The universal theory $\overline{\mathbf{I}\Delta_0}$ has function symbols for each function in the Linear Time Hierarchy, and forms a conservative extension of $\mathbf{I}\Delta_0$. This theory serves as a prototype for universal theories defined in later chapters for other complexity classes.

Chapter 4 introduces the syntax and intended semantics for the two-sorted theories, which will be used throughout the remaining chapters. Here Σ_0^B is defined to be the class of formulas with no string quantifiers, and with all number quantifiers bounded. The Σ_1^B-formulas begin with zero or more bounded existential string quantifiers followed by a Σ_0^B-formula, and more generally Σ_i^B-formulas begin with at most i alternating blocks of bounded string quantifiers $\exists\forall\exists\ldots$. Representation theorems are proved which state that formulas in the syntactic class Σ_0^B

represent precisely the (two-sorted) $\mathbf{AC}^0$ relations, and for $i \geq 1$, formulas in Σ_i^B represent the relations in the i-th level of the polynomial hierarchy.

Chapter 5 introduces the hierarchy of two-sorted theories $\mathbf{V}^0 \subset \mathbf{V}^1 \subseteq \mathbf{V}^2 \subseteq \dots$. For $\imath \geq 1$, $\mathbf{V}^i$ is the two-sorted version of Buss's single-sorted theory $\mathbf{S}_2^i$, which is associated with the ith level of the polynomial hierarchy. In this chapter we concentrate on $\mathbf{V}^0$, which is associated with the complexity class $\mathbf{AC}^0$. All two-sorted theories considered in later chapters are extensions of $\mathbf{V}^0$. A Buss-style witnessing theorem is proved for $\mathbf{V}^0$, showing that the existential string quantifiers in a Σ_1^B-theorem of $\mathbf{V}^0$ can be witnessed by $\mathbf{AC}^0$-functions. Since Σ_1^B-formulas have all string quantifiers in front, the proof is easier than for the usual Buss-style witnessing theorems. (The same applies to the witnessing theorems proved in later chapters.) The final section proves that $\mathbf{V}^0$ is finitely axiomatizable.

Chapter 6 concentrates on the theory $\mathbf{V}^1$, which is associated with the complexity class $\mathbf{P}$. All (and only) polynomial time functions are Σ_1^B-definable in $\mathbf{V}^1$. The positive direction is shown in two ways: by analyzing Turing machine computations and by using Cobham's characterization of these functions. The witnessing theorem for $\mathbf{V}^1$ is shown using (two-sorted versions of) the anchored proofs described in Chapter 2, and implies that only polynomial time functions are Σ_1^B-definable in $\mathbf{V}^1$.

Chapter 7 gives a general definition of propositional proof system. The goal is to associate a proof system with each theory so that each Σ_0^B-theorem of the theory translates into a polynomial size family of proofs in the proof system. Further the theory should prove the soundness of the proof system. In this chapter, translations are defined from $\mathbf{V}^0$ to bounded-depth $\mathbf{PK}$-proofs (i.e. bounded-depth Frege proofs), and also from $\mathbf{V}^1$ to extended Frege proofs. Systems $\mathbf{G}_i$ and $\mathbf{G}_i^\star$ for the quantified propositional calculus are defined, and for $i \geq 1$ we show how to translate bounded theorems of $\mathbf{V}^i$ to polynomial size families of proofs in the system $\mathbf{G}_i^\star$. The two-sorted treatment makes these translations simple and natural.

Chapter 8 begins by introducing other two-sorted theories associated with polynomial time. The finitely axiomatized theory $\mathbf{VP}$ and its universal conservative extension $\mathbf{VPV}$ both appear to be weaker than $\mathbf{V}^1$, although they have the same Σ_1^B theorems as $\mathbf{V}^1$. $\mathbf{VP} = \mathbf{TV}^0$ is the base of the hierarchy of theories $\mathbf{TV}^0 \subseteq \mathbf{TV}^1 \subseteq \dots$, where for $i \geq 1$, $\mathbf{TV}^i$ is isomorphic to Buss's single-sorted theory $\mathbf{T}_2^i$. The definable problems in $\mathbf{TV}^1$ have the complexity of Polynomial Local Search. A form of the Herbrand Theorem known as KPT Witnessing is proved and applied to show independence of the Replacement axiom scheme from some theories, and to relating the collapse of the $\mathbf{V}^\infty$ hierarchy with the provable collapse of the polynomial hierarchy. The

Σ_j^B-definable search problems in $\mathbf{V}^i$ and $\mathbf{TV}^i$ are characterized for many i and j. The RSUV isomorphism theorem between $\mathbf{S}_2^i$ and $\mathbf{V}^i$ is proved.

See Table 1 page 239 for a summary of which search problems are definable in $\mathbf{V}^i$ and $\mathbf{TV}^i$.

Chapter 9 gives a uniform way of introducing minimal canonical theories for many complexity classes between $\mathbf{AC}^0$ and $\mathbf{P}$, including those mentioned earlier in (1). Each finitely axiomatized theory is defined as an extension of $\mathbf{V}^0$ obtained by adding a single axiom stating the existence of a computation solving a complete problem for the associated complexity class. The "minimality" of each theory is established by defining a universal theory whose axioms are simply the defining axioms for all the functions in the associated complexity class. These functions are defined as the function $\mathbf{AC}^0$-closure of the complexity class, or (as is the case for $\mathbf{P}$) using a recursion-theoretic characterization of the function class. The main theorem in each case is that the universal theory is a conservative extension of the finitely axiomatized theory.

Chapter 10 extends Chapter 7 by presenting quantified propositional proof systems associated with various complexity classes, and defining translations from the bounded theorems of the theories introduced in Chapter 9 to the appropriate proof system. The theories prove the (restricted) soundness of the associated proof system (this is the *reflection principle*).

Chapter 2

THE PREDICATE CALCULUS AND THE SYSTEM LK

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

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

The first section treats the simple case of propositional calculus.

2A. Propositional calculus

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

Formulas are built according to the following rules:

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

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

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

A *truth assignment* is an assignment of truth values F, T to atoms. Given a truth assignment τ, the truth value A^τ of a formula A is defined inductively as follows: $\bot^\tau = F$, $\top^\tau = T$, $P^\tau = \tau(P)$ for atom P, $(A \wedge B)^\tau = T$ iff both $A^\tau = T$ and $B^\tau = T$, $(A \vee B)^\tau = T$ iff either $A^\tau = T$ or $B^\tau = T$, $(\neg A)^\tau = T$ iff $A^\tau = F$.

DEFINITION 2.1. *A truth assignment τ* satisfies *A iff $A^\tau = T$; τ* satisfies *a set Φ of formulas iff τ satisfies A for all $A \in \Phi$. Φ is* satisfiable *iff some τ satisfies Φ; otherwise Φ is* unsatisfiable. *Similarly for A. $\Phi \models A$ (i.e., A is a* logical consequence *of Φ) iff τ satisfies A for every τ such that τ satisfies Φ. A formula A is* valid *iff $\models A$ (i.e., $A^\tau = T$ for all τ). A valid propositional formula is called a* tautology. *We say that A and B are* equivalent *(written $A \Longleftrightarrow B$) iff $A \models B$ and $B \models A$.*

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

2A.1. Gentzen's propositional proof system PK. We present the propositional part **PK** of Gentzen's sequent-based proof system **LK**. Each line in a proof in the system **PK** is a *sequent* of the form

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

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

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

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

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

DEFINITION 2.2. *A* **PK** proof *of a sequent S is a finite tree whose nodes are (labeled with) sequents, whose root (called the* endsequent*) is S and is written at the bottom, whose leaves (or* initial sequents*) are logical axioms (see below), such that each non-leaf sequent follows from the sequent(s) immediately above by one of the rules of inference given below.*

The *logical axioms* are of the form

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

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

weakening rules

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

exchange rules

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

contraction rules

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

$\neg$ introduction rules

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

$\wedge$ introduction rules

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

$\vee$ introduction rules

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

cut rule

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

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

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

Note that repeated use of the exchange rules allows us to execute an arbitrary reordering of the formulas in the antecedent or succedent of a sequent. In presenting a proof in the system **PK**, we will usually omit mention of the steps requiring the exchange rules, but of course they are there implicitly.

Definition 2.3. *A* **PK** *proof of a formula A is a* **PK** *proof of the sequent $\longrightarrow A$.*

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

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

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

$$\begin{prooftree}
\AxiomC{$P \longrightarrow P$}
\RightLabel{(\textit{weakening})}
\UnaryInfC{$P \longrightarrow P, \neg Q$}
\RightLabel{($\neg$ \textit{right})}
\UnaryInfC{$\longrightarrow P, \neg P, \neg Q$}
\AxiomC{$Q \longrightarrow Q$}
\RightLabel{(\textit{weakening})}
\UnaryInfC{$Q \longrightarrow Q, \neg P$}
\RightLabel{($\neg$ \textit{right})}
\UnaryInfC{$\longrightarrow Q, \neg P, \neg Q$}
\RightLabel{($\wedge$ \textit{right})}
\BinaryInfC{$\longrightarrow P \wedge Q, \neg P, \neg Q$}
\RightLabel{($\vee$ \textit{right})}
\UnaryInfC{$\longrightarrow P \wedge Q, \neg P \vee \neg Q$}
\RightLabel{($\neg$ \textit{left})}
\UnaryInfC{$\neg(P \wedge Q) \longrightarrow \neg P \vee \neg Q$}
\end{prooftree}$$

Exercise 2.4. *Give* **PK** *proofs for each of the following valid sequents:*

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

Exercise 2.5. *Show that the contraction rules can be derived from the cut rule (with weakenings and exchanges).*

Exercise 2.6. *Suppose that we allowed $\supset$ as a primitive connective, rather than one introduced by definition. Give the appropriate left and right introduction rules for $\supset$.*

2A.2. Soundness and completeness of PK. Now we prove that **PK** is both sound and complete. That is, a propositional sequent is provable in **PK** iff it is valid.

Theorem 2.7 (Soundness Theorem). *Every sequent provable in* **PK** *is valid.*

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

THEOREM 2.8 (Completeness Theorem). *Every valid propositional sequent is provable in* **PK** *without using cut or contraction.*

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

LEMMA 2.9 (Inversion Principle). *For each* **PK** *rule except for weakenings, if the bottom sequent is valid, then all top sequents are valid.*

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

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

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

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

2A.3. PK proofs from assumptions. We generalize the (semantic) definition of logical consequence from formulas to sequents in the obvious way: A sequent S is a *logical consequence* of a set Φ of sequents iff every truth assignment τ that satisfies Φ also satisfies S. We generalize the (syntactic) definition of a **PK** proof of a sequent S to a **PK** proof of S *from a set* Φ *of sequents* (also called a **PK**-Φ proof) by allowing sequents in Φ to be leaves (called *nonlogical axioms*) in the proof tree, in addition to the logical axioms. It turns out that soundness and completeness generalize to this setting.

THEOREM 2.10 (Derivational Soundness and Completeness Theorem). *A propositional sequent S is a logical consequence of a set Φ of sequents iff S has a* **PK**-Φ *proof.*

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

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

In general, to prove S from Φ the cut rule is required. For example, there is no **PK** proof of $\longrightarrow P$ from $\longrightarrow P \wedge Q$ without using the cut rule. This follows from the *subformula property*, which states that in a cut-free proof π of a sequent S, every formula in every sequent of π is a subformula of some formula in S. This is stated more generally in the Proposition 2.15.

EXERCISE 2.11. *Let A_S be the formula giving the meaning of a sequent S, as in (8). Show that there is a cut-free* **PK** *derivation of $\longrightarrow A_S$ from S.*

From the above easy exercise and from the earlier Completeness Theorem and from Theorem 2.16, Form 2 (compactness), we obtain an easy proof of derivational completeness. Suppose that the sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of sequents $S_1, ..., S_k$. Then by the above exercise we can derive each of the sequents $\longrightarrow A_{S_1}$,...,$\longrightarrow A_{S_k}$ from the sequents $S_1, ..., S_k$. Also the sequent

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

is valid, and hence has a **PK** proof by Theorem 2.8. Finally from (9) using successive cuts with cut formulas $A_{S_1}, ..., A_{S_k}$ we obtain the desired **PK** derivation of $\Gamma \longrightarrow \Delta$ from the the sequents $S_1, ..., S_k$. □

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

DEFINITION 2.12 (Anchored Proof). *An instance of the cut rule in a* **PK**-Φ *proof π is* anchored *if the cut formula A (also) occurs as a formula (rather than a subformula) in some nonlogical axiom of π. A* **PK**-Φ *proof π is* anchored *if every instance of cut in π is anchored.*

Our *anchored* proofs are similar to *free-cut-free* proofs in [55] and elsewhere. Our use of the term *anchored* is inspired by [19].

The derivational completeness theorem can be strengthened as follows.

THEOREM 2.13 (Anchored Completeness Theorem). *If a propositional sequent S is a logical consequence of a set Φ of sequents, then there is an anchored* **PK**-Φ *proof of S.*

We illustrate the proof of the anchored completeness theorem by proving the special case in which Φ consists of the single sequent $A \longrightarrow B$. Assume that the sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of $A \longrightarrow B$. Then both of the sequents $\Gamma \longrightarrow \Delta, A$ and $B, A, \Gamma \longrightarrow \Delta$ are valid (why?). Hence by Theorem 2.8 they have **PK** proofs π_1 and π_2, respectively. We can use these proofs to get a proof of $\Gamma \longrightarrow \Delta$ from $A \longrightarrow B$ as shown below, where the double line indicates the rules weakening and exchange have been applied.

$$\dfrac{\begin{array}{c}\vdots\,\pi_1\\ \Gamma \longrightarrow \Delta, A\end{array} \qquad \dfrac{\dfrac{A \longrightarrow B}{A, \Gamma \longrightarrow \Delta, B} \qquad \begin{array}{c}\vdots\,\pi_2\\ B, A, \Gamma \longrightarrow \Delta\end{array}}{A, \Gamma \longrightarrow \Delta}\,(cut)}{\Gamma \longrightarrow \Delta}\,(cut)$$

Next consider the case in which Φ has the form

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

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

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

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

$$\dfrac{\dfrac{\longrightarrow A_1}{A_2, ..., A_k, \Gamma \longrightarrow \Delta, A_1} \qquad \begin{array}{c}\vdots\,\pi\\ A_1, A_2, ..., A_k, \Gamma \longrightarrow \Delta\end{array}}{A_2, ..., A_k, \Gamma \longrightarrow \Delta}\,(cut)$$

Exercise 2.14. *Prove the anchored completeness theorem for the more general case in which Φ is any finite set of sequents. Use induction on the number of sequents in Φ.*

A nice property of anchored proofs is the following.

Proposition 2.15 (Subformula Property). *If π is an anchored* **PK**-Φ *proof of S, then every formula in every sequent of π is a subformula of a formula either in S or in some nonlogical axiom of π.*

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

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

2A.4. Propositional compactness. We conclude our treatment of the propositional calculus with a fundamental result which also plays an important role in the predicate calculus.

THEOREM 2.16 (Propositional Compactness Theorem). *We state three different forms of this result. All three are equivalent.*

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

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

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

EXERCISE 2.17. *Prove the equivalence of the three forms. (Note that Form 3 is the contrapositive of Form 1.)*

PROOF OF FORM 1. Let Φ be an unsatisfiable set of formulas. By our definition of propositional formula, all propositional variables in Φ come from a countable list $P_1, P_2, \ldots$. (See Exercise 2.19 for the uncountable case.) Organize the set of truth assignments into an infinite rooted binary tree B. Each node except the root is labeled with a literal P_i or $\neg P_i$. The two children of the root are labeled P_1 and $\neg P_1$, indicating that P_1 is assigned T or F, respectively. The two children of each of these nodes are labeled P_2 and $\neg P_2$, respectively, indicating the truth value of P_2. Thus each infinite branch in the tree represents a complete truth assignment, and each path from the root to a node represents a truth assignment to the atoms $P_1, \ldots, P_i$, for some i.

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

LEMMA 2.18 (König's Lemma). *Suppose T is a rooted tree in which every node has only finitely many children. If every branch in T is finite, then T is finite.*

PROOF. We prove the contrapositive: If T is infinite (but every node has only finitely many children) then T has an infinite branch. We can define an infinite path in T as follows: Start at the root. Since T is infinite but the root has only finitely many children, the subtree rooted at one of these children must be infinite. Choose such a child as the second node in the branch, and continue. $\dashv$

Exercise 2.19. (For those with some knowledge of set theory or point set topology) *The above proof of the propositional compactness theorem only works when the set of atoms is countable, but the result still holds even when Φ is an uncountable set with an uncountable set $\mathcal{A}$ of atoms. Complete each of the two proof outlines below.*

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

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

2B. Predicate calculus

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

2B.1. Syntax of the predicate calculus. A *first-order language* (or just *language*, or *vocabulary*) $\mathcal{L}$ is specified by the following:

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

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

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

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

DEFINITION 2.20 ($\mathcal{L}$-Terms). *Let $\mathcal{L}$ be a first-order vocabulary.*

1) *Every variable is an $\mathcal{L}$-term.*
2) *If f is an n-ary function symbol of $\mathcal{L}$ and $t_1, \ldots, t_n$ are $\mathcal{L}$-terms, then $ft_1 \ldots t_n$ is an $\mathcal{L}$-term.*

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

DEFINITION 2.21 ($\mathcal{L}$-Formulas). *Let $\mathcal{L}$ be a first-order language. First-order formulas in $\mathcal{L}$ (or $\mathcal{L}$-*formulas*, or just* formulas*) are defined inductively as follows:*

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

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

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

It can be shown that no proper initial segment of a term is a term, and hence every term can be parsed uniquely according to Definition 2.20. A similar remark applies to formulas, and Definition 2.21.

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

DEFINITION 2.22 (The Language of Arithmetic).

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

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

DEFINITION 2.23 (Free and Bound Variables). *An occurrence of x in A is* bound *iff it is in a subformula of A of the form $\forall x B$ or $\exists x B$. Otherwise the occurrence is* free.

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

Definition 2.24. *A formula is* closed *if it contains no free occurrence of a variable. A term is* closed *if it contains no variable. A closed formula is called a* sentence*.*

2B.2. Semantics of predicate calculus.

Definition 2.25 ($\mathcal{L}$-Structure). *If $\mathcal{L}$ is a first-order language, then an $\mathcal{L}$-structure $\mathcal{M}$ consists of the following:*

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

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

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

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

Definition 2.26 (Object Assignment). *An* object assignment *σ for a structure $\mathcal{M}$ is a mapping from variables to the universe M.*

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

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

Definition 2.27 (Basic Semantic Definition). *Let $\mathcal{L}$ be a first-order language, let $\mathcal{M}$ be an $\mathcal{L}$-structure, and let σ be an object assignment for $\mathcal{M}$. Each $\mathcal{L}$-term t is assigned an element $t^{\mathcal{M}}[\sigma]$ in M, defined by structural induction on terms t, as follows (refer to the definition of $\mathcal{L}$-term):*

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

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

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

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

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

DEFINITION 2.28 (Standard Model). *The* standard model $\underline{\mathbb{N}}$ *for the language $\mathcal{L}_A$ is a structure with universe $M = \mathbb{N} = \{0, 1, 2, \dots\}$, where $0, 1, +, \cdot, =, \leq$ get their usual meanings on the natural numbers.*

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

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

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

DEFINITION 2.29. (a) $\mathcal{M} \models \Phi[\sigma]$ *iff* $\mathcal{M} \models A[\sigma]$ *for all* $A \in \Phi$.
(b) $\mathcal{M} \models \Phi$ *iff* $\mathcal{M} \models \Phi[\sigma]$ *for all* σ.
(c) $\Phi \models A$ *iff for all $\mathcal{M}$ and all σ, if* $\mathcal{M} \models \Phi[\sigma]$ *then* $\mathcal{M} \models A[\sigma]$.
(d) $\models A$ *(A is* valid*) iff* $\mathcal{M} \models A[\sigma]$ *for all $\mathcal{M}$ and σ.*
(e) $A \iff B$ *(A and B are* logically equivalent*, or just* equivalent*) iff for all $\mathcal{M}$ and all σ,* $\mathcal{M} \models A[\sigma]$ *iff* $\mathcal{M} \models B[\sigma]$.

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

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

Definition 2.30 (Substitution). *Let s, t be terms, and A a formula. Then $t(s/x)$ is the result of replacing all occurrences of x in t by s, and $A(s/x)$ is the result of replacing all* free *occurrences of x in A by s.*

Lemma 2.31. *For each structure $\mathcal{M}$ and each object assignment σ,*

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

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

Proof. Structural induction on s. ⊣

Definition 2.32. *A term t is* freely substitutable for x in A *iff no free occurrence of x in A is in a subformula of A of the form $\forall yB$ or $\exists yB$, where y occurs in t.*

Theorem 2.33 (Substitution Theorem). *If t is freely substitutable for x in A then for all structures $\mathcal{M}$ and all object assignments σ, $\mathcal{M} \models A(t/x)[\sigma]$ iff $\mathcal{M} \models A[\sigma(m/x)]$, where $m = t^{\mathcal{M}}[\sigma]$.*

Proof. Structural induction on A. ⊣

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

Theorem 2.34 (Formula Replacement Theorem). *If B and B' are equivalent and A' results from A by replacing some occurrence of B in A by B', then A and A' are equivalent.*

Proof. Structural induction on A relative to B. ⊣

2B.3. The first-order proof system LK. We now extend the propositional proof system **PK** to the first-order sequent proof system **LK**. For this it is convenient to introduce two kinds of variables: *free variables* denoted by $a, b, c, \ldots$ and *bound variables* denoted by $x, y, z, \ldots$. A first-order sequent has the form $A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell$, where now the A_i's and B_j's are first-order formulas satisfying the restriction that they have no free occurrences of the "bound" variables $x, y, z, \ldots$ and no bound occurrences of the "free" variables $a, b, c, \ldots$.

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

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

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

$\forall$ introduction rules

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

$\exists$ introduction rules

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

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

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

DEFINITION 2.35 (Semantics of first-order sequents). *The semantics of first-order sequents is a natural generalization of the semantics of propositional sequents. Again the sequent $A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell$ has the same meaning as its associated formula*

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

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

THEOREM 2.36 (Soundness Theorem for **LK**). *Every sequent provable in* **LK** *is valid.*

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

EXERCISE 2.37. *Give examples to show that the restriction given on the quantifier rules, that b must not occur in the conclusion in $\forall$-right and $\exists$-left, is necessary to ensure that these rules preserve validity.*

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

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

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

Exercise 2.38. *Give* **LK** *proofs for the following valid sequents:*

(a) $\forall x Px \wedge \forall x Qx \longrightarrow \forall x(Px \wedge Qx)$
(b) $\forall x(Px \wedge Qx) \longrightarrow \forall x Px \wedge \forall x Qx$
(c) $\exists x(Px \vee Qx) \longrightarrow \exists x Px \vee \exists x Qx$
(d) $\exists x Px \vee \exists x Qx \longrightarrow \exists x(Px \vee Qx)$
(e) $\exists x(Px \wedge Qx) \longrightarrow \exists x Px \wedge \exists x Qx$
(f) $\exists y \forall x Pxy \longrightarrow \forall x \exists y Pxy$
(g) $\forall x Px \longrightarrow \exists x Px$

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

(h) $\exists x Px \wedge \exists x Qx \longrightarrow \exists x(Px \wedge Qx)$
(i) $\forall x \exists y Pxy \longrightarrow \exists y \forall x Pxy$

2B.4. Free variable normal form. In future chapters it will be useful to assume that **LK** proofs satisfy certain restrictions on free variables.

Definition 2.39 (Free Variable Normal Form). *Let π be an* **LK** *proof with endsequent S. A free variable in S is called a* parameter variable *of π. We say π is in* free variable normal form *if (1) no free variable is eliminated from any sequent in π by any rule except possibly ∀-right and ∃-left, and in these cases the eigenvariable which is eliminated is not a parameter variable, and (2) every nonparameter free variable appearing in π is used exactly once as an eigenvariable.*

Thus if a proof is in free variable normal form, then any occurrence of a parameter variable persists until the endsequent, and any occurrence of a nonparameter free variable persists until it is eliminated as an eigenvariable in ∀-right or ∃-left.

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

It is important that π have a tree structure in order for the procedure to work.

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

2B.5. Completeness of LK without equality. Notation. Let Φ be a set of formulas. Then $\longrightarrow \Phi$ is the set of all sequents of the form $\longrightarrow A$, where A is in Φ.

Definition 2.40. *Assume that the underlying vocabulary does not contain $=$. If Φ is a set of formulas, then an* **LK**-Φ *proof is an* **LK** *proof in which sequents at the leaves may be either logical axioms or nonlogical axioms of the form $\longrightarrow A$, where A is in Φ.*

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

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

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

Definition 2.41. *Suppose that A is a formula whose free variables comprise the list $a_1, \dots, a_n$. Then the* universal closure *of A, written $\forall A$, is the sentence $\forall x_1 \dots \forall x_n A(x_1/a_1, \dots, x_n/a_n)$, where $x_1, \dots, x_n$ is a list of new (bound) variables. If Φ is a set of formulas, then $\forall\Phi$ is the set of all sentences $\forall A$, for A in Φ.*

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

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

Theorem 2.42 (Derivational Soundness and Completeness of **LK**). *Assume that the underlying language does not contain $=$. Let Φ be a set of formulas and let $\Gamma \longrightarrow \Delta$ be a sequent. Then there is an* **LK**-Φ *proof* *of* $\Gamma \longrightarrow \Delta$ *iff* $\Gamma \longrightarrow \Delta$ *is a logical consequence of* $\forall\Phi$. *The soundness (only if) direction holds also when the underlying language contains* $=$.

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

Thus we define the closure $\forall S$ of a sequent S to be the closure of its associated formula A_S (Definition 2.35). Note that if $S =_{\text{syn}} \Gamma \longrightarrow \Delta$, then $\forall S$ is *not* equivalent to $\forall\Gamma \longrightarrow \forall\Delta$ in general.

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

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

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

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

Lemma 2.43 (Completeness Lemma). *Assume that the underlying language does not contain* $=$. *If* $\Gamma \longrightarrow \Delta$ *is a sequent and* Φ *is a (possibly infinite) set of formulas such that* $\Gamma \longrightarrow \Delta$ *is a logical consequence of* Φ, *then there is a finite subset* $\{C_1, \dots, C_n\}$ *of* Φ *such that the sequent*

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

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

Note that a form of the Compactness Theorem for predicate calculus sentences without equality follows from the above lemma. See Theorem 2.61 for a more general form of compactness.

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

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

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

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

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

PROOF OF THE COMPLETENESS LEMMA. We loosely follow the proof of the Cut-free Completeness Theorem, pp 33-36 of Buss [19]. (Warning: our definition of logical consequence differs from Buss's when the formulas in the hypotheses have free variables.) We will only prove it for the case in which the underlying first-order language $\mathcal{L}$ has a countable set (including the case of a finite set) of function and predicate symbols; i.e., the function symbols form a list $f_1, f_2, \ldots$ and the predicate symbols form a list $P_1, P_2, \ldots$. This may not seem like much of a restriction, but for example in developing the model theory of the real numbers, it is sometimes useful to introduce a distinct constant symbol e_c for every real number c; and there are uncountably many real numbers. The completeness theorem and lemma hold for the uncountable case, but we shall not prove them for this case.

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

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

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

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

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

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

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

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

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

Step 2: If A_i is atomic, do nothing and proceed to the next stage. Otherwise, modify π at the active sequents which contain A_i by applying the appropriate introduction rule in reverse, much as in the proof of propositional completeness (Theorem 2.8). (It suffices to pick any one occurrence of A_i in each active sequent.) For example, if A_i is of the form $B \vee C$, then every active sequent in π of the form $\Gamma', B \vee C, \Gamma'' \longrightarrow \Delta'$ is replaced by the derivation

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

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

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

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

where c is a new free variable, not used in π yet. (Also c may not occur in any formula in Φ, because otherwise at a later stage, Step 1 of the procedure might cause the variable restriction in the $\exists$-left rule to be violated.) In addition, any active sequent of the form $\Gamma' \longrightarrow \Delta', \exists x B(x), \Delta''$ is replaced by the derivation

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

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

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

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

EXERCISE 2.44. *Carry out the case above in which A_i begins with a universal quantifier.*

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

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

If π has only finitely many nodes, then at least one leaf node must be active (and contain only atomic formulas), since otherwise the algorithm would terminate. In this case, let β be a path in π from the root extending up to this active node. If on the other hand π has infinitely many nodes, then by Lemma 2.18 (König), there must be an infinite branch β in π starting at the root and extending up through the tree. Thus in either case, β is a branch in π starting at the root, extending up through the tree, and such that all sequents on β were once active, and hence have no formula occurring on both the left and right, no $\top$ on the right and no $\bot$ on the left.

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

DEFINITION 2.45 (Construction of the "Term Model" $\mathcal{M}$). *The universe M of $\mathcal{M}$ is the set of all $\mathcal{L}$-terms t (which contain only "free" variables $a, b, c, \dots$). The object assignment σ just maps every variable a to itself.*

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

Exercise 2.46. *Prove by structural induction that for every term* t*,* $t^{\mathcal{M}}[\sigma] = t$*.*

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

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

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

For the induction step, there is a different case for each of the ways of constructing a formula from simpler formulas (see Definition 2.21). In general, if A occurs in some sequent in β, then A persists upward in every higher sequent of β until it becomes the active formula ($A =_{\text{syn}} A_i$). Each case is handled by the corresponding introduction rule used in the algorithm. For example, if A is of the form $B \vee C$ and A occurs on the left of a sequent in β, then the rule $\vee$-left is applied in reverse, so that when β is extended upward either it will have some antecedent containing B or one containing C. In the case of B, we know that $\mathcal{M}$ and σ satisfy B by the induction hypothesis, and hence they satisfy $B \vee C$. (Similarly for C.)

Now consider the interesting case in which A is $\exists x B(x)$ and A occurs in some succedent of β. (See Step 2 above to find out what happens when A becomes active in this case.) The path β will hit a succedent with $B(t_j)$ in the succedent, and by the induction hypothesis, $\mathcal{M}$ and σ falsify $B(t_j)$. But this succedent still has a copy of $\exists x B(x)$, and in fact this copy will be in *every* succedent of β above this point. Hence *every* $\mathcal{L}$-term t will eventually be of the form t_j and so the formula $B(t)$ will occur as a succedent on β. (This is why we assumed that every term appears infinitely often in the sequence $t_1, t_2, \dots$.) Therefore $\mathcal{M}$ and σ

falsify $B(t)$ for every term t (i.e., for every element in the universe of $\mathcal{M}$). Therefore they falsify $\exists x B(x)$, as required.

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

We now wish to strengthen the derivational completeness of **LK** and show that cuts can be restricted so that cut formulas are in Φ. The definition of *anchored* **PK** proof (Definition 2.12) can be generalized to *anchored* **LK** proof. We will continue to restrict our attention to the case in which all nonlogical axioms have the simple form $\longrightarrow A$, although an analog of the following theorem does hold for an arbitrary set of nonlogical axioms, provided they are closed under substitution of terms for variables.

THEOREM 2.47 (Anchored **LK** Completeness Theorem). *Assume that the underlying language does not contain =. Suppose that Φ is a set of formulas closed under substitution of terms for variables. (I.e., if $A(b)$ is in Φ, and t is any term not containing "bound" variables $x, y, z, \ldots$, then $A(t)$ is also in Φ.) Suppose that $\Gamma \longrightarrow \Delta$ is a sequent that is a logical consequence of $\forall\Phi$. Then there is an* **LK**-Φ *proof of $\Gamma \longrightarrow \Delta$ in which the cut rule is restricted so that the only cut formulas are formulas in Φ.*

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

EXERCISE 2.48. *Show how to modify the proof of the Completeness Lemma to obtain a proof of the Anchored* **LK** *Completeness Theorem. Explain the following modifications to that proof.*

(a) *The definition of* active sequent *on page 25 must be modified, since now we are allowing nonlogical axioms in π. Give the precise new definition.*
(b) *Step 1 of the procedure on page 25 must be modified, because now we are looking for a derivation of $\Gamma \longrightarrow \Delta$ from nonlogical axioms, rather than a proof of $C_1, \ldots, C_n, \Gamma \longrightarrow \Delta$. Describe the modification. (We still need to bring formulas A_i of Φ somehow into the proof, and your modification will involve adding a short derivation to π.)*
(c) *The restriction given in Step 2 for the case in which $\exists x B(x)$ is in the antecedent, that the variable c must not occur in any formula in Φ, must be dropped. Explain why.*

(d) *Explain why the term model $\mathcal{M}$ and object assignment σ, described on page 26 (Definition 2.45), satisfy $\forall\Phi$. This should follow from the Claim on page 27, and your modification of Step 1, which should ensure that each formula in Φ occurs in the antecedent of some sequent in every branch in π. Conclude that $\Gamma \longrightarrow \Delta$ is not a logical consequence of $\forall\Phi$ (when the procedure does not terminate).*

2C. Equality axioms

Definition 2.49. *A* weak *$\mathcal{L}$-structure $\mathcal{M}$ is an $\mathcal{L}$-structure in which we drop the requirement that $=^{\mathcal{M}}$ is the equality relation (i.e., $=^{\mathcal{M}}$ can be any binary relation on M.)*

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

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

Definition 2.50 (Equality Axioms of $\mathcal{L}$ ($\mathcal{E}_{\mathcal{L}}$)).

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

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

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

EA4. $\forall x_1 \dots \forall x_n \forall y_1 \dots \forall y_n(x_1 = y_1 \wedge \cdots \wedge x_n = y_n) \supset fx_1 \dots x_n = fy_1 \dots y_n$ *for each $n \geq 1$ and each n-ary function symbol f in $\mathcal{L}$.*

EA5. $\forall x_1 \dots \forall x_n \forall y_1 \dots \forall y_n(x_1 = y_1 \wedge \cdots \wedge x_n = y_n) \supset (Px_1 \dots x_n \supset Py_1 \dots y_n)$ *for each $n \geq 1$ and each n-ary predicate symbol P in $\mathcal{L}$ other than $=$.*

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

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

Theorem 2.51 (Equality Theorem). *Let Φ be any set of $\mathcal{L}$-formulas. Then Φ is satisfiable iff $\Phi \cup \mathcal{E}_{\mathcal{L}}$ is satisfied by some weak $\mathcal{L}$-structure.*

Corollary 2.52. *$\Phi \models A$ iff for every weak $\mathcal{L}$-structure $\mathcal{M}$ and every object assignment σ, if $\mathcal{M}$ satisfies $\Phi \cup \mathcal{E}_{\mathcal{L}}$ under σ then $\mathcal{M}$ satisfies A under σ.*

Corollary 2.53. *$\forall\Phi \models A$ iff A has an* **LK**-*Ψ proof, where $\Psi = \Phi \cup \mathcal{E}_{\mathcal{L}}$.*

Corollary 2.52 follows immediately from the Equality Theorem and the fact that $\Phi \models A$ iff $\Phi \cup \{\neg A\}$ is unsatisfiable. Corollary 2.53 follows from Corollary 2.52 and the derivational soundness and completeness of **LK** (page 22), where in applying that theorem we treat $=$ as just another binary relation (so we can assume $\mathcal{L}$ does not have the official equality symbol).

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

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

2C.1. Equality axioms for LK. For the purpose of using an **LK** proof to establish $\Phi \models A$, we can replace the standard equality axioms **EA1**, ..., **EA5** by the following quantifier-free sequent schemes, where we must include an instance of the sequent for all terms t, u, v, t_i, u_i (not involving "bound" variables $x, y, z, \dots$).

DEFINITION 2.54 (Equality Axioms for **LK**).

E1. $\longrightarrow t = t$
E2. $t = u \longrightarrow u = t$
E3. $t = u, u = v \longrightarrow t = v$
E4. $t_1 = u_1, \dots, t_n = u_n \longrightarrow ft_1 \dots t_n = fu_1 \dots u_n$, *for each* f *in* $\mathcal{L}$
E5. $t_1 = u_1, \dots, t_n = u_n, Pt_1 \dots t_n \longrightarrow Pu_1 \dots u_n$, *for each* P *in* $\mathcal{L}$ *(Here* P *is not* $=$*)*

Note that the universal closures of **E**1, ... , **E**5 are semantically equivalent to **EA**1, ... , **EA**5, and in fact using the **LK** rule $\forall$-right repeatedly, $\longrightarrow$ **EA**i is easily derived in **LK** from **E**i (with terms t, u, etc., taken to be distinct variables), $i = 1, \dots, 5$. Thus Corollary 2.53 above still holds when $\Psi = \Phi \cup \{$**E**1, ... , **E**5$\}$.

DEFINITION 2.55 (Revised Definition of **LK** with $=$). *If* Φ *is a set of* $\mathcal{L}$*-formulas, where* $\mathcal{L}$ *includes* $=$*, then by an* **LK**-Φ *proof we now mean an* **LK**-Ψ *proof in the sense of the earlier definition, page 22, where* Ψ *is* Φ *together with all instances of the equality axioms* **E1**, ..., **E5**. *If* Φ *is empty, we simply refer to an* **LK**-*proof (but allow* **E1**, ..., **E5** *as axioms).*

2C.2. Revised soundness and completeness of LK.

Theorem 2.56 (Revised Soundness and Completeness of **LK**). *For any set Φ of formulas and sequent S,*

$$\forall\Phi \models S \text{ iff } S \text{ has an } \mathbf{LK}\text{-}\Phi \text{ proof}$$

Notation. $\Phi \vdash A$ means that there is an **LK**-Φ proof of $\longrightarrow A$.

Recall that if Φ is a set of sentences, then $\forall\Phi$ is the same as Φ. Therefore

$$\Phi \models A \text{ iff } \Phi \vdash A, \quad \text{if } \Phi \text{ is a set of sentences}$$

Restricted use of cut. Note that **E**1, ... , **E**5 have no universal quantifiers, but instead have instances for all terms $t, u, \ldots$. Recall that in an anchored **LK** proof, cuts are restricted so that cut formulas must occur in the nonlogical axioms. In the presence of equality, the nonlogical axioms must include **E**1, ... , **E**5, but the only formulas occurring here are equations of the form $t = u$. Since the Anchored **LK** Completeness Theorem (page 28) still holds when Φ is a set of sequents rather than a set of formulas, and since **E**1, ... , **E**5 are closed under substitution of terms for variables, we can extend this theorem so that it works in the presence of equality.

Definition 2.57 (Anchored **LK** Proof). *An* **LK**-Φ *proof* π *is* anchored[1] *provided every cut formula in* π *is a formula in some nonlogical axiom of* π *(including possibly* **E**1, ... , **E**5*).*

Theorem 2.58 (Anchored **LK** Completeness Theorem with Equality). *Suppose that* Φ *is a set of formulas closed under substitution of terms for variables and that the sequent* S *is a logical consequence of* $\forall\Phi$*. Then there is an anchored* **LK**-Φ *proof of* S*.*

The proof is immediate from the Anchored **LK** Completeness Theorem (page 28) and the above discussion about axioms **E**1, ... , **E**5.

We are interested in anchored proofs because of their subformula property. The following result generalizes Proposition 2.15.

Theorem 2.59 (Subformula Property of Anchored **LK** Proofs). *If* π *is an anchored* **LK**-Φ *proof of a sequent* S*, then every formula in every sequent of* π *is a term substitution instance of a subformula of a formula either in* S *or in a nonlogical axiom of* π *(including* **E**1, ... , **E**5*).*

Proof sketch. The proof is by induction on the number of sequents in π. The induction step is proved by inspecting each **LK** rule. The case of the cut rule uses the fact that every cut formula in an anchored proof is a formula in some nonlogical axiom. The reason that we must consider term substitutions is because of the four quantifier

[1] The definition of *anchored* in [19] is slightly stronger and more complicated

rules. For example, in $\exists$-right, the formula $A(t)$ occurs on top, and this is a substitution instance of a subformula of $\exists x A(x)$, which occurs on the bottom. $\dashv$

2D. Major corollaries of completeness

Theorem 2.60 (Löwenheim/Skolem Theorem). *If a set Φ of formulas from a countable language is satisfiable, then Φ is satisfiable in a countable (possibly finite) universe.*

Proof. Suppose that Φ is a satisfiable set of sentences. We apply the proof of the Completeness Lemma (Lemma 2.43), treating $=$ as any binary relation, replacing Φ by $\Phi' = \Phi \cup \mathcal{E}_{\mathcal{L}}$, and taking $\Gamma \longrightarrow \Delta$ to be the empty sequent (always false). In this case $\Gamma \longrightarrow \Delta$ is not a logical consequence of Φ', so the proof constructs a term model $\mathcal{M}$ satisfying Φ' (see page 26). This structure has a countable universe M consisting of all the $\mathcal{L}$-terms. By the proof of the Equality Theorem, we can pass to equivalence classes and construct a countable structure $\hat{\mathcal{M}}$ which satisfies Φ (and interprets $=$ as true equality). $\dashv$

As an application of the above theorem, we conclude that no countable set of first-order sentences can characterize the real numbers. This is because if the field of real numbers forms a model for the sentences, then there will also be a countable model for the sentences. But the countable model cannot be isomorphic to the field of reals, because there are uncountably many real numbers.

Theorem 2.61 (Compactness Theorem). *If Φ is an unsatisfiable set of predicate calculus formulas then some finite subset of Φ is unsatisfiable.*

(See also the three alternative forms in Theorem 2.16.)

Proof. First note that we may assume that Φ is a set of sentences, by replacing the free variables in Φ by distinct new constant symbols. The resulting set of sentences is satisfiable iff the original set of formulas is satisfiable. Since Φ is unsatisfiable iff the empty sequent $\longrightarrow$ is a logical consequence of Φ, and since **LK**-Ψ proofs are finite, the theorem now follows from Corollary 2.53. $\dashv$

Theorem 2.62. *Suppose $\mathcal{L}$ has only finitely many function and predicate symbols (or recursively enumerable sets of function and predicate symbols.) Then the set of valid $\mathcal{L}$-sentences is recursively enumerable. Similarly for the set of unsatisfiable $\mathcal{L}$-sentences.*

Concerning this theorem, a set is *recursively enumerable* if there is an algorithm for enumerating its members. To enumerate the valid formulas, enumerate finite **LK** proofs. To enumerate the unsatisfiable formulas, note that A is unsatisfiable iff $\neg A$ is valid.

Exercise 2.63 (Application of Compactness). *Show that if a set Φ of sentences has arbitrarily large finite models, then Φ has an infinite model. (Hint: For each n construct a sentence A_n which is satisfiable in any universe with n or more elements but not satisfiable in any universe with fewer than n elements.)*

2E. The Herbrand Theorem

The Herbrand Theorem provides a complete method for proving the unsatisfiability of a set of universal sentences. It can be extended to a complete method for proving the unsatisfiability of an arbitrary set of first-order sentences by first converting the sentences to universal sentences by introducing "Skolem" functions for the existentially quantified variables. This forms the basis of the resolution proof method, which is used extensively by automated theorem provers.

Definition 2.64. *A formula A is* quantifier-free *if A has no occurrence of either of the quantifiers $\forall$ or $\exists$. A $\forall$-sentence is a sentence of the form $\forall x_1 \ldots \forall x_k B$ where $k \geq 0$ and B is a quantifier-free formula. A* ground instance *of this sentence is a sentence of the form $B(t_1/x_1)(t_2/x_2)\ldots(t_k/x_k)$, where $t_1, \ldots, t_k$ are ground terms (i.e., terms with no variables) from the underlying language.*

Notice that a ground instance of a $\forall$-sentence A is a logical consequence of A. Therefore if a set Φ_0 of ground instances of A is unsatisfiable, then A is unsatisfiable. The Herbrand Theorem implies a form of the converse.

Definition 2.65 ($\mathcal{L}$-Truth Assignment). *An* $\mathcal{L}$-truth assignment *(or just* truth assignment*) is a map*

$$\tau : \{\mathcal{L}\text{-atomic formulas}\} \to \{T, F\}$$

We extend τ to the set of all quantifier-free $\mathcal{L}$-formulas by applying the usual rules for propositional connectives.

The above definition of truth assignment is the same as in the propositional calculus, except now we take the set of atoms to be the set of $\mathcal{L}$-atomic formulas. Thus we say that a set Φ_0 of quantifier-free formulas is *propositionally unsatisfiable* if no truth assignment satisfies every member of Φ_0.

Lemma 2.66. *If a set Φ_0 of quantifier-free sentences is propositionally unsatisfiable, then Φ_0 is unsatisfiable (in the first-order sense).*

Proof. We prove the contrapositive: Suppose that Φ_0 is satisfiable, and let $\mathcal{M}$ be a first-order structure which satisfies Φ_0. Then $\mathcal{M}$ induces a truth assignment τ by the definition $B^\tau = T$ iff $\mathcal{M} \models B$ for each atomic sentence B. Then $B^\tau = T$ iff $\mathcal{M} \models B$ for each quantifier-free sentence B, so τ satisfies Φ_0. $\dashv$

We can now state our simplified proof method, which applies to sets of $\forall$-sentences: Simply take ground instances of sentences in Φ together with the equality axioms $\mathcal{E}_{\mathcal{L}}$ until a propositionally unsatisfiable set Φ_0 is found. The method does not specify how to check for propositional unsatisfiability: any method (such as truth tables) for that will do. Notice that by propositional compactness, it is sufficient to consider finite sets Φ_0 of ground instances. The Herbrand Theorem states that this method is sound and complete.

THEOREM 2.67 (Herbrand Theorem, Form 1). *Suppose that the underlying language $\mathcal{L}$ has at least one constant symbol, and let Φ be a set of $\forall$-sentences. Then Φ is unsatisfiable iff some finite set Φ_0 of ground instances of sentences in $\Phi \cup \mathcal{E}_{\mathcal{L}}$ is propositionally unsatisfiable.*

COROLLARY 2.68 (Herbrand Theorem, Form 2). *Let Φ be a set of $\forall$-sentences and let $A(\vec{x}, y)$ be a quantifier-free formula with all free variables indicated such that*

$$\Phi \models \forall\vec{x}\exists y A(\vec{x}, y)$$

Then there exist finitely many terms $t_1(\vec{x}), \ldots, t_k(\vec{x})$ in the vocabulary of Φ and $A(\vec{x}, y)$ such that

$$\Phi \models \forall\vec{x}\big(A(\vec{x}, t_1(\vec{x})) \vee \cdots \vee A(\vec{x}, t_k(\vec{x}))\big)$$

We will use Form 2 in later chapters to prove "witnessing theorems" for various theories. The idea is that one of the terms $t_1(\vec{x}), \ldots, t_k(\vec{x})$ "witnesses" the existential quantifier $\exists y$ in the formula $\forall\vec{x}\exists y A(\vec{x}, y)$.

EXERCISE 2.69. *Prove Form 2 from Form 1. Start by showing that under the hypotheses of Form 2, $\Phi \cup \{\forall y \neg A(\vec{c}, y)\}$ is unsatisfiable, where $\vec{c}$ is a list of new constants.*

EXAMPLE 2.70. *Let c be a constant symbol, and let*

$$\Phi = \{\forall x(Px \supset Pfx), Pc, \neg Pffc\}.$$

Then the set $\mathcal{H}$ of ground terms is $\{c, fc, ffc, \ldots\}$. We can take the set Φ_0 of ground instances to be

$$\Phi_0 = \{(Pc \supset Pfc), (Pfc \supset Pffc), Pc, \neg Pffc\}.$$

Then Φ_0 is propositionally unsatisfiable, so Φ is unsatisfiable.

PROOF OF THE SOUNDNESS DIRECTION OF THEOREM 2.67. If Φ_0 is propositionally unsatisfiable, then Φ is unsatisfiable. This follows easily from Lemma 2.66, since Φ_0 is a logical consequence of Φ. $\dashv$

PROOF OF THE COMPLETENESS DIRECTION OF THEOREM 2.67. This follows from the Anchored **LK** Completeness Theorem (see Exercise 2.72 below). Here we give a direct proof.

We prove the contrapositive: If every finite set of ground instances of $\Phi \cup \mathcal{E}_{\mathcal{L}}$ is propositionally satisfiable, then Φ is satisfiable. By Corollary 2.52, we may ignore the special status of $=$.

Let Φ_0 be the set of *all* ground instances of $\Phi \cup \mathcal{E}_{\mathcal{L}}$ (using ground terms from $\mathcal{L}$). Assuming that every finite subset of Φ_0 is propositionally satisfiable, it follows from propositional compactness (Theorem 2.16, Form 3) that the entire set Φ_0 is propositionally satisfiable. Let τ be a truth assignment which satisfies Φ_0. We use τ to construct an $\mathcal{L}$-structure $\mathcal{M}$ which satisfies Φ. We use a term model, similar to that used in the proof of the Completeness Lemma (Definition 2.45).

Let the universe M of $\mathcal{M}$ be the set $\mathcal{H}$ of all ground $\mathcal{L}$-terms.

For each n-ary function symbol f define

$$f^{\mathcal{M}}(t_1, \ldots, t_n) = ft_1 \ldots t_n.$$

(In particular, $c^{\mathcal{M}} = c$ for each constant c, and it follows by induction that $t^{\mathcal{M}} = t$ for each ground term t.)

For each n-ary predicate symbol P of $\mathcal{L}$, define

$$P^{\mathcal{M}} = \{\langle t_1, \ldots, t_n \rangle : (Pt_1 \ldots t_n)^{\tau} = T\}$$

This completes the specification of $\mathcal{M}$. It follows easily by structural induction that $\mathcal{M} \models B$ iff $B^{\tau} = T$, for each quantifier-free $\mathcal{L}$-sentence B with no variables. Thus $\mathcal{M} \models B$ for every ground instance B of any sentence in Φ. Since every member of Φ is a $\forall$-sentence, and since the elements of the universe are precisely the ground terms, it follows that $\mathcal{M}$ satisfies every member of Φ. (A formal proof would use the Basic Semantic Definition (Definition 2.27) and the Substitution Theorem (Theorem 2.33). $\dashv$

Exercise 2.71. *Show (from the proof of the Herbrand Theorem) that a satisfiable set of $\forall$ sentences without $=$ and without function symbols except the constants $c_1, \ldots, c_n$ for $n \geq 1$ has a model with exactly n elements in the universe. Give an example showing that $n-1$ elements would not suffice in general.*

Exercise 2.72. *Show that the completeness direction of the Herbrand Theorem (Form 1) follows from the Anchored* **LK** *Completeness Theorem (with equality, Definition 2.57 and Theorem 2.58) and the following syntactic lemma.*

Lemma 2.73. *Let Φ be a set of formulas closed under substitution of terms for variables. Let π be an* **LK**-Φ *proof in which all formulas are quantifier-free, let t be a term and let b be a variable, and let $\pi(t/b)$ be the result of replacing every occurrence of b in π by t. Then $\pi(t/b)$ is an* **LK**-Φ *proof.*

Definition 2.74 (Prenex Form). *We say that a formula A is in* prenex form *if A has the form $\mathsf{Q}_1 x_1 \ldots \mathsf{Q}_n x_n B$, where each Q_i is either $\forall$ or $\exists$, and B is a quantifier-free formula.*

Theorem 2.75 (Prenex Form Theorem). *There is a simple procedure which, given a formula A, produces an equivalent formula A' in prenex form.*

Proof. First rename all quantified variables in A so that they are all distinct (see the remark on page 19). Now move all quantifiers out past the connectives $\wedge, \vee, \neg$ by repeated use of the equivalences below. (Recall that by the Formula Replacement Theorem (Theorem 2.34), we can replace a subformula in A by an equivalent formula and the result is equivalent to A.)

Note. In each of the following equivalences, we must assume that x does not occur free in C.

$$\begin{array}{rlrl}
(\forall x B \wedge C) & \Longleftrightarrow \forall x(B \wedge C) & (\forall x B \vee C) & \Longleftrightarrow \forall x(B \vee C) \\
(C \wedge \forall x B) & \Longleftrightarrow \forall x(C \wedge B) & (C \vee \forall x B) & \Longleftrightarrow \forall x(C \vee B) \\
(\exists x B \wedge C) & \Longleftrightarrow \exists x(B \wedge C) & (\exists x B \vee C) & \Longleftrightarrow \exists x(B \vee C) \\
(C \wedge \exists x B) & \Longleftrightarrow \exists x(C \wedge B) & (C \vee \exists x B) & \Longleftrightarrow \exists x(C \vee B) \\
\neg \forall x B & \Longleftrightarrow \exists x \neg B & \neg \exists x B & \Longleftrightarrow \forall x \neg B
\end{array}$$

$\dashv$

2F. Notes

Our treatment of **PK** in sections 2A.1 and 2A.2 is adapted from Section 1.2 of [19].

Sections 2B.1 to 2B.3 roughly follow Sections 2.1 and 2.2 of [19]. However an important difference is that the definition of $\Phi \models A$ in [19] treats free variables as though they are universally quantified, but our definition does not.

The proof of the Anchored **LK** Completeness Theorem outlined in Exercise 2.48 grew out of discussions with S. Buss.

Chapter 3

PEANO ARITHMETIC AND ITS SUBSYSTEMS

In this chapter we discuss Peano Arithmetic and some of its subsystems. We focus on $\mathbf{I\Delta_0}$, which plays an essential role in the development of the theories in later chapters: All (two-sorted) theories introduced in this book extend $\mathbf{V}^0$, a conservative extension of $\mathbf{I\Delta_0}$. At the end of the chapter we briefly discuss Buss's hierarchy $\mathbf{S}_2^1 \subseteq \mathbf{T}_2^1 \subseteq \mathbf{S}_2^2 \dots$. These single-sorted theories establish a link between bounded arithmetic and the polynomial time hierarchy, and have played a central role in the study of bounded arithmetic. In later chapters we introduce their two-sorted versions, including $\mathbf{V}^1$, a theory that characterizes $\mathbf{P}$. The theories considered in this chapter are singled-sorted, and the intended domain is $\mathbb{N} = \{0, 1, 2, ...\}$.

Subsection 3C.3 shows that the relation $y = 2^x$ is definable by a bounded formula in the language of $\mathbf{I\Delta_0}$, and in Section 3D this is used to show that bounded formulas represent precisely the relations in the Linear Time Hierarchy (**LTH**).

3A. Peano Arithmetic

Definition 3.1. *A* theory *over a vocabulary $\mathcal{L}$ is a set $\mathcal{T}$ of formulas over $\mathcal{L}$ which is closed under logical consequence and universal closure.*

We often specify a theory by a set Γ of *axioms* for $\mathcal{T}$, where Γ is a set of $\mathcal{L}$-formulas. In that case

$$\mathcal{T} = \{A \mid A \text{ is an } \mathcal{L}\text{-formula and } \forall\Gamma \models A\}$$

Here $\forall\Gamma$ is the set of universal closures of formulas in Γ (Definition 2.41).

Note that it is more usual to require that a theory be a set of sentences, rather than formulas. Our version of a usual theory $\mathcal{T}$ is $\mathcal{T}$ together with all formulas (with free variables) which are logical consequences of $\mathcal{T}$. Recall $\forall A \models A$, for any formula A.

Notation. We sometimes write $\mathcal{T} \vdash A$ to mean $A \in \mathcal{T}$. If $\mathcal{T} \vdash A$ we say that A is a *theorem* of $\mathcal{T}$.

The theories that we consider in this section have the language of arithmetic

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

as the underlying vocabulary (Definition 2.22). Recall that the *standard model* $\underline{\mathbb{N}}$ for $\mathcal{L}_A$ has universe $M = \mathbb{N}$ and $0, 1, +, \cdot, =, \leq$ get their standard meanings in $\mathbb{N}$.

Notation. $t < u$ stands for $(t \leq u \wedge t \neq u)$. For each $n \in \mathbb{N}$ we define a term $\underline{n}$ called the *numeral for* n inductively as follows:

$$\underline{0} = 0,\ \underline{1} = 1, \qquad \text{for } n \geq 1,\ \underline{n+1} = (\underline{n} + 1)$$

For example, $\underline{3}$ is the term $((1 + 1) + 1)$. In general, the term $\underline{n}$ is interpreted as n in the standard model.

Definition 3.2. **TA** *(True Arithmetic) is the theory over $\mathcal{L}_A$ consisting of all formulas whose universal closures are true in the standard model:*

$$\mathbf{TA} = \{A \mid \underline{\mathbb{N}} \models \forall A\}$$

It follows from Gödel's Incompleteness Theorem that **TA** has no computable set of axioms. The theories we define below are all subtheories of **TA** with nice, computable sets of axioms.

Note that by Definition 2.25, = is interpreted as true equality in all $\mathcal{L}_A$-structures, and hence we do not need to include the Equality Axioms in our list of axioms. (Of course **LK** proofs still need equality axioms: see Definition 2.55 and Corollaries 2.52, 2.53).

We start by listing nine "basic" quantifier-free formulas **B**1, ..., **B**8 and **C**, which comprise the axioms for our basic theory. See Figure 1 below.

B1. $x + 1 \neq 0$	**B**5. $x \cdot 0 = 0$
B2. $x + 1 = y + 1 \supset x = y$	**B**6. $x \cdot (y + 1) = (x \cdot y) + x$
B3. $x + 0 = x$	**B**7. $(x \leq y \wedge y \leq x) \supset x = y$
B4. $x + (y + 1) = (x + y) + 1$	**B**8. $x \leq x + y$
C. $0 + 1 = 1$	

Figure 1. 1-**BASIC**

These axioms provide recursive definitions for + and $\cdot$, and some basic properties of $\leq$. Axiom **C** is not necessary in the presence of induction, since it then follows from the theorem $0+x = x$ (see Example 3.8, **O**2). However we put it in so that $\forall\mathbf{B}1, \ldots, \forall\mathbf{B}8, \forall\mathbf{C}$ alone imply all true quantifier-free sentences over $\mathcal{L}_A$.

Lemma 3.3. *If φ is a quantifier-free sentence of $\mathcal{L}_A$, then*

$$\mathbf{TA} \vdash \varphi \qquad \textit{iff} \qquad 1\text{-}\mathbf{BASIC} \vdash \varphi$$

Proof. The direction $\Longleftarrow$ holds because the axioms of 1-**BASIC** are valid in $\underline{\mathbb{N}}$.

For the converse, we start by proving by induction on m that if $m < n$, then 1-**BASIC** $\vdash \underline{m} \neq \underline{n}$. The base case follows from **B**1 and **C**, and the induction step follows from **B**2 and **C**.

Next we use **B**3, **B**4 and **C** to prove by induction on n that if $m+n = k$, then 1-**BASIC** $\vdash \underline{m} + \underline{n} = \underline{k}$. Similarly we use **B**5, **B**6 and **C** to prove that if $m \cdot n = k$ then 1-**BASIC** $\vdash \underline{m} \cdot \underline{n} = \underline{k}$.

Now we use the above results to prove by structural induction on t, that if t is any term without variables, and t is interpreted as n in the standard model $\mathbb{N}$, then 1-**BASIC** $\vdash t = \underline{n}$.

It follows from the above results that if t and u are any terms without variables, then **TA** $\vdash t = u$ implies 1-**BASIC** $\vdash t = u$, and **TA** $\vdash t \neq u$ implies 1-**BASIC** $\vdash t \neq u$.

Consequently, if $m \leq n$, then for some k, 1-**BASIC** $\vdash \underline{n} = \underline{m} + \underline{k}$, and hence by **B**8, 1-**BASIC** $\vdash \underline{m} \leq \underline{n}$. Also if not $m \leq n$, then $n < m$, so by the above 1-**BASIC** $\vdash \underline{m} \neq \underline{n}$ and 1-**BASIC** $\vdash \underline{n} \leq \underline{m}$, so by **B**7, 1-**BASIC** $\vdash \neg \underline{m} \leq \underline{n}$.

Finally let φ be any quantifier-free sentence. We prove by structural induction on φ that if **TA** $\vdash \varphi$ then 1-**BASIC** $\vdash \varphi$ and if **TA** $\vdash \neg\varphi$ then 1-**BASIC** $\vdash \neg\varphi$. For the base case φ is atomic and has one of the forms $t = u$ or $t \leq u$, so the base case follows from the above. The induction step involves the three cases $\wedge$, $\vee$, and $\neg$, which are immediate. $\dashv$

Definition 3.4 (Induction Scheme). *If Φ is a set of formulas, then* Φ-**IND** *axioms are the formulas*

$$[\varphi(0) \wedge \forall x,\ \varphi(x) \supset \varphi(x+1)] \supset \forall z \varphi(z) \tag{10}$$

where φ is a formula in Φ. Note that $\varphi(x)$ is permitted to have free variables other than x.

Definition 3.5 (Peano Arithmetic). *The theory* **PA** *has as axioms* **B**1*, ... ,* **B**8*, together with the* Φ-**IND** *axioms, where Φ is the set of all $\mathcal{L}_A$ formulas.*

(As we noted earlier, **C** is provable from the other axioms in the presence of induction.)

PA is a powerful theory capable of formalizing the major theorems of number theory. We define subsystems of **PA** by restricting the induction axiom to certain sets of formulas. We use the following notation.

Definition 3.6 (Bounded Quantifiers). *If the variable x does not occur in the term t, then $\exists x \leq tA$ stands for $\exists x(x \leq t \wedge A)$, and $\forall x \leq tA$ stands for $\forall x(x \leq t \supset A)$. Quantifiers that occur in this form are said to be* bounded*, and a* bounded formula *is one in which every quantifier is bounded.*

Notation. Let $\exists \vec{x}$ stand for $\exists x_1 \exists x_2 ... \exists x_k$, $k \geq 0$.

Definition 3.7 (**IOPEN**, $\mathbf{I}\Delta_0$, $\mathbf{I}\Sigma_1$). **OPEN** *is the set of open (i.e., quantifier-free) formulas;* Δ_0 *is the set of bounded formulas; and* Σ_1 *is the set of formulas of the form* $\exists\vec{x}\varphi$*, where* φ *is bounded and* $\vec{x}$ *is a possibly empty vector of variables. The theories* **IOPEN**, $\mathbf{I}\Delta_0$*, and* $\mathbf{I}\Sigma_1$ *are the subsystems of* **PA** *obtained by restricting the induction scheme so that* Φ *is* **OPEN**, Δ_0*, and* Σ_1*, respectively.*

Note that the underlying language of the theories defined above is $\mathcal{L}_A$.

Example 3.8. *The following formulas (and their universal closures) are theorems of* **IOPEN***:*
O1. $(x + y) + z = x + (y + z)$ *(Associativity of +)*
O2. $x + y = y + x$ *(Commutativity of +)*
O3. $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$ *(Distributive law)*
O4. $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ *(Associativity of* $\cdot$*)*
O5. $x \cdot y = y \cdot x$ *(Commutativity of* $\cdot$*)*
O6. $x + z = y + z \supset x = y$ *(Cancellation law for +)*
O7. $0 \le x$
O8. $x \le 0 \supset x = 0$
O9. $x \le x$
O10. $x \ne x + 1$

Proof. **O**1. induction on z
O2. induction on y, first establishing the special cases $y = 0$ and $y = 1$
O3. induction on z
O4. induction on z, using **O**3
O5. induction on y, after establishing $(y+1)\cdot x = y\cdot x+x$ by induction on x
O6. induction on z
O7. **B**8, **O**2, **B**3
O8. **O**7, **B**7
O9. **B**8, **B**3
O10. induction on x and **B**2. $\dashv$

Recall that $x < y$ stands for $(x \le y \wedge x \ne y)$

Example 3.9. *The following formulas (and their universal closures) are theorems of* $\mathbf{I}\Delta_0$*:*
D*1.* $x \ne 0 \supset \exists y \le x(x = y + 1)$ *(Predecessor)*
D*2.* $\exists z(x + z = y \vee y + z = x)$
D*3.* $x \le y \leftrightarrow \exists z(x + z = y)$
D*4.* $(x \le y \wedge y \le z) \supset x \le z$ *(Transitivity)*
D*5.* $x \le y \vee y \le x$ *(Total order)*
D*6.* $x \le y \leftrightarrow x + z \le y + z$
D*7.* $x \le y \supset x \cdot z \le y \cdot z$
D*8.* $x \le y + 1 \leftrightarrow (x \le y \vee x = y + 1)$ *(Discreteness 1)*
D*9.* $x < y \leftrightarrow x + 1 \le y$ *(Discreteness 2)*
D*10.* $x \cdot z = y \cdot z \wedge z \ne 0 \supset x = y$ *(Cancellation law for* $\cdot$*)*

Proof. **D**1. Induction on x
D2. Induction on x. Base case: **B**2, **O**2. Induction step: **B**3, **B**4, **D**1.
D3. $\Longrightarrow$: **D**2, **B**3 and **B**7; $\Longleftarrow$: **B**8.
D4. **D**3, **O**1.
D5. **D**2, **B**8.
D6. $\Longrightarrow$: **D**3, **O**1, **O**2; $\Longleftarrow$: **D**3, **O**6.
D7. **D3** and algebra (**O**1, ... , **O**8).
D8. $\Longrightarrow$: **D**3, **D**1, and algebra; $\Longleftarrow$: **O**9, **B**8, **D**4.
D9. $\Longrightarrow$: **D**3, **D**1, and algebra; $\Longleftarrow$: **D**3 and algebra.
D10. Exercise. $\dashv$

Taken together, these results show that all models of $\mathbf{I}\Delta_0$ are commutative discretely-ordered semi-rings.

Exercise 3.10. *Show that* $\mathbf{I}\Delta_0$ *proves the division theorem:*

$$\mathbf{I}\Delta_0 \vdash \forall x \forall y(0 < x \supset \exists q\, \exists r < x,\ y = x \cdot q + r)$$

It follows from Gödel's Incompleteness Theorem that there is a bounded formula $\varphi(x)$ such that $\forall x\varphi(x)$ is true but $\mathbf{I}\Delta_0 \not\vdash \forall x\varphi(x)$. However if φ is a true sentence in which all quantifiers are bounded, then intuitively φ expresses information about only finitely many tuples of numbers, and in this case we can show $\mathbf{I}\Delta_0 \vdash \varphi$. The same applies more generally to true Σ_1 sentences φ.

Lemma 3.11. *If* φ *is a* Σ_1 *sentence, then* $\mathbf{TA} \vdash \varphi$ *iff* $\mathbf{I}\Delta_0 \vdash \varphi$.

Proof. The direction $\Longleftarrow$ follows because all axioms of $\mathbf{I}\Delta_0$ are true in the standard model.

For the converse, we prove by structural induction on bounded sentences φ that if $\mathbf{TA} \vdash \varphi$ then $\mathbf{I}\Delta_0 \vdash \varphi$, and if $\mathbf{TA} \vdash \neg\varphi$ then $\mathbf{I}\Delta_0 \vdash \neg\varphi$. The base case is φ is atomic, and this follows from Lemma 3.3. For the induction step, the cases $\vee$, $\wedge$, and $\neg$ are immediate. The remaining cases are φ is $\forall x \le t\psi(x)$ and φ is $\exists x \le t\psi(x)$, where t is a term without variables, and $\psi(x)$ is a bounded formula with no free variable except possibly x. These cases follow from Lemma 3.3 and Lemma 3.12 below.

Now suppose that φ is a true Σ_1 sentence of the form $\exists\vec{x}\psi(\vec{x})$, where $\psi(\vec{x})$ is a bounded formula. Then $\psi(\vec{n})$ is a true bounded sentence for some numerals $\underline{n}_1, \ldots, \underline{n}_k$, so $\mathbf{I}\Delta_0 \vdash \psi(\vec{n})$. Hence $\mathbf{I}\Delta_0 \vdash \varphi$. $\dashv$

Lemma 3.12. *For each* $n \in \mathbb{N}$,

$$\mathbf{I}\Delta_0 \vdash x \le \underline{n} \leftrightarrow (x = \underline{0} \vee x = \underline{1} \vee ... \vee x = \underline{n})$$

Proof. Induction on n. The base case $n = 0$ follows from **O**7 and **O**8, and the induction step follows from **D**8. $\dashv$

3A.0.1. *Minimization.*

Definition 3.13 (Minimization). *The minimization axioms (or* least number principle *axioms) for a set* Φ *of formulas are denoted* Φ-**MIN** *and consist of the formulas*

$$\exists z\varphi(z) \supset \exists y\big(\varphi(y) \wedge \neg\exists x(x < y \wedge \varphi(x))\big)$$

where φ is a formula in Φ.

THEOREM 3.14. $\mathbf{I}\Delta_0$ *proves* Δ_0-**MIN**.

PROOF. The contrapositive of the minimization axiom for $\varphi(z)$ follows from the induction axiom for the bounded formula $\psi(z) \equiv \forall y \leq z(\neg\varphi(y))$. ⊣

EXERCISE 3.15. *Show that* $\mathbf{I}\Delta_0$ *can be alternatively axiomatized by* **B***1, ..., * **B***8,* **O***10 (Example 3.8),* **D***1 (Example 3.9), and the axiom scheme* Δ_0-**MIN**.

3A.0.2. *Bounded Induction Scheme.* The Δ_0-**IND** scheme of $\mathbf{I}\Delta_0$ can be replaced by the following *bounded induction scheme* for Δ_0 formulas, i.e.,

$$\big(\varphi(0) \wedge \forall x < z(\varphi(x) \supset \varphi(x+1))\big) \supset \varphi(z) \tag{11}$$

where $\varphi(x)$ is any Δ_0 formula. (Note that the **IND** formula (10) for $\varphi(x)$ is a logical consequence of the universal closure of this.)

EXERCISE 3.16. *Prove that* $\mathbf{I}\Delta_0$ *remains the same if the* Δ_0-**IND** *scheme is replaced by the above bounded induction scheme for* Δ_0 *formulas. (It suffices to show that the new scheme is provable in* $\mathbf{I}\Delta_0$*.)*

3A.0.3. *Strong Induction Scheme.* The *strong induction axiom* for a formula $\varphi(x)$ is the following formula:

$$\forall x\big((\forall y < x\varphi(y)) \supset \varphi(x)\big) \supset \forall z\varphi(z) \tag{12}$$

EXERCISE 3.17. *Show that* $\mathbf{I}\Delta_0$ *proves the strong induction axiom scheme for* Δ_0 *formulas.*

3B. Parikh's Theorem

By the results in the previous section, $\mathbf{I}\Delta_0$ can be axiomatized by a set of bounded formulas. We say that it is a *polynomial-bounded theory*, a concept we will now define.

In general, a theory $\mathcal{T}$ may have symbols other than those in $\mathcal{L}_A$. We say that a term $t(\vec{x})$ is a *bounding term* for a function symbol $f(\vec{x})$ in $\mathcal{T}$ if

$$\mathcal{T} \vdash \forall\vec{x}\; f(\vec{x}) \leq t(\vec{x}) \tag{13}$$

We say that f is *polynomially bounded* (or just *p-bounded*) in $\mathcal{T}$ if it has a bounding term in the language $\mathcal{L}_A$.

Exercise 3.18. *Let $\mathcal{T}$ be an extension of $\mathbf{I\Delta_0}$ and let $\mathcal{L}$ be the vocabulary of $\mathcal{T}$. Suppose that the functions of $\mathcal{L}$ are polynomially bounded in $\mathcal{T}$. Show that for each $\mathcal{L}$-term $s(\vec{x})$, there is an $\mathcal{L}_A$-term $t(\vec{x})$ such that*

$$\mathcal{T} \vdash \forall \vec{x}\ s(\vec{x}) \leq t(\vec{x}).$$

Suppose that a theory $\mathcal{T}$ is an extension of $\mathbf{I\Delta_0}$. We can still talk about bounded formulas φ in $\mathcal{T}$ using the same definition (Definition 3.6) as before, but now φ may have function and predicate symbols not in the vocabulary $[0, 1, +, \cdot; =, \leq]$ of $\mathbf{I\Delta_0}$, and in particular the terms t bounding the quantifiers $\exists x \leq t$ and $\forall x \leq t$ may have extra function symbols. Note that by the exercise above, in the context of polynomial-bounded theories (defined below) we may assume without loss of generality that the bounding terms are $\mathcal{L}_A$-terms.

Definition 3.19 (Polynomial-Bounded Theory). *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L}$. Then $\mathcal{T}$ is a* polynomial-bounded theory *(or just* p-bounded theory*) if (i) it extends $\mathbf{I\Delta_0}$; (ii) it can be axiomatized by a set of bounded formulas; and (iii) each function $f \in \mathcal{L}$ is polynomially bounded in $\mathcal{T}$.*

Note that $\mathbf{I\Delta_0}$ is a polynomial-bounded theory.
Theories which satisfy (ii) are often called *bounded theories.*

Theorem 3.20 (Parikh's Theorem). *If $\mathcal{T}$ is a polynomial-bounded theory and $\varphi(\vec{x}, y)$ is a bounded formula with all free variables displayed such that $\mathcal{T} \vdash \forall\vec{x}\exists y\varphi(\vec{x}, y)$, then there is a term t involving only variables in $\vec{x}$ such that $\mathcal{T}$ proves $\forall\vec{x}\exists y \leq t\varphi(\vec{x}, y)$.*

It follows from Exercise 3.18 that the bounding term t can be taken to be an $\mathcal{L}_A$-term. In fact, Parikh's Theorem can be generalized to say that if φ is a bounded formula and $\mathcal{T} \vdash \exists\vec{y}\varphi$, then there are $\mathcal{L}_A$-terms $t_1, ..., t_k$ not involving any variable in $\vec{y}$ or any variable not occurring free in φ such that $\mathcal{T}$ proves $\exists y_1 \leq t_1 ... \exists y_k \leq t_k\varphi$. This follows from the above remark, and the following lemma.

Lemma 3.21. *Let $\mathcal{T}$ be an extension of $\mathbf{I\Delta_0}$. Let z be a variable distinct from $y_1, ..., y_k$ and not occurring in φ. Then*

$$\mathcal{T} \vdash \exists\vec{y}\varphi \leftrightarrow \exists z\exists y_1 \leq z ... \exists y_k \leq z\ \varphi$$

Exercise 3.22. *Give a careful proof of the above lemma, using the theorems of $\mathbf{I\Delta_0}$ described in Example 3.9.*

In section 3C.3 we will show how to represent the relation $y = 2^x$ by a bounded formula $\varphi_{exp}(x, y)$. It follows immediately from Parikh's Theorem that

$$\mathbf{I\Delta_0} \nvdash \forall x\exists y\varphi_{exp}(x, y)$$

On the other hand $\mathbf{PA}$ easily proves the $\exists y\varphi_{exp}(x, y)$ by induction on x. Therefore $\mathbf{I\Delta_0}$ is a proper sub-theory of $\mathbf{PA}$.

Our proof of Parikh's Theorem will be based on the Anchored **LK** Completeness Theorem with Equality (2.58). Let $\mathcal{T}$ be a polynomial-bounded theory and $\forall\vec{x}\exists y\varphi(\vec{x}, y)$ a theorem of $\mathcal{T}$. We will look into an anchored proof of $\forall\vec{x}\exists y\varphi(\vec{x}, y)$ and show that a term t (not involving y) can be constructed so that $\forall\vec{x}\exists y \leq t\varphi(\vec{x}, y)$ is also a theorem of $\mathcal{T}$. In order to apply the Anchored **LK** Completeness Theorem (with Equality), we need to find an axiomatization of $\mathcal{T}$ which is closed under substitution of terms for variables. Note that $\mathcal{T}$ is already axiomatized by a set of bounded formulas (Definition 3.19). The desired axiomatization of $\mathcal{T}$ is obtained by substituting terms for all the free variables. We will consider the example where $\mathcal{T}$ is $\mathbf{I}\Delta_0$. The general case is similar.

Recall that the axioms for $\mathbf{I}\Delta_0$ consist of **B**1–**B**8 (page 38) and the Δ_0-**IND** scheme, which can be replaced by the Bounded Induction Scheme (11).

DEFINITION 3.23 ($\mathbf{ID}_0$). $\mathbf{ID}_0$ *is the set of all term substitution instances of* **B***1*–**B***8 and the Bounded Induction Scheme, where now the terms contain only "free" variables* $a, b, c, \ldots$.

Note that all formulas in $\mathbf{ID}_0$ are bounded.

For example $(c \cdot b) + 1 \neq 0$ is an instance of **B**1, and hence is in $\mathbf{ID}_0$. Also

$$a + 0 = 0 + a \wedge \forall x < b(a + x = x + a \supset a + (x + 1) = (x + 1) + a) \supset a + b = b + a$$

is an instance of (11) useful in proving the commutative law $a+b = b+a$ by induction on b, and is in $\mathbf{ID}_0$.

The following is an immediate consequence of the Anchored **LK** Completeness Theorem 2.58 and Derivational Soundness of **LK** (Theorem 2.42).

THEOREM 3.24 (**LK**-$\mathbf{ID}_0$ Adequacy Theorem). *Let A be an $\mathcal{L}_A$ formula satisfying the* **LK** *constraint that only variables $a, b, c, \ldots$ occur free and only $x, y, z, \ldots$ occur bound. Then $\mathbf{I}\Delta_0 \vdash A$ iff A has an anchored* **LK**-$\mathbf{ID}_0$ *proof.*

PROOF OF PARIKH'S THEOREM. Suppose that $\mathcal{T}$ is a polynomial-bounded theory which is axiomatized by a set of bounded axioms such that $\mathcal{T} \vdash \forall\vec{x}\exists y\varphi(\vec{x}, y)$, where $\varphi(\vec{x}, y)$ is a bounded formula. Let **T** be the set of all term substitution instances of the axioms of $\mathcal{T}$. By arguing as above in the case $\mathcal{T} = \mathbf{I}\Delta_0$, we can assume that $\longrightarrow \exists y\varphi(\vec{a}, y)$ has an anchored **LK-T** proof π. Further we may assume that π is in free variable normal form (Section 2B.4). By the sub-formula property of anchored proofs (2.59), every formula in every sequent of π is either bounded, or a substitution instance of the endsequent $\exists y\varphi(\vec{a}, y)$. But in fact the proof of the sub-formula property actually shows more: Every formula in π is either bounded or it must be syntactically identical to $\exists y\varphi(\vec{a}, y)$, and in the latter case it must occur in the consequent (right side) of a sequent. The reason is that once an unbounded quantifier is introduced in π, the resulting formula can never be altered by any rule, since cut formulas are restricted to the bounded formulas occurring in $\mathbf{T}$, and since no altered version of $\exists y\varphi(\vec{a}, y)$ occurs in the endsequent. (We may assume that $\exists y\varphi(\vec{a}, y)$ is an unbounded formula, since otherwise there is nothing to prove.)

We will convert π to an **LK-T** proof π' of $\exists y \le t\varphi(y)$ for some term t not containing y, by replacing each sequent $\mathcal{S}$ in π by a suitable sequent $\mathcal{S}'$, sometimes with a short derivation $\mathbf{D}(\mathcal{S})$ of $\mathcal{S}'$ inserted.

Here and in general we treat the cedents Γ and Δ of a sequent $\Gamma \longrightarrow \Delta$ as multi-sets in which the order of formulas is irrelevant. In particular we ignore instances of the exchange rule.

The conversion of a sequent $\mathcal{S}$ in π to $\mathcal{S}'$, and the associated derivation $\mathbf{D}(\mathcal{S})$, are defined by induction on the depth of $\mathcal{S}$ in π such that the following is satisfied:

Induction Hypothesis: If $\mathcal{S}$ has no occurrence of $\exists y\varphi$, then $\mathcal{S}' = \mathcal{S}$. If $\mathcal{S}$ has one or more occurrences of $\exists y\varphi$, then $\mathcal{S}'$ is a sequent which is the same as $\mathcal{S}$ except all occurrences of $\exists y\varphi$ are replaced by a single occurrence of $\exists y \le t\varphi$, where the term t depends on $\mathcal{S}$ and the placement of $\mathcal{S}$ in π. Further t satisfies the condition

(14) Every variable in t occurs free in the original sequent $\mathcal{S}$.

Thus the endsequent of π' has the form $\longrightarrow \exists y \le t\varphi$, where every variable in t occurs free in $\exists y\varphi$.

In order to maintain the condition (14) we use our assumption that π is in free variable normal form. Thus if the variable b occurs in t in the formula $\exists y \le t\varphi$, so b occurs in $\mathcal{S}$, then b cannot be eliminated from the descendants of $\mathcal{S}$ except by the rule $\forall$-right or $\exists$-left. These rules require special attention in the argument below.

We consider several cases, depending on the inference rule in π forming $\mathcal{S}$, and whether $\exists y\varphi$ is the principle formula of that rule.

Case I: $\mathcal{S}$ is the result of $\exists$-right applied to $\varphi(s)$ for some term s, so the inference has the form

(15)
$$\frac{\Gamma \longrightarrow \Delta, \varphi(s)}{\Gamma \longrightarrow \Delta, \exists y\varphi(y)}$$

where $\mathcal{S}$ is the bottom sequent. Suppose first that Δ has no occurrence of $\exists y\varphi$. Since $\mathbf{ID}_0$ proves $s \le s$ there is a short **LK-T** derivation of

(16)
$$\Gamma \longrightarrow \Delta, \exists y \le s\varphi(y)$$

from the top sequent. Let $\mathbf{D}(\mathcal{S})$ be that derivation and let $\mathcal{S}'$ be the sequent (16).

If Δ has one or more occurrence of $\exists y\varphi$, then by the induction hypothesis the top sequent $\mathcal{S}_1$ of (15) was converted to a sequent $\mathcal{S}_1'$ in which all of these occurrences have been replaced by a single occurrence of the form $\exists y \leq t\varphi$. We proceed as before, producing a sequent of the form

$$\Gamma \longrightarrow \Delta', \exists y \leq t\varphi, \exists y \leq s\varphi \tag{17}$$

Since $\mathbf{ID}_0$ proves the two sequents $\longrightarrow s \leq s+t$ and $\longrightarrow t \leq s+t$, it follows that $\mathcal{T}$ proves

$$\exists y \leq s\varphi \longrightarrow \exists y \leq (s+t)\varphi$$

and

$$\exists y \leq t\varphi \longrightarrow \exists y \leq (s+t)\varphi$$

We can use these and (17) with two cuts and a contraction to obtain a derivation of

$$\Gamma \longrightarrow \Delta', \exists y \leq (s+t)\varphi(y) \tag{18}$$

Let $\mathbf{D}(\mathcal{S})$ be this derivation and let $\mathcal{S}'$ be the resulting sequent (18).

Case II: $\mathcal{S}$ is the result of weakening right, which introduces $\exists y\varphi$. Thus the inference has the form

$$\frac{\Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, \exists y\varphi} \tag{19}$$

where $\mathcal{S}$ is the bottom sequent. If Δ does not contain $\exists y\varphi$, then define $\mathcal{S}'$ to be

$$\Gamma \longrightarrow \Delta, \exists y \leq 0\ \varphi$$

(introduced by weakening). If Δ contains one or more occurrences of $\exists y\varphi$, then take $\mathcal{S}' = \mathcal{S}_1'$, where $\mathcal{S}_1$ is the top sequent of (19).

Case III: $\mathcal{S}$ is the result of $\forall$-right or $\exists$-left. We consider the case $\exists$-left. The other case is similar and we leave it as an exercise. The new quantifier introduced must be bounded, since all formulas in π except $\exists y\varphi$ are bounded, and the latter must occur on the right. Thus the inference has the form

$$\frac{b \leq r \wedge \psi(b), \Gamma \longrightarrow \Delta}{\exists x \leq r\psi(x), \Gamma \longrightarrow \Delta} \tag{20}$$

where $\mathcal{S}$ is the bottom sequent. If Δ has no occurrence of $\exists y\varphi$, then define $\mathcal{S}' = \mathcal{S}$ and let $\mathbf{D}(\mathcal{S})$ be the derivation (20). Otherwise, by the induction hypothesis, the top sequent was converted to a sequent of the form

$$b \leq r \wedge \psi(b), \Gamma \longrightarrow \Delta', \exists y \leq s(b)\varphi(y) \tag{21}$$

Note that b may appear on the succedent and thus violate the Restriction of the $\exists$-left rule (page 20).

In order to apply the $\exists$-left rule (and continue to satisfy the condition (14)), we replace the bounding term $s(b)$ by an $\mathcal{L}_A$-term t that does not contain b. This is possible since the functions of $\mathcal{T}$ are polynomially bounded in $\mathcal{T}$. In particular, by Exercise 3.18, we know that there are $\mathcal{L}_A$-terms r', $s'(b)$ such that $\mathcal{T}$ proves both

$$r \leq r' \qquad \text{and} \qquad s(b) \leq s'(b)$$

Let $t = s'(r')$. Then by the monotonicity of $\mathcal{L}_A$-terms, $\mathcal{T}$ proves $b \leq r \longrightarrow s(b) \leq t$. Thus $\mathcal{T}$ proves

$$b \leq r, \exists y \leq s(b)\varphi(y) \longrightarrow \exists y \leq t\varphi(y)$$

(i.e., the above sequent has an **LK-T** derivation). From this and (21) applying cut with cut formula $\exists y \leq s(b)\varphi$ we obtain

$$b \leq r \wedge \psi(b), \Gamma \longrightarrow \Delta', \exists y \leq t\varphi(y)$$

where t does not contain b. We can now apply the $\exists$-left rule to obtain

$$\exists x \leq r\psi(x), \Gamma \longrightarrow \Delta', \exists y \leq t\varphi(y) \tag{22}$$

Let $\mathbf{D}(\mathcal{S})$ be this derivation and let $\mathcal{S}'$ be the resulting sequent (22).

Case IV: $\mathcal{S}$ results from a rule with two parents. Note that if this rule is cut, then the cut formula cannot be $\exists y\varphi$, because π is anchored. The only difficulty in converting $\mathcal{S}$ is that the two consequents Δ' and Δ'' of the parent sequents may have been converted to consequents with different bounded formulas $\exists y \leq t_1\varphi$ and $\exists y \leq t_2\varphi$. In this case proceed as in the second part of **Case I** to combine these two formulas to the single formula $\exists y \leq (t_1 + t_2)\varphi$.

Case V: All remaining cases. The inference is of the form derive $\mathcal{S}$ from the single sequent $\mathcal{S}_1$. Then take $\mathcal{S}'$ to be the result of applying the same rule in the same way to $\mathcal{S}'_1$, except in the case of contraction right when the principle formula is $\exists y\varphi$. In this case take $\mathcal{S}' = \mathcal{S}'_1$. $\dashv$

Exercise 3.25. *Work out the sub-case $\forall$-right in* **Case III**.

3C. Conservative Extensions of $\mathbf{I}\Delta_0$

In this section we occasionally present simple model-theoretic arguments, and the following standard definition from model theory is useful.

Definition 3.26 (Expansion of a Model). *Let $\mathcal{L}_1 \subseteq \mathcal{L}_2$ be vocabularies and let $\mathcal{M}_i$ be an $\mathcal{L}_i$ structure for $i = 1, 2$. We say $\mathcal{M}_2$ is an* expansion *of $\mathcal{M}_1$ if $\mathcal{M}_1$ and $\mathcal{M}_2$ have the same universe and the same interpretation for symbols in $\mathcal{L}_1$.*

3C.1. Introducing New Function and Predicate Symbols. In the following discussion we assume that all predicate and function symbols have a standard interpretation in the set $\mathbb{N}$ of natural numbers. A theory $\mathcal{T}$ which extends $\mathbf{I}\Delta_0$ has defining axioms for each predicate and function symbol in its vocabulary which ensure that they receive their standard interpretations in a model of $\mathcal{T}$ which is an expansion of the standard model $\underline{\mathbb{N}}$. We often use the same notation for both the function symbol and the function that it is intended to represent. For example, the predicate symbol P might be $Prime$, where $Prime(x)$ is intended to mean that x is a prime number. Or f might be LPD, where $LPD(x)$ is intended to mean the least prime number dividing x (or x if $x \leq 1$).

Notation (unique existence). $\exists! x\varphi(x)$ stands for $\exists x,\ \varphi(x) \wedge \forall y(\varphi(y) \supset x = y)$, where y is a new variable not appearing in $\varphi(x)$. $\exists! x \leq t\varphi(x)$, where t does not involve x, stands for

$$\exists x \leq t,\ \varphi(x) \wedge \forall y \leq t(\varphi(y) \supset x = y)$$

where y is a new variable not appearing in $\varphi(x)$ or t.

DEFINITION 3.27 (Definable Predicates and Functions). *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L}$, and let Φ be a set of $\mathcal{L}$-formulas.*
(a) *We say that a predicate symbol $P(\vec{x})$ not in $\mathcal{L}$ is* Φ-definable *in $\mathcal{T}$ if there is an $\mathcal{L}$-formula $\varphi(\vec{x})$ in Φ such that*

$$P(\vec{x}) \leftrightarrow \varphi(\vec{x}) \tag{23}$$

(b) *We say that a function symbol $f(\vec{x})$ not in $\mathcal{L}$ is Φ-definable in $\mathcal{T}$ if there is a formula $\varphi(\vec{x}, y)$ in Φ such that*

$$\mathcal{T} \vdash \forall \vec{x} \exists! y \varphi(\vec{x}, y), \tag{24}$$

and that

$$y = f(\vec{x}) \leftrightarrow \varphi(\vec{x}, y) \tag{25}$$

We say that (23) *is a* defining axiom *for $P(\vec{x})$ and* (25) *is a* defining axiom *for $f(\vec{x})$. We say that a symbol is* definable *in $\mathcal{T}$ if it is Φ-definable in $\mathcal{T}$ for some Φ.*

Although the choice of φ in the above definition is not uniquely determined by the predicate or function symbol, we will assume that a specific φ has been chosen, so we will speak of *the* defining axiom for the symbol.

For example, the defining axiom for the predicate $Prime(x)$ (in any theory whose vocabulary contains $\mathcal{L}_A$) might be

$$Prime(x) \leftrightarrow 1 < x \wedge \forall y < x \forall z < x(y \cdot z \neq x).$$

Notation. Note that Δ_0 and Σ_1 (Definition 3.7) are sets of $\mathcal{L}_A$-formulas. In general, given a language $\mathcal{L}$ the sets $\Delta_0(\mathcal{L})$ and $\Sigma_1(\mathcal{L})$ are defined as in Definition 3.7 but the formulas are from $\mathcal{L}$. In this case we require that the terms bounding the quantifiers are $\mathcal{L}_A$-terms.

In Definition 3.27, if $\Phi = \Delta_0(\mathcal{L})$ (resp. $\Phi = \Sigma_1(\mathcal{L})$) then we sometimes omit mention of $\mathcal{L}$ and simply say that the symbols P, f are Δ_0-definable (resp. Σ_1-definable) in $\mathcal{T}$.

In the case of functions, the choice $\Phi = \Sigma_1(\mathcal{L})$ plays a special role. A Σ_1-definable function in $\mathcal{T}$ is also called a *provably total function* in $\mathcal{T}$. It turns out that the provably total functions of $\mathbf{I}\Sigma_1$ are precisely the primitive recursive functions and of $\mathbf{S}_2^1$ (see Section 3E) the polytime functions. In Section 3D we will show that the provably total functions of $\mathbf{I}\Delta_0$ are precisely the functions of the Linear Time Hierarchy.

Exercise 3.28. *Suppose that the functions*

$$f(x_1, \dots, x_m) \text{ and } h_i(x_1, \dots, x_n) \text{ (for } 1 \le i \le m)$$

are Σ_1-definable in a theory $\mathcal{T}$. Show that the function $f(h_1(\vec{x}), \dots, h_m(\vec{x}))$ (where $\vec{x}$ stands for $x_1, \dots, x_n$) is also Σ_1-definable in $\mathcal{T}$. (In other words, show that Σ_1-definable functions are closed under composition.)

Definition 3.29 (Conservative Extension). *Suppose that $\mathcal{T}_1$ and $\mathcal{T}_2$ are two theories, where $\mathcal{T}_1 \subseteq \mathcal{T}_2$, and the vocabulary of $\mathcal{T}_2$ may contain function or predicate symbols not in $\mathcal{T}_1$. We say $\mathcal{T}_2$ is a* conservative extension of $\mathcal{T}_1$ *if for every formula A in the vocabulary of $\mathcal{T}_1$, if $\mathcal{T}_2 \vdash A$ then $\mathcal{T}_1 \vdash A$.*

Theorem 3.30 (Extension by Definition Theorem). *If $\mathcal{T}_2$ results from $\mathcal{T}_1$ by expanding the vocabulary of $\mathcal{T}_1$ to include definable symbols, and by adding the defining axioms for these symbols, then $\mathcal{T}_2$ is a conservative extension of $\mathcal{T}_1$.*

Proof. We give a simple model-theoretic argument. Suppose that A is a formula in the vocabulary of $\mathcal{T}_1$ and suppose that $\mathcal{T}_2 \vdash A$. Let $\mathcal{M}_1$ be a model of $\mathcal{T}_1$. We expand $\mathcal{M}_1$ to a model $\mathcal{M}_2$ of $\mathcal{T}_2$ by interpreting each new predicate and function symbol so that its defining axiom (23) or (25) is satisfied. Notice that this interpretation is uniquely determined by the defining axiom, and in the case of a function symbol the provability condition (24) is needed (both existence and uniqueness of y) in order to ensure that both directions of the equivalence (25) hold.

Since $\mathcal{M}_2$ is a model of $\mathcal{T}_2$, it follows that $\mathcal{M}_2 \models A$, and hence $\mathcal{M}_1 \models A$. Since $\mathcal{M}_1$ is an arbitrary model of $\mathcal{T}_1$, it follows that $\mathcal{T}_1 \vdash A$. $\dashv$

Corollary 3.31. *Let $\mathcal{T}$ be a theory and $\mathcal{T}_0 = \mathcal{T} \subset \mathcal{T}_1 \subset \dots$ be a sequence of extensions of $\mathcal{T}$ where each $\mathcal{T}_{n+1}$ is obtained by adding to $\mathcal{T}_n$ a definable symbol (in the vocabulary of $\mathcal{T}_n$) and its defining axiom. Let $\mathcal{T}_\infty = \bigcup_{n \ge 0} \mathcal{T}_n$. Then $\mathcal{T}_\infty$ is a conservative extension of $\mathcal{T}$.*

Exercise 3.32. *Prove the corollary using the Extension by Definition Theorem and the Compactness Theorem.*

As an application of the Extension by Definition Theorem, we can conservatively extend **PA** to include symbols for all the *arithmetical* predicates (i.e., predicates definable by $\mathcal{L}_A$-formulas). In fact, the extension of **PA** remains conservative even if we allow induction on *formulas over the expanded vocabulary.*

Similarly we can also obtain a conservative extension of $\mathbf{I}\Delta_0$ by adding to it predicate symbols and their defining axioms for all arithmetical predicates. However such a conservative extension of $\mathbf{I}\Delta_0$ no longer proves the induction axiom scheme on bounded formulas over the expanded vocabulary. It does so if we only add Δ_0-definable symbols, and in fact we may add both Δ_0-definable predicate and function symbols. To show this, we start with the following important application of Parikh's Theorem.

Theorem 3.33 (Bounded Definability Theorem). *Let $\mathcal{T}$ be a polynomial-bounded theory. A function $f(\vec{x})$ (not in $\mathcal{T}$) is Σ_1-definable in $\mathcal{T}$ iff it has a defining axiom*

$$y = f(\vec{x}) \leftrightarrow \varphi(\vec{x}, y)$$

where φ is a bounded formula with all free variables indicated, and there is an $\mathcal{L}_A$-term $t = t(\vec{x})$ such that $\mathcal{T}$ proves $\forall\vec{x}\exists!y \le t\varphi(\vec{x}, y)$.

Proof. The IF direction is immediate from Definition 3.27. The ONLY IF direction follows from the discussion after Parikh's Theorem (3.20). ⊣

Corollary 3.34. *If $\mathcal{T}$ is a polynomial-bounded theory, then a function f is Σ_1-definable in $\mathcal{T}$ iff f is Δ_0-definable in $\mathcal{T}$.*

From the above theorem we see that the function 2^x is not Σ_1-definable in any polynomial-bounded theory, even though we shall show in Section 3C.3 that the *relation* $(y = 2^x)$ is Δ_0-definable in $\mathbf{I}\Delta_0$. Since the function 2^x is Σ_1-definable in **PA**, it follows that $\mathbf{I}\Delta_0 \subsetneq \mathbf{PA}$.

Lemma 3.35 (Conservative Extension Lemma). *Suppose that $\mathcal{T}$ is a polynomial-bounded theory and $\mathcal{T}^+$ is the conservative extension of $\mathcal{T}$ obtained by adding to $\mathcal{T}$ a Δ_0-definable predicate or a Σ_1-definable function symbol and its defining axiom. Then $\mathcal{T}^+$ is a polynomial-bounded theory and every bounded formula φ^+ in the vocabulary of $\mathcal{T}^+$ can be translated into a bounded formula φ in the vocabulary of $\mathcal{T}$ such that*

$$\mathcal{T}^+ \vdash \varphi^+ \leftrightarrow \varphi$$

The following corollary follows immediately from the lemma.

Corollary 3.36. *Let $\mathcal{T}$ and $\mathcal{T}^+$ be as in the Conservative Extension Lemma. Let $\mathcal{L}$ and $\mathcal{L}^+$ denote the vocabulary of $\mathcal{T}$ and $\mathcal{T}^+$, respectively. Assume further that $\mathcal{T}$ proves the $\Delta_0(\mathcal{L})$-**IND** axiom scheme. Then $\mathcal{T}^+$ proves the $\Delta_0(\mathcal{L}^+)$-**IND** axiom scheme.*

Proof of the Conservative Extension Lemma. First, suppose that $\mathcal{T}^+$ is obtained from $\mathcal{T}$ by adding to it a Δ_0-definable predicate symbol P and its defining axiom (23). That $\mathcal{T}^+$ is polynomial-bounded is immediate from Definition 3.19. Now each bounded formula in the vocabulary of $\mathcal{T}^+$ can be translated to a bounded formula in the vocabulary of $\mathcal{T}$ simply by replacing each occurrence of a formula of the form $P(\vec{t})$ by $\varphi(\vec{t})$ (see the Formula Replacement Theorem, 2.34). Note that the defining axiom (23) becomes the valid formula $\varphi(\vec{x}) \leftrightarrow \varphi(\vec{x})$.

Next suppose that $\mathcal{T}^+$ is obtained from $\mathcal{T}$ by adding to it a Σ_1-definable function symbol f and its defining axiom (25). That $\mathcal{T}^+$ is polynomial-bounded follows from Theorem 3.33.

Start translating φ^+ by replacing every bounded quantifier $\forall x \le u\psi$ by $\forall x \le u'(x \le u \supset \psi)$, where u' is obtained from u by replacing every occurrence of every function symbol other than $+, \cdot$ by its bounding term in $\mathcal{L}_A$. Similarly replace $\exists x \le u\psi$ by $\exists x \le u'(x \le u \wedge \psi)$.

Now we may suppose by Theorem 3.33 that f has a bounded defining axiom

$$y = f(\vec{x}) \leftrightarrow \varphi_1(\vec{x}, y)$$

and $f(\vec{x})$ has an $\mathcal{L}_A$ bounding term $t(\vec{x})$. Repeatedly remove occurrences of f in an atomic formula $\theta(s(f(\vec{u})))$ by replacing this with

$$\exists y \le t(\vec{u}),\ \varphi_1(\vec{u}, y) \wedge \theta(s(y)) \qquad \dashv$$

Now we summarize the previous results.

Theorem 3.37 (Conservative Extension Theorem). *Let $\mathcal{T}_0$ be a polynomial-bounded theory over a vocabulary $\mathcal{L}_0$ which proves the $\Delta_0(\mathcal{L}_0)$-$\mathbf{IND}$ axioms. Let $\mathcal{T}_0 \subset \mathcal{T}_1 \subset \mathcal{T}_2 \subset \ldots$ be a sequence of extensions of $\mathcal{T}_0$ where each $\mathcal{T}_{i+1}$ is obtained from $\mathcal{T}_i$ by adding a Σ_1-definable function symbol f_{i+1} (or a Δ_0-definable predicate symbol P_{i+1}) and its defining axiom. Let*

$$\mathcal{T} = \bigcup_{i \ge 0} \mathcal{T}_i$$

Then $\mathcal{T}$ is a polynomial-bounded theory and is a conservative extension of $\mathcal{T}_0$. Furthermore, if $\mathcal{L}$ is the language of $\mathcal{T}$, then $\mathcal{T}$ proves the equivalence of each $\Delta_0(\mathcal{L})$ formula with some $\Delta_0(\mathcal{L}_0)$ formula, and $\mathcal{T} \vdash \Delta_0(\mathcal{L})$-$\mathbf{IND}$.

Proof. First, we prove by induction on i that

1) $\mathcal{T}_i$ is a polynomial-bounded theory;
2) $\mathcal{T}_i$ is a conservative extension of $\mathcal{T}_0$; and
3) $\mathcal{T}_i$ proves that each $\Delta_0(\mathcal{L}_i)$ formula is equivalent to some $\Delta_0(\mathcal{L}_0)$ formula, where $\mathcal{L}_i$ is the vocabulary of $\mathcal{T}_i$.

The induction step follows from the Conservative Extension Lemma.

It follows from the induction arguments above that $\mathcal{T}$ is a polynomial-bounded theory, and that $\mathcal{T}$ proves the equivalence of each $\Delta_0(\mathcal{L})$ formula with some $\Delta_0(\mathcal{L}_0)$ formula, and $\mathcal{T} \vdash \Delta_0(\mathcal{L})$-**IND**. It follows from Corollary 3.31 that $\mathcal{T}$ is a conservative extension of $\mathcal{T}_0$. ⊣

3C.2. $\overline{\mathbf{I}\Delta}_0$: A Universal Conservative Extension of $\mathbf{I}\Delta_0$. (This subsection is not needed for the remainder of this chapter, but it is needed for later chapters.)

We begin by introducing terminology that allows us to restate the Herbrand Theorem (see Section 2E).

A *universal formula* is a formula in prenex form (Definition 2.74) in which all quantifiers are universal. A*universal theory* is a theory which can be axiomatized by universal formulas. Note that by definition (3.1), a universal theory can be equivalently axiomatized by a set of quantifier-free formulas, or by a set of $\forall$ sentences (Definition 2.64). We can now restate Form 2 of the Herbrand Theorem 2.68 as follows.

THEOREM 3.38 (Herbrand Theorem, Form 2). *Let $\mathcal{T}$ be a universal theory, and let $\varphi(x_1, \ldots, x_m, y)$ be a quantifier-free formula with all free variables indicated such that*

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_m \exists y \varphi(\vec{x}, y). \tag{26}$$

Then there exist finitely many terms $t_1(\vec{x}), \ldots, t_n(\vec{x})$ such that

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_m \left(\varphi(\vec{x}, t_1(\vec{x})) \vee \cdots \vee \varphi(\vec{x}, t_n(\vec{x}))\right)$$

Note that the theorem easily extends to the case where

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_m \exists y_1 \ldots \exists y_k \varphi(\vec{x}, \vec{y}).$$

instead of (26), where $\varphi(\vec{x}, \vec{y})$ is a quantifier-free formula.

PROOF. As we have remarked earlier, $\mathcal{T}$ can be axiomatized by a set Γ of $\forall$ sentences. From (26) it follows that

$$\Gamma \cup \{\exists x_1 \ldots \exists x_m \forall y \neg\varphi(\vec{x}, y)\} \tag{27}$$

is unsatisfiable. Let $c_1, \ldots, c_m$ be new constant symbols. Then it is easy to check that (27) is unsatisfiable if and only if

$$\Gamma \cup \{\forall y \neg\varphi(\vec{c}, y)\}$$

is unsatisfiable. (We will need only the ONLY IF ($\Longrightarrow$) direction.)

Now by Form 1 (Theorem 2.67), there are terms $t_1(\vec{c}), \ldots, t_n(\vec{c})$ such that

$$\Gamma \cup \{\neg\varphi(\vec{c}, t_1(\vec{c})), \ldots, \neg\varphi(\vec{c}, t_n(\vec{c}))\}$$

is unsatisfiable. (We can assume that $n \geq 1$, since $n = 0$ implies that Γ is itself unsatisfiable, and in that case the theorem is vacuously true.) Then it follows easily that

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_m \left(\varphi(\vec{x}, t_1(\vec{x})) \vee \cdots \vee \varphi(\vec{x}, t_n(\vec{x}))\right)$$ ⊣

As stated, the Herbrand Theorem applies only to universal theories. However every theory has a universal conservative extension, which can be obtained by introducing "Skolem functions". The idea is that these functions explicitly *witness* the existence of existentially quantified variables. Thus we can replace each axiom (which contains $\exists$) of a theory $\mathcal{T}$ by a universal axiom.

LEMMA 3.39. *Suppose that $\psi(\vec{x}) \equiv \exists y \varphi(\vec{x}, y)$ is an axiom of a theory $\mathcal{T}$. Let f be a new function symbol, and let $\mathcal{T}'$ be the theory over the extended vocabulary with the same set of axioms as $\mathcal{T}$ except that $\psi(\vec{x})$ is replaced by*

$$\varphi(\vec{x}, f(\vec{x}))$$

Then $\mathcal{T}'$ is a conservative extension of $\mathcal{T}$.

The new function f is called a *Skolem function.*

EXERCISE 3.40. *Prove the above lemma by a simple model-theoretic argument showing that every model of $\mathcal{T}$ can be expanded to a model of $\mathcal{T}'$. It may be helpful to assume that the language of $\mathcal{T}$ is countable, so by the Löwenheim/Skolem Theorem (Theorem 2.60) we may restrict attention to countable models.*

By the lemma, for each axiom of $\mathcal{T}$ we can successively eliminate the existential quantifiers, starting from the outermost quantifier, using the Skolem functions. It follows that every theory has a universal conservative extension. For example, we can obtain a universal conservative extension of $\mathbf{I\Delta_0}$ by introducing Skolem functions for every instance of the Δ_0-**IND** axiom scheme. Let $\varphi(z)$ be a Δ_0 formula (possibly with other free variables $\vec{x}$). Then the induction scheme for $\varphi(z)$ can be written as

$$\forall \vec{x} \forall z \big(\varphi(z) \vee \neg\varphi(0) \vee \exists y(\varphi(y) \wedge \neg\varphi(y+1))\big)$$

Consider the simple case where φ is an open formula. The single Skolem function (as a function of $\vec{x}, z$) for the above formula is required to "witness" the existence of y (in case such a y exists).

Although the Skolem functions witness the existence of existentially quantified variables, it is not specified which values they take (and in general there may be many different values). Here we can construct a universal conservative extension of $\mathbf{I\Delta_0}$ by explicitly taking the smallest values of the witnesses if they exist. Using the least number principle (Definition 3.13), these functions are indeed definable in $\mathbf{I\Delta_0}$.

Let $\varphi(z)$ be an open formula (possibly with other free variables), and t a term. Let $\vec{x}$ be the list of all variables of t and other free variables of $\varphi(z)$ (thus $\vec{x}$ may contain z if t does). Let $f_{\varphi(z),t}(\vec{x})$ be the least $y < t$ such that $\varphi(y)$ holds, or t if no such y exists. Then $f_{\varphi(z),t}$ is total and can be defined as follows (we assume that y, v do not appear in $\vec{x}$):

$$(28) \quad y = f_{\varphi(z),t}(\vec{x}) \leftrightarrow \big(y \leq t \;\wedge\; (y < t \supset \varphi(y)) \;\wedge\; \forall v < y \neg\varphi(v)\big)$$

Note that (28) contains an implicit existential quantifier $\exists v$ (consider the direction $\leftarrow$). Our universal theory will contain the following equivalent axiom instead:

$$f(\vec{x}) \leq t \ \wedge \ \big(f(\vec{x}) < t \supset \varphi(f(\vec{x}))\big) \ \wedge \ \big(v < f(\vec{x}) \supset \neg\varphi(v)\big) \tag{29}$$

(here $f = f_{\varphi(z),t}$).

Although the predecessor function $pd(x)$ can be defined by a formula of the form (29), we will use the following two recursive defining axioms instead.

D1$'$. $pd(0) = 0$ **D**1$''$. $x \neq 0 \supset pd(x) + 1 = x$

Note that **D**1$''$ implies **D**1 (see Example 3.9), and **D**1$'$ is needed to define $pd(0)$.

We are now ready to define the language $\mathcal{L}_{\Delta_0}$ of the universal theory $\overline{\mathbf{I}\Delta}_0$. This language has a function symbol for every Δ_0-definable function in $\mathbf{I}\Delta_0$.

DEFINITION 3.41 ($\mathcal{L}_{\Delta_0}$). *Let $\mathcal{L}_{\Delta_0}$ be the smallest set that satisfies*

1) *$\mathcal{L}_{\Delta_0}$ includes $\mathcal{L}_A \cup \{pd\}$;*
2) *For each open $\mathcal{L}_{\Delta_0}$-formula $\varphi(z)$ and $\mathcal{L}_A$-term t there is a function $f_{\varphi(z),t}$ in $\mathcal{L}_{\Delta_0}$.*

Note that $\mathcal{L}_{\Delta_0}$ can be alternatively defined as follows. Let

$$\mathcal{L}_0 = \mathcal{L}_A \cup \{pd\}$$

for $n \geq 0$: $\mathcal{L}_{n+1} = \mathcal{L}_n \cup \{f_{\varphi(z),t} : \varphi(z)$ is an open $\mathcal{L}_n$-formula, t is an $\mathcal{L}_A$-term$\}$

Then

$$\mathcal{L}_{\Delta_0} = \bigcup_{n\geq 0} \mathcal{L}_n$$

Our universal theory $\overline{\mathbf{I}\Delta}_0$ requires two more axioms in the style of 1-**BASIC**.

B8$'$. $0 \leq x$
B8$''$. $x < x + 1$

DEFINITION 3.42 ($\overline{\mathbf{I}\Delta}_0$). *Let $\overline{\mathbf{I}\Delta}_0$ be the theory over $\mathcal{L}_{\Delta_0}$ with the following set of axioms:* **B**1, ... , **B**8, **B**8$'$, **B**8$''$, **D**1$'$, **D**1$''$ *and* (29) *for each function $f_{\varphi(z),t}$ of $\mathcal{L}_{\Delta_0}$.*

Thus $\overline{\mathbf{I}\Delta}_0$ is a universal theory. Note that there is no induction scheme among its axioms. Nevertheless we show below that $\overline{\mathbf{I}\Delta}_0$ proves the Δ_0-**IND** axiom scheme, and hence $\overline{\mathbf{I}\Delta}_0$ extends $\mathbf{I}\Delta_0$. From this it is easy to verify that $\overline{\mathbf{I}\Delta}_0$ is a polynomial-bounded theory.

THEOREM 3.43. *$\overline{\mathbf{I}\Delta}_0$ is a conservative extension of $\mathbf{I}\Delta_0$.*

To show that $\overline{\mathbf{I\Delta}}_0$ extends $\mathbf{I\Delta}_0$ we show that it proves the Δ_0-**IND** axiom scheme. Note that if the functions of $\mathcal{L}_{\Delta_0}$ receive their intended meaning, then every bounded $\mathcal{L}_A$-formula is equivalent to an open $\mathcal{L}_{\Delta_0}$-formula. Therefore, roughly speaking, the Δ_0-**MIN** (and thus Δ_0-**IND**) axiom scheme is satisfied by considering the appropriate functions of $\mathcal{L}_{\Delta_0}$.

Lemma 3.44. *For each $\Delta_0(\mathcal{L}_A)$ formula φ, there is an open $\mathcal{L}_{\Delta_0}$-formula φ' such that $\overline{\mathbf{I\Delta}}_0 \vdash \varphi \leftrightarrow \varphi'$.*

Proof. We use structural induction on φ. The only interesting cases are for bounded quantifiers. It suffices to consider the case when φ is $\exists y \le t\psi(y)$. Then take φ' to be $\psi'(f_{\psi',t}(\vec{x}))$. It is easy to check that $\overline{\mathbf{I\Delta}}_0 \vdash \varphi \leftrightarrow \varphi'$ using (29). No properties of $\le$ and $<$ are needed for this implication except the definition $y < f(\vec{x})$ stands for $(y \le f(\vec{x}) \wedge y \ne f(\vec{x}))$. $\dashv$

Proof of Theorem 3.43. First we show that $\overline{\mathbf{I\Delta}}_0$ is an extension of $\mathbf{I\Delta}_0$, i.e., Δ_0-**IND** is provable in $\overline{\mathbf{I\Delta}}_0$.

By the above lemma, it suffices to show that $\overline{\mathbf{I\Delta}}_0$ proves the Induction axiom scheme for open $\mathcal{L}_{\Delta_0}$-formulas. Let $\varphi(\vec{x}, z)$ be any open $\mathcal{L}_{\Delta_0}$-formula. We need to show that (omitting $\vec{x}$)

$$\overline{\mathbf{I\Delta}}_0 \vdash (\varphi(0) \wedge \neg\varphi(z)) \;\supset\; \exists y(\varphi(y) \wedge \neg\varphi(y+1))$$

Assuming $(\varphi(0) \wedge \neg\varphi(z))$, we show in $\overline{\mathbf{I\Delta}}_0$ that $(\varphi(y) \wedge \neg\varphi(y+1))$ holds for $y = pd(f_{\neg\varphi,z}(\vec{x}, z))$, using (29). We need to be careful when arguing about $\le$, because the properties **O**1–**O**9 and **D**1–**D**10 which we have been using for reasoning in $\mathbf{I\Delta}_0$ require induction to prove.

First we rewrite (29) for the case f is $f_{\neg\varphi,z}$.

$$(30)\quad f(\vec{x}, z) \le z \wedge \big(f(\vec{x}, z) < z \supset \neg\varphi(f(\vec{x}, z))\big) \wedge \big(v < f(\vec{x}, z) \supset \varphi(v)\big)$$

Now $0 < z$ by $\mathbf{B}8'$ and our assumptions $\varphi(0)$ and $\neg\varphi(z)$, so $f(\vec{x}, z) \ne 0$ by (30). Hence $y + 1 = pd(f(\vec{x}, z)) + 1 = f(\vec{x}, z)$ by $\mathbf{D}1''$. Therefore $\neg\varphi(y+1)$ by (30) and the assumption $\neg\varphi(z)$.

To establish $\varphi(y)$ it suffices by (30) to show $y < f(\vec{x}, z)$. This holds because $f(\vec{x}, z) = y + 1$ as shown above, and $y < y + 1$ by $\mathbf{B}8''$.

This completes the proof that $\overline{\mathbf{I\Delta}}_0$ extends $\mathbf{I\Delta}_0$. Next, we show that $\overline{\mathbf{I\Delta}}_0$ is conservative over $\mathbf{I\Delta}_0$. Let $f_1 = pd, f_2, f_3, \dots$ be an enumeration of $\mathcal{L}_{\Delta_0} \setminus \mathcal{L}_A$ such that for $n \ge 1$, f_{n+1} is defined using some $\mathcal{L}_A$-term t and $(\mathcal{L}_A \cup \{f_1, \dots, f_n\})$-formula φ as in (29).

For $n \ge 0$ let $\mathcal{L}_n$ denote $\mathcal{L}_A \cup \{f_1, \dots, f_n\}$. Let $\mathcal{T}_0 = \mathbf{I\Delta}_0$, and for $n \ge 0$ let $\mathcal{T}_{n+1}$ be the theory over $\mathcal{L}_{n+1}$ which is obtained from $\mathcal{T}_n$ by adding the defining axiom for f_{n+1} (in particular, $\mathcal{T}_1$ is axiomatized by $\mathbf{I\Delta}_0$ and $\mathbf{D}1'$, $\mathbf{D}1''$). Then

$$\mathcal{T}_0 = \mathbf{I\Delta}_0 \subset \mathcal{T}_1 \subset \mathcal{T}_2 \subset \dots \qquad \text{and} \qquad \overline{\mathbf{I\Delta}}_0 = \bigcup_{n \ge 0} \mathcal{T}_n.$$

By Corollary 3.31, it suffices to show that for each $n \geq 0$, f_{n+1} is definable in $\mathcal{T}_n$. In fact, we prove the following by induction on $n \geq 0$:

1) $\mathcal{T}_n$ proves the $\Delta_0(\mathcal{L}_n)$-**IND** axiom scheme;
2) f_{n+1} is $\Delta_0(\mathcal{L}_n)$-definable in $\mathcal{T}_n$.

Consider the induction step. Suppose that the hypothesis is true for n ($n \geq 0$). We prove it for $n+1$. By the induction hypothesis, $\mathcal{T}_n$ proves the $\Delta_0(\mathcal{L}_n)$-**IND** axiom scheme and $\Delta_0(\mathcal{L}_n)$-defines f_{n+1}. Therefore by Corollary 3.36, $\mathcal{T}_{n+1}$ proves the $\Delta_0(\mathcal{L}_{n+1})$-**IND** axiom scheme. Consequently, $\mathcal{T}_{n+1}$ also proves the $\Delta_0(\mathcal{L}_{n+1})$-**MIN** axiom scheme. The defining equation for f_{n+2} has the form (29), and hence $\mathcal{T}_{n+1}$ proves (28) where f is f_{n+2}. Thus (28) is a defining axiom which shows that f_{n+2} is $\Delta_0(\mathcal{L}_{n+1})$-definable in $\mathcal{T}_{n+1}$. Here we use the $\Delta_0(\mathcal{L}_{n+1})$-**MIN** axiom scheme to prove $\exists y$ in (24). $\dashv$

3C.2.1. *An alternative proof of Parikh's Theorem for* $\mathbf{I}\Delta_0$. Now we will present an alternative proof of Parikh's Theorem for $\mathbf{I}\Delta_0$ from Herbrand Theorem applied to $\overline{\mathbf{I}\Delta}_0$, using the fact that $\overline{\mathbf{I}\Delta}_0$ is a conservative extension of $\mathbf{I}\Delta_0$.

Note that in proving that $\overline{\mathbf{I}\Delta}_0$ is conservative over $\mathbf{I}\Delta_0$ (see the proof of Theorem 3.43), in the induction step we have used Corollary 3.36 (the case of adding Σ_1-definable function) to show that $\mathcal{T}_n$ proves the $\Delta_0(\mathcal{L}_n)$-**IND** axiom scheme. The proof of Corollary 3.36 (and of the Conservative Extension Lemma) in turns relies on the Bounded Definability Theorem 3.33, which is proved using Parikh's Theorem. However, for $\overline{\mathbf{I}\Delta}_0$, the function f_{n+1} in the induction step in the proof of Theorem 3.43 is already Δ_0-definable in $\mathcal{T}_n$ and comes with a bounding term t. Therefore we have actually used only a simple case of Corollary 3.36 (i.e., adding Δ_0-definable functions with bounding terms). Thus in fact Parikh's Theorem is not necessary in proving Theorem 3.43.

PROOF OF PARIKH'S THEOREM. Suppose that $\forall\vec{x}\exists y\varphi(\vec{x},y)$ is a theorem of $\mathbf{I}\Delta_0$, where φ is a bounded formula. We will show that there is an $\mathcal{L}_A$-term s such that

$$\mathbf{I}\Delta_0 \vdash \forall\vec{x}\exists y \leq s\varphi(\vec{x},y)$$

By Lemma 3.44, there is an open $\mathcal{L}_{\Delta_0}$-formula $\varphi'(\vec{x},y)$ such that

$$\overline{\mathbf{I}\Delta}_0 \vdash \forall\vec{x}\forall y(\varphi(\vec{x},y) \leftrightarrow \varphi'(\vec{x},y))$$

Then since $\overline{\mathbf{I}\Delta}_0$ extends $\mathbf{I}\Delta_0$, it follows that

$$\overline{\mathbf{I}\Delta}_0 \vdash \forall\vec{x}\exists y\varphi'(\vec{x},y)$$

Now since $\overline{\mathbf{I}\Delta}_0$ is a universal theory, by Form 2 of the Herbrand Theorem 3.38 there are $\mathcal{L}_{\Delta_0}$-terms $t_1,\ldots,t_n$ such that

$$\overline{\mathbf{I}\Delta}_0 \vdash \forall\vec{x}\big(\varphi'(\vec{x},t_1(\vec{x})) \vee \cdots \vee \varphi'(\vec{x},t_n(\vec{x}))\big) \tag{31}$$

Also since $\overline{\mathbf{I\Delta}}_0$ is a polynomial-bounded theory, there is an $\mathcal{L}_A$-term s such that

$$\overline{\mathbf{I\Delta}}_0 \vdash t_i(\vec{x}) < s(\vec{x}) \qquad \text{for all } i,\ 1 \le i \le n$$

Consequently,

$$\overline{\mathbf{I\Delta}}_0 \vdash \forall \vec{x} \exists y < s \varphi'(\vec{x}, y)$$

Hence

$$\overline{\mathbf{I\Delta}}_0 \vdash \forall \vec{x} \exists y < s \varphi(\vec{x}, y)$$

By the fact that $\overline{\mathbf{I\Delta}}_0$ is conservative over $\mathbf{I\Delta}_0$ we have

$$\mathbf{I\Delta}_0 \vdash \forall \vec{x} \exists y < s \varphi(\vec{x}, y) \qquad \dashv$$

Note that we have proved more than a bound on the existential quantifier $\exists y$. In fact, (31) allows us to explicitly define a Skolem function $y = f(\vec{x})$, using definition by cases. This idea will serve as a method for proving witnessing theorems in future chapters.

3C.3. Defining $y = 2^x$ and $BIT(i, x)$ in $\mathbf{I\Delta}_0$. In this subsection we show that the relation $BIT(i, x)$ is Δ_0-definable in $\mathbf{I\Delta}_0$, where $BIT(i, x)$ holds iff the i-th bit in the binary notation for x is 1. This is useful particularly in Section 3D where we show that $\mathbf{I\Delta}_0$ characterizes the Linear Time Hierarchy.

In order to define BIT we will show that the relation $y = 2^x$ is Δ_0-definable in $\mathbf{I\Delta}_0$. Note that on the other hand, by Parikh's Theorem 3.20, the *function* $f(x) = 2^x$ is not Σ_1-definable in $\mathbf{I\Delta}_0$, because it grows faster than any polynomial.

Our method is to introduce a sequence of new function and predicate symbols, and show that each can be Δ_0-defined in $\mathbf{I\Delta}_0$ extended by the previous symbols. These new symbols together with their defining axioms determine a sequence of conservative extensions of $\mathbf{I\Delta}_0$, and according to the Conservative Extension Theorem 3.37, bounded formulas using the new symbols are provably equivalent to bounded formulas in the vocabulary $\mathcal{L}_A$ of $\mathbf{I\Delta}_0$, and hence the induction scheme is available on bounded formulas with the new symbols. Finally the bounded formula $\varphi_{exp}(x, y)$ given in (34) defines $(y = 2^x)$, and the bounded formula $BIT(i, x)$ given in (35) defines the BIT predicate. These formulas are provably equivalent to bounded formulas in $\mathbf{I\Delta}_0$, and $\mathbf{I\Delta}_0$ proves the properties of their translations, such as those in Exercise 3.53.

We start by Δ_0-defining the following functions in $\mathbf{I\Delta}_0$: $x \dot{-} y$, $\lfloor x/y \rfloor$, $x \bmod y$ and $\lfloor \sqrt{x} \rfloor$. We will show in detail that $x \dot{-} y$ is Δ_0-definable in $\mathbf{I\Delta}_0$. A detailed proof for other functions is left as an exercise. It might be helpful to revisit the basic properties $\mathbf{O}1, \ldots, \mathbf{O}10$, $\mathbf{D}1, \ldots, \mathbf{D}10$ of $\mathbf{I\Delta}_0$ in Examples 3.8, 3.9.

1) **Limited subtraction**: The function $x \dot{-} y = max\{0, x - y\}$ can be defined by

$$z = x \dot{-} y \leftrightarrow \big((y + z = x) \vee (x \leq y \wedge z = 0)\big)$$

In order to show that $\mathbf{I}\Delta_0$ can Δ_0-define this function we must show that

$$\mathbf{I}\Delta_0 \vdash \forall x \forall y \exists! z \varphi(x, y, z)$$

where φ is the RHS of the above equivalence (see Definition 3.27(b)).

For the existence of z, by **D**2 we know that there is some z' such that

$$x + z' = y \vee y + z' = x.$$

If $y + z' = x$ then simply take $z = z'$. Otherwise $x + z' = y$, then by **B**8, $x \leq x + z'$, hence $x \leq y$, and thus we can take $z = 0$.

For the uniqueness of z, first suppose that $x \leq y$. Then we have to show that $y + z = x \supset z = 0$. Assume $y + z = x$. By **B**8, $y \leq y + z$, hence $y \leq x$. Therefore $x = y$ by **B**7. Now from $x + 0 = x$ (**B**3) and $x + z = x$ we have $z = 0$, by **O**2 (Commutativity of $+$) and **O**6 (Cancellation law for $+$).

Next, suppose that $\neg(x \leq y)$. Then $y + z = x$, and by **O**2 and **O**6, $y + z = x \wedge y + z' = x \supset z = z'$.

2) **Division**: The function $x \text{ div } y = \lfloor x/y \rfloor$ can be defined by

$$z = \lfloor x/y \rfloor \leftrightarrow \big((y \cdot z \leq x \wedge x < y \cdot (z + 1)) \vee (y = 0 \wedge z = 0)\big)$$

The existence of z is proved by induction on x. The uniqueness of z follows from transitivity of $\leq$ (**D**4), Total Order (**D**5), and **O**5, **D**7.

3) **Remainder**: The function $x \bmod y$ can be defined by

$$x \bmod y = x \dot{-} (y \cdot \lfloor x/y \rfloor)$$

Since $x \bmod y$ is a composition of Σ_1-definable functions, it is Σ_1-definable by Exercise 3.28. Hence it is Δ_0-definable by Corollary 3.34.

4) **Square root**:

$$y = \lfloor \sqrt{x} \rfloor \leftrightarrow \big(y \cdot y \leq x \wedge x < (y + 1)(y + 1)\big)$$

The existence of y follows from the least number principle. The uniqueness of y follows from Transitivity of $\leq$ (**D**4), Total Order (**D**5), and **O**5, **D**7.

EXERCISE 3.45. *Show carefully that the functions $\lfloor x/y \rfloor$ and $\lfloor \sqrt{x} \rfloor$ are Δ_0-definable in $\mathbf{I}\Delta_0$.*

Next we define the following relations $x|y$, *Pow2*(x), *Pow4*(x) and *LenBit*(y, x):

5) **Divisibility**: This relation is defined by

$$x|y \leftrightarrow \exists z \leq y(x \cdot z = y)$$

6) **Powers of 2 and 4**:
x is a power of 2:

$$\mathit{Pow2}(x) \leftrightarrow \big(x \neq 0 \wedge \forall y \leq x((1 < y \wedge y|x) \supset 2|y)\big)$$

x is a power of 4: $\mathit{Pow4}(x) \leftrightarrow (\mathit{Pow2}(x) \wedge x \bmod 3 = 1)$.

7) **LenBit**: We want the relation $\mathit{LenBit}(2^i, x)$ to hold iff the i-th bit in the binary expansion of x is 1, where the least significant bit is bit 0. Although we cannot yet define $y = 2^i$, we can define

$$\mathit{LenBit}(y, x) \leftrightarrow (\lfloor x/y \rfloor \bmod 2 = 1)$$

Note that we intend to use $\mathit{LenBit}(y, x)$ only when y is a power of 2, but it is defined for all values of y.

Notation. $(\forall 2^i)$ stands for "for all powers of 2", i.e.,

$(\forall 2^i)\ A(2^i)$	stands for	$\forall x\ (\mathit{Pow2}(x) \supset A(x))$
$(\forall 2^i \leq t)\ A(2^i)$	stands for	$\forall x\ ((\mathit{Pow2}(x) \wedge x \leq t) \supset A(x))$

Same for $(\exists 2^i)$ and $(\exists 2^i \leq t)$.

Exercise 3.46. *Show that the following are theorems of* $\mathbf{I}\Delta_0$*:*

(a) $\mathit{Pow2}(x) \leftrightarrow \mathit{Pow2}(2x)$.
(b) $(\forall 2^i)(\forall 2^j)(2^i < 2^j \supset 2^i|2^j)$. *(Hint: using strong induction* (12)*.)*
(c) $(\forall 2^i)(\forall 2^j \leq 2^i)\ \mathit{Pow2}(\lfloor 2^i/2^j \rfloor)$.
(d) $(\forall 2^i)(\forall 2^j)(2^i < 2^j \supset 2 \cdot 2^i \leq 2^j)$.
(e) $(\forall 2^i)(\forall 2^j)\ \mathit{Pow2}(2^i \cdot 2^j)$.
(f) $(\forall 2^i)(\exists 2^j \leq 2^i)\ ((2^j)^2 = 2^i \vee 2(2^j)^2 = 2^i))$.

We also need the following function:

8) **Greatest power of 2 less than or equal to x:**

$$y = gp(x) \leftrightarrow ((x = 0 \wedge y = 0) \vee (\mathit{Pow2}(y) \wedge y \leq x \wedge (\forall 2^i \leq x)\ 2^i \leq y))$$

Exercise 3.47. *Show that* $\mathbf{I}\Delta_0$ *can* Δ_0*-define* $gp(x)$*. (Hint: Use induction on* x*.)*

Exercise 3.48. *Prove the following in* $\mathbf{I}\Delta_0$*:*

(a) $x > 0 \supset (gp(x) \leq x < 2gp(x))$.
(b) $x > 0 \supset \mathit{LenBit}(gp(x), x)$.
(c) $y = x \dot{-} gp(x) \supset (\forall 2^i \leq y)\ (\mathit{LenBit}(2^i, y) \leftrightarrow \mathit{LenBit}(2^i, x))$.

It is a theorem of $\mathbf{I}\Delta_0$ that the binary representation of a number uniquely determines the number. This theorem can be proved in $\mathbf{I}\Delta_0$ by using strong induction (12) and part (c) of the above exercise. Details are left as an exercise.

THEOREM 3.49.

$$\mathbf{I\Delta_0} \vdash \forall y \forall x < y (\exists 2^i \leq y)(LenBit(2^i, y) \wedge \neg LenBit(2^i, x))$$

EXERCISE 3.50. *Prove the above theorem.*

3C.3.1. *Defining the Relation $y = 2^x$.* This is much more difficult to Δ_0-define than any of the previous relations and functions. A first attempt to define $y = 2^x$ might be to assert the existence of a number s coding the sequence $\langle 2^0, 2^1, ..., 2^x \rangle$. The main difficulty in this attempt is that the number of bits in s is $\Omega(|y|^2)$ (where $|y|$ is the number of bits in y), and so s cannot be bounded by any $\mathbf{I\Delta_0}$ term in x and y.

We get around this by coding a much shorter sequence, of length $|x|$ instead of length x, of numbers of the form 2^z. Suppose that $x > 0$, and $(x_{k-1} \dots x_0)_2$ is the binary representation of x (where $x_{k-1} = 1$), i.e.,

$$x = \sum_{i=0}^{k-1} x_i 2^i \qquad \text{(and } x_{k-1} = 1\text{)}$$

We start by coding the sequence $\langle a_1, a_2, ..., a_k \rangle$, where a_i consists of the first i high-order bits of x, so $a_k = x$. Then we code the sequence $\langle b_1, ..., b_k \rangle$, where $b_i = 2^{a_i}$, so $y = b_k$.

We have (note that $x_{k-1} = 1$):

(32)
$$\begin{aligned} &a_1 = 1, \qquad b_1 = 2 \\ &\text{For } 1 \leq i < k\text{: } a_{i+1} = x_{k-i-1} + 2a_i \qquad b_{i+1} = 2^{x_{k-i-1}} b_i^2 \end{aligned}$$

Note that $a_i < 2^i$ and $b_i < 2^{2^i}$ for $1 \leq i \leq k$.

We will code the sequences $\langle a_1, \dots, a_k \rangle$ and $\langle b_1, \dots, b_k \rangle$ by the numbers a and b, respectively, such that a_i and b_i are represented by the bits 2^i to $2^{i+1} - 1$ of a and b, respectively. In order to extract a_i and b_i from a and b we use the function

(33)
$$ext(u, z) = \lfloor z/u \rfloor \bmod u$$

Thus if $u = 2^{2^i}$ then $a_i = ext(u, a)$ and $b_i = ext(u, b)$. It is easy to see that the function ext is Δ_0-definable in $\mathbf{I\Delta_0}$.

Note that $a, b < 2^{2^{k+1}}$, and $y \geq 2^{2^{k-1}}$. Hence the numbers a and b can be bounded by $a, b < y^4$. Below we will explain how to express the condition that a number has the form 2^{2^i}. Once this is done, we can express

(34)
$$\begin{aligned} &y = 2^x \leftrightarrow \varphi_{exp}(x, y) \\ \text{where} \quad &\varphi_{exp} \equiv (x = 0 \wedge y = 1) \vee \exists a, b < y^4 \psi_{exp}(x, y, a, b) \end{aligned}$$

and $\psi_{exp}(x, y, a, b)$ is the formula stating that the following conditions (expressing the above recurrences) hold, for $x > 0, y > 1$:

1) $ext(2^{2^1}, a) = 1$, and $ext(2^{2^1}, b) = 2$
2) For all u, $2^{2^1} \leq u \leq y$ of the form 2^{2^i}, either

(a) $ext(u^2, a) = 2ext(u, a)$ and $ext(u^2, b) = (ext(u, b))^2$, or
(b) $ext(u^2, a) = 1 + 2ext(u, a)$ and $ext(u^2, b) = 2(ext(u, b))^2$.

3) There is $u \leq y^2$ of the form 2^{2^i} such that $ext(u, a) = x$ and $ext(u, b) = y$.

Note that condition (2)(a) holds if $x_{k-i} = 0$, and condition (2)(b) holds if $x_{k-i} = 1$. The conditions do not need to mention x_{k-i} explicitly, because condition (3) ensures that $a_i = x$ for some i, so all bits of x must have been chosen correctly up to this point.

It remains to express "x has the form 2^{2^i}". First, the set of numbers of the form

$$m_\ell = \sum_{i=0}^{\ell} 2^{2^i}$$

can be Δ_0-defined by the formula

$$\begin{aligned}\varphi_p(x) \equiv &\neg LenBit(1, x) \wedge LenBit(2, x) \wedge \\ &\forall 2^i \leq x,\ 2 < 2^i \supset (LenBit(2^i, x) \leftrightarrow (Pow4(2^i) \wedge LenBit(\lfloor\sqrt{2^i}\rfloor, x)))\end{aligned}$$

From this we can Δ_0-define numbers of the form $x = 2^{2^i}$ as the powers of 2 for which $LenBit(x, m_\ell)$ holds for some $m_\ell < 2x$:

x is of form 2^{2^i} : $\quad PPow2(x) \leftrightarrow Pow2(x) \wedge \exists m < 2x\, (\varphi_p(m) \wedge LenBit(x, m))$

This completes our description of the defining axiom $\varphi_{exp}(x, y)$ for the relation $y = 2^x$. It remains to show that $\mathbf{I}\Delta_0$ proves some properties of this relation. First we need to verify in $\mathbf{I}\Delta_0$ the properties of *PPow2*.

EXERCISE 3.51. *The following are theorems of* $\mathbf{I}\Delta_0$*:*
(a) $PPow2(z) \leftrightarrow PPow2(z^2)$.
(b) $(PPow2(z) \wedge PPow2(z') \wedge z < z') \supset z^2 \leq z'$.
(c) $(PPow2(x) \wedge 4 \leq x) \supset \lfloor\sqrt{x}\rfloor^2 = x$.

We have noted earlier that $a_i < 2^i$ and $b_i < 2^{2^i}$. Here we need to show that these are indeed provable in $\mathbf{I}\Delta_0$. We will need this fact in order to prove (in $\mathbf{I}\Delta_0$) the correctness of our defining axiom φ_{exp} for the relation $y = 2^x$ (e.g., Exercise 3.53 (c) and (d)).

EXERCISE 3.52. *Assuming* $(y > 1 \wedge \psi_{exp}(x, y, a, b))$*, show in* $\mathbf{I}\Delta_0$ *that*
(a) $\forall u \leq y^2,\ (PPow2(u) \wedge 4 \leq u) \supset 1 + ext(u, a) < u$.
(b) $\forall u \leq y^2,\ (PPow2(u) \wedge 4 \leq u) \supset 2ext(u, b) \leq u$.

EXERCISE 3.53. *Show that* $\mathbf{I}\Delta_0$ *proves the following:*
(a) $\varphi_{exp}(x, y) \supset Pow2(y)$.
(b) $Pow2(y) \supset \exists x < y\ \varphi_{exp}(x, y)$. *(Hint: strong induction on y, using Exercise 3.46 (f).)*
(c) $\varphi_{exp}(x, y_1) \wedge \varphi_{exp}(x, y_2) \supset y_1 = y_2$.
(d) $\varphi_{exp}(x_1, y) \wedge \varphi_{exp}(x_2, y) \supset x_1 = x_2$.
(e) $\varphi_{exp}(x + 1, 2y) \leftrightarrow \varphi_{exp}(x, y)$. *(Hint: Look at the least significant 0 bit of x.)*

(f) $\varphi_{exp}(x_1, y_1) \wedge \varphi_{exp}(x_2, y_2) \supset \varphi_{exp}(x_1+x_2, y_1 \cdot y_2)$ *(Hint: Induction on y_2.)*

Although the function 2^x is not Δ_0-definable in $\mathbf{I}\Delta_0$, it is easy to see using φ_{exp} (and useful to know) that the function

$$Exp(x, y) = \min(2^x, y)$$

is Δ_0-definable in $\mathbf{I}\Delta_0$.

Exercise 3.54. *The relation $y = z^x$ can be defined using the same techniques that have been used to define the relation $y = 2^x$. Here the sequence $\langle b_1, \ldots, b_k \rangle$ needs to be modified.*

(a) *Modify the recurrence in* (32).

Each b_i now may not fit in the bits 2^i to $2^{i+1} - 1$ of b, but it fits in a bigger segment of b. Let ℓ be the least number such that

$$z \leq 2^{2^\ell}$$

(b) *Show that for $1 \leq i \leq k$, $zb_i \leq 2^{2^{\ell+i}}$*

(c) *Show that the function $lpp(z)$, which is the least number of the form 2^{2^i} that is $\geq z$, is Δ_0-definable in $\mathbf{I}\Delta_0$.*

(d) *Show that $\mathbf{I}\Delta_0 \vdash z > 1 \supset (z \leq lpp(z) < z^2)$.*

(e) *What are the bounds on the values of the numbers a and b that respectively code the sequences $\langle a_1, \ldots, a_k \rangle$ and $\langle b_1, \ldots, b_k \rangle$?*

(f) *Give a formula that defines the relation $y = z^x$ by modifying the conditions 1–3.*

3C.3.2. *The BIT and NUMONES Relations.* The relation $BIT(i, x)$ can be defined as follows, where $BIT(i, x)$ holds iff the i-th bit (i.e., coefficient of 2^i) of the binary notation for x is 1:

$$BIT(i, x) \leftrightarrow \exists z \leq x(z = 2^i \wedge LenBit(z, x)) \tag{35}$$

Exercise 3.55. *Show that the length function, $|x| = \lceil \log_2(x+1) \rceil$, is Δ_0-definable in $\mathbf{I}\Delta_0$.*

Lemma 3.56. *The relation $NUMONES(x, y)$, asserting that y is the number of one-bits in the binary notation for x, is Δ_0-definable.*

Proof sketch. We code a sequence $\langle s_0, s_1, \ldots, s_n \rangle$ of numbers s_i of at most ℓ bits each using a number s such that bits $i\ell$ to $i\ell + \ell - 1$ of s are the bits of s_i. Then we can extract s_i from s using the equation

$$s_i = \lfloor s/2^{i\ell} \rfloor \bmod 2^\ell$$

Our first attempt to define $NUMONES(x, y)$ might be to state the existence of a sequence $\langle s_0, s_1, \ldots, s_n \rangle$, where $n = |x|$, s_i is the number of ones in the first i bits of x, and $\ell = ||x||$. However the number coding this sequence has $n \log n$ bits, which is too many.

We get around this problem using "Bennett's Trick" [10], which is to state the existence of a sparse subsequence of $\langle s_0, s_1, \ldots, s_n\rangle$ and assert that adjacent pairs in the subsequence can be filled in. Thus

$$\begin{gathered} \mathit{NUMONES}(x,y) \ \leftrightarrow\ \exists m \leq |x|\big(|x| \leq m^2 \wedge \\ \exists\langle t_0,\ldots,t_m\rangle\big(t_0 = 0 \wedge t_m = y \wedge \forall i < m\ \exists\langle u_0,\ldots,u_m\rangle \\ (u_0 = t_i \wedge u_m = t_{i+1} \wedge \forall j < m\ (u_{j+1} = u_j + \mathit{FBIT}(im+j,x))))\big)\big) \end{gathered}$$

where the function $\mathit{FBIT}(i,x)$ is bit i of x. ⊣

3D. $\mathbf{I\Delta_0}$ and the Linear Time Hierarchy

3D.1. The Polynomial and Linear Time Hierarchies. An element of a complexity class such as $\mathbf{P}$ (polynomial time) is often taken to be a *language* L, where L is a set of finite strings over some fixed finite alphabet Σ. In the context of bounded arithmetic, it is convenient to consider elements of $\mathbf{P}$ to be subsets of $\mathbb{N}$, or more generally relations over $\mathbb{N}$, and in this case it is assumed that numbers are presented in binary notation to the accepting machine. In this context, the notation Σ_0^p is sometimes used for polynomial time. Thus $\Sigma_0^p = \mathbf{P}$ is the set of all relations $R(x_1,...,x_k), k \geq 1$ over $\mathbb{N}$ such that some polynomial time Turing machine M_R, given input $x_1,...,x_k$ (k numbers in binary notation separated by blanks) determines whether $R(x_1,...,x_k)$ holds.

The class Σ_0^p has a generalization to $\Sigma_i^p, i \geq 0$, which is the i-th level of the polynomial-time hierarchy. This can be defined inductively by the recurrence

$$\Sigma_{i+1}^p = \mathbf{NP}^{\Sigma_i^p}$$

where $\mathbf{NP}^{\Sigma_i^p}$ is the set of relations accepted by a nondeterministic polynomial time Turing machine which has access to an oracle in Σ_i^p.

For $i \geq 1$, Σ_i^p can be characterized as the set of relations accepted by some alternating Turing machine (ATM) in polynomial time, making at most i alternations, beginning with an existential state. In any case,

$$\Sigma_1^p = \mathbf{NP}$$

We define the polynomial time hierarchy by

$$\mathbf{PH} = \bigcup_{i=0}^{\infty} \Sigma_i^p$$

In the context of $\mathbf{I\Delta_0}$, we are interested in the Linear Time Hierarchy (**LTH**), which is defined analogously to **PH**. We use **NLinTime** to denote time $O(n)$ on a nondeterministic multi-tape Turing machine. Then

$$\Sigma_1^{lin} = \mathbf{NLinTime} \tag{36}$$

and for $i \geq 1$

$$\Sigma_{i+1}^{lin} = \mathbf{NLinTime}^{\Sigma_i^{lin}} \tag{37}$$

Alternatively, we can define Σ_i^{lin} to be the relations accepted in linear time on an ATM with i alternations, beginning with an existential state. In either case,[2]

$$\mathbf{LTH} = \bigcup_{i=1}^{\infty} \Sigma_i^{lin}$$

LinTime is not as robust a class as polynomial time; for example it is plausible that a $k+1$-tape deterministic linear time Turing machine can accept sets not accepted by any k tape such machine, and linear time Random Access Machines may accept sets not in **LinTime**. However it is not hard to see that **NLinTime** is more robust, in the sense that every set in this class can be accepted by a two tape nondeterministic linear time Turing machine.

3D.2. Representability of LTH Relations. Recall the definition of definable predicates and functions (Definition 3.27). If Φ is a class of $\mathcal{L}$-formulas, $\mathcal{T}$ a theory over $\mathcal{L}$, and R a Φ-definable relation (over the natural numbers) in $\mathcal{T}$, then we simply say that R is Φ-definable (or Φ-*representable*).

Thus when Φ is a class of $\mathcal{L}_A$-formulas, a k-*ary* relation R over the natural numbers is Φ-definable if there is a formula $\varphi(x_1, \ldots, x_k) \in \Phi$ such that for all $(n_1, \ldots, n_k) \in \mathbb{N}^k$,

$$(n_1, \ldots, n_k) \in R \quad \text{iff} \quad \underline{\mathbb{N}} \models \varphi(n_1, \ldots, n_k) \tag{38}$$

More generally, if Φ is a class of $\mathcal{L}$-formulas for some language $\mathcal{L}$ extending $\mathcal{L}_A$, then instead of $\underline{\mathbb{N}}$ we will take the expansion of $\underline{\mathbb{N}}$ where the extra symbols in $\mathcal{L}$ have their intended meaning.

(Note that a relation $R(\vec{x})$ is sometimes called *representable* (or *weakly representable*) *in a theory* $\mathcal{T}$ if there is some formula $\varphi(\vec{x})$ so that for all $\vec{n} \in \mathbb{N}$,

$$R(\vec{n}) \quad \text{iff} \quad \mathcal{T} \vdash \varphi(\underline{\vec{n}})$$

Our notation here is the special case where $\mathcal{T} = \mathbf{TA}$.)

For example, the class of Σ_1-representable sets (i.e., unary relations) is precisely the class of r.e. sets. In the context of Buss's $\mathbf{S}_2^i$ hierarchy (Section 3E), **NP** relations are precisely the Σ_1^b-representable relations. (Σ_1^b is defined for the language $\mathcal{L}_{\mathbf{S}_2}$ of $\mathbf{S}_2$.) Here we show that the **LTH** relations are exactly the Δ_0-representable relations.

DEFINITION 3.57. $\Delta_0^{\mathbb{N}}$ *is the class of* Δ_0*-representable relations.*

[2] **LTH** is different from **LH**, the *logtime-hierarchy* discussed in Section 4A

For instance, we have shown that the relations *BIT* and *NUMONES* are in $\Delta_0^{\mathbb{N}}$. So is the relation $Prime(x)$ (x is a prime number), because

$$Prime(x) \equiv 1 < x \wedge \forall y < x \forall z < x(y \cdot z \neq x)$$

THEOREM 3.58 (**LTH** Theorem). $\mathbf{LTH} = \Delta_0^{\mathbb{N}}$

PROOF SKETCH. First consider the inclusion $\mathbf{LTH} \subseteq \Delta_0^{\mathbb{N}}$. This can be done using the recurrence (36), (37). The hard part here is the base case, showing $\mathbf{NLinTime} \subseteq \Delta_0^{\mathbb{N}}$. Once this is done we can show the induction step by, given a nondeterministic linear time oracle Turing machine M, defining the relation $R_M(x, y, b)$ to assert "M accepts input x, assuming that it makes the sequence of oracle queries coded by y, and the answers to those queries are coded by b." This relation R_M is accepted by some nondeterministic linear time Turing machine (with no oracle), and hence it is in $\Delta_0^{\mathbb{N}}$ by the base case.

To show $\mathbf{NLinTime} \subseteq \Delta_0^{\mathbb{N}}$ we need to represent the computation of a nondeterministic linear time Turing machine by a constant number k of strings $x_1, \ldots, x_k$ of linear length. One string will code the sequence of states of the computation, and for each tape there is a string coding the sequence of symbols printed and another string coding the head moves. In order to check that the computation is correctly encoded it is necessary to deduce the position of each tape head at each step of the computation, from the sequence of head moves. This can be done by counting the number of left shifts and of right shifts, using the relation $NUMONES(x, y)$, and subtracting. It is also necessary to determine the symbol appearing on a given tape square at a given step, and this can be done by determining the last time that the head printed a symbol on that square.

We prove the inclusion $\Delta_0^{\mathbb{N}} \subseteq \mathbf{LTH}$ by structural induction on Δ_0 formulas. The induction step is easy, since bounded quantifiers correspond to $\exists$ and $\forall$ states in an ATM. The only interesting case is one of the base cases: the atomic formula $x \cdot y = z$. To show that this relation $R(x, y, z)$ is in **LTH** we use Corollary 3.61 below which shows that **L** $\subseteq$ **LTH**. (**L** is the class of relations computable in logarithmic space using Turing machines. See Appendix A.1.) It is not hard to see that using the school algorithm for multiplication the relation $x \cdot y = z$ can be checked in space $O(\log n)$, and thus it is in **L**. $\dashv$

EXERCISE 3.59. *Give more details of the proof showing* $\mathbf{LTH} \subseteq \Delta_0^{\mathbb{N}}$.

THEOREM 3.60 (Nepomnjaščij's Theorem). *Let ϵ be a rational number, $0 < \epsilon < 1$, and let a be a positive integer. Then*

$$\mathbf{NTimeSpace}(n^a, n^\epsilon) \subseteq \mathbf{LTH}$$

In the above, $\mathbf{NTimeSpace}(f(n), g(n))$ consists of all relations accepted simultaneously in time $O(f(n))$ and space $O(g(n))$ on a *nondeterministic multi-tape* Turing machine.

Proof Idea. We use Bennett's Trick, as in the proof of Lemma 3.56. Suppose we want to show

$$\mathbf{NTimeSpace}(n^2, n^{0.6}) \subseteq \mathbf{LTH}$$

Let M be a nondeterministic TM running in time n^2 and space $n^{0.6}$. Then M accepts an input x iff

$$\exists \vec{y}(\vec{y} \text{ represents an accepting computation for } x)$$

Here $\vec{y} = y_1, ..., y_{n^2}$, where each y_i is a string of length $n^{0.6}$ representing a configuration of M. The total length of $\vec{y}$ is $|\vec{y}| = n^{2.6}$, which is too long for an ATM to guess in linear time.

So we guess a vector $\vec{z} = z_1, ..., z_n$ representing every n-th string in $\vec{y}$, so now M accepts x iff

$$\exists \vec{z} \forall i < n \exists \vec{u}(\vec{u} \text{ shows } z_{i+1} \text{ follows from } z_i \text{ in } n \text{ steps and } z_n \text{ is accepting})$$

Now the lengths of $\vec{z}$ and $\vec{u}$ are only $n^{1.6}$, and we have made progress. Two more iterations of this idea (one for the $\exists\vec{y}$, one for the $\exists\vec{u}$; increasing the nesting depth of quantifiers) will get the lengths of the quantified strings below linear. $\dashv$

For the following corollary, **NL** is the class of relations computable by *nondeterministic* Turing machines in logarithmic space. See Appendix B.

Corollary 3.61. $\mathbf{NL} \subseteq \mathbf{LTH}$.

Proof. We use the fact that $\mathbf{NL} \subseteq \mathbf{NTimeSpace}(n^{O(1)}, \log n)$. $\dashv$

Remark. We know

$$\mathbf{L} \subseteq \mathbf{LTH} \subseteq \mathbf{PH} \subseteq \mathbf{PSPACE}$$

where no two adjacent inclusions are known to be proper, although we know $\mathbf{L} \subset \mathbf{PSPACE}$ by a simple diagonal argument.

Also $\mathbf{LTH} \subseteq \mathbf{LinSpace} \subset \mathbf{PSPACE}$, where the first inclusion is not known to be proper. Finally **P** and **LTH** are thought to be incomparable, but no proof is known. In fact it is difficult to find a natural example of a problem in **P** which seems not to be in **LTH**.

3D.3. Characterizing the LTH by IΔ_0. First note that **LTH** is a class of *relations*. The corresponding class of functions is defined in terms of *function graphs*. Given a function $f(\vec{x})$, its graph $G_f(\vec{x}, y)$ is the relation

$$G_f(\vec{x}, y) \leftrightarrow (y = f(\vec{x}))$$

Definition 3.62 (**FLTH**). *A function $f : \mathbb{N}^k \to \mathbb{N}$ is in* **FLTH** *if its graph $G_f(\vec{x}, y)$ is in* **LTH** *and its length has at most linear growth, i.e.,*

$$f(\vec{x}) = (x_1 + ... + x_k)^{O(1)}$$

EXERCISE 3.63. *In future chapters we will define the class of functions associated with a class of relations using the bit graph* $B_f(i, \vec{x}, y)$ *of* f *instead of the graph* $G_f(\vec{x}, y)$*, where*

$$B_f(i, \vec{x}) \leftrightarrow BIT(i, f(\vec{x}))$$

Show that the class **FLTH** *remains the same if* B_f *replaces* G_f *in the above definition.*

In general, in order to associate a theory with a complexity class we should show that the functions in the class coincide with the Σ_1-definable functions in the theory. The next result justifies associating the theory $\mathbf{I}\Delta_0$ with the complexity class **LTH**.

THEOREM 3.64 ($\mathbf{I}\Delta_0$-Definability Theorem). *A function is* Σ_1*-definable in* $\mathbf{I}\Delta_0$ *iff it is in* **FLTH**.

PROOF. The $\Longrightarrow$ direction follows from the Bounded Definability Theorem 3.33, the above definition of **LTH** functions and the **LTH** Theorem 3.58.

For the $\Longleftarrow$ direction, suppose $f(\vec{x})$ is an **LTH** function. By definition the graph $(y = f(\vec{x}))$ is an **LTH** relation, and hence by the **LTH** Theorem 3.58 there is a Δ_0 formula $\varphi(\vec{x}, y)$ such that

$$y = f(\vec{x}) \leftrightarrow \varphi(\vec{x}, y)$$

Further, by definition, $|f(\vec{x})|$ is linear bounded, so there is an $\mathcal{L}_A$-term $t(\vec{x})$ such that

$$f(\vec{x}) \le t(\vec{x}) \tag{39}$$

The sentence $\forall \vec{x} \exists ! y \varphi(\vec{x}, y)$ is true, but unfortunately there is no reason to believe that it is provable in $\mathbf{I}\Delta_0$. We can solve the problem of proving uniqueness by taking the least y satisfying $\varphi(\vec{x}, y)$. In general, for any formula $A(y)$, we define $Min_y[A(y)](y)$ to mean that y is the least number satisfying $A(y)$. Thus

$$Min_y[A(y)](y) \equiv_{def} A(y) \wedge \forall z < y(\neg A(z))$$

If $A(y)$ is bounded, then we can apply the least number principle to $A(y)$ to obtain

$$\mathbf{I}\Delta_0 \vdash \exists y A(y) \supset \exists ! y Min_y[A(y)](y) \tag{40}$$

This solves the problem of proving uniqueness. To prove existence, we modify φ and define

$$\psi(\vec{x}, y) \equiv_{def} (\varphi(\vec{x}, y) \vee y = t(\vec{x}) + 1)$$

where $t(\vec{x})$ is the bounding term from (39). Now define

$$\varphi'(\vec{x}, y) \equiv Min_y[\psi(\vec{x}, y)](\vec{x}, y)$$

Then $\varphi'(\vec{x}, y)$ also represents the relation $(y = f(\vec{x}))$, and since trivially $\mathbf{I}\Delta_0$ proves $\exists y \psi(\vec{x}, y)$ we have by (40)

$$\mathbf{I}\Delta_0 \vdash \forall \vec{x} \exists ! y \varphi'(\vec{x}, y) \qquad \dashv$$

3E. Buss's S_2^i Hierarchy: The Road Not Taken

Buss's PhD thesis *Bounded Arithmetic* (published as a book in 1986, [12]) introduced the hierarchies of bounded theories

$$\mathbf{S}_2^1 \subseteq \mathbf{T}_2^1 \subseteq \mathbf{S}_2^2 \subseteq \mathbf{T}_2^2 \subseteq \ldots \subseteq \mathbf{S}_2^i \subseteq \mathbf{T}_2^i \subseteq \ldots$$

These theories, whose definable functions are those in the polynomial hierarchy, are of central importance in the area of bounded arithmetic.

Here we present a brief overview of the original theories $\mathbf{S}_2^i$ and $\mathbf{T}_2^i$, and their union $\mathbf{S}_2 = \mathbf{T}_2 = \bigcup_{i=1}^{\infty} \mathbf{S}_2^i$. The idea is to modify the theory $\mathbf{I}\Delta_0$ so that the definable functions are those in the polynomial hierarchy as opposed to the Linear Time Hierarchy, and more importantly to introduce the theory $\mathbf{S}_2^1$ whose definable functions are precisely the polynomial time functions. In order to do this, the underlying language is augmented to include the function symbol $\#$, whose intended interpretation is $x\#y = 2^{|x|\cdot|y|}$. Thus terms in $\mathbf{S}_2$ represent functions which grow at the rate of polynomial time functions, as opposed to the linear-time growth rate of $\mathbf{I}\Delta_0$ terms. The full vocabulary for $\mathbf{S}_2$ is

$$\mathcal{L}_{\mathbf{S}_2} = [0, S, +, \cdot, \#, |x|, \lfloor \frac{1}{2}x \rfloor;\ =, \leq]$$

(S is the *Successor* function, $|x|$ is the length (of the binary representation) of x).

Sharply bounded quantifiers have the form $\forall x \leq |t|$ or $\exists x \leq |t|$ (where x does not occur in t). These are important because sharply bounded (as opposed to just bounded) formulas represent polynomial time relations (and in fact $\mathbf{TC}^0$ relations). The syntactic class Σ_i^b (b for "bounded") consists essentially of those formulas with at most i blocks of bounded quantifiers beginning with $\exists$, with any number of sharply bounded quantifiers of both kinds mixed in. The formulas in Σ_1^b represent precisely the $\mathbf{NP}$ relations, and more generally formulas in Σ_i^b represent precisely the relations in the level Σ_i^p in the polynomial hierarchy. In summary, bounded formulas in the language of $\mathbf{S}_2$ represent precisely the relations in the polynomial hierarchy.

The axioms for $\mathbf{T}_2^i$ consist of 32 $\forall$-sentences called **BASIC** which define the symbols of $\mathcal{L}_{\mathbf{S}_2}$, together with the Σ_i^b-**IND** scheme. The axioms for $\mathbf{S}_2^i$ are the same as those of $\mathbf{T}_2^i$, except for Σ_i^b-**IND** is replaced by the Σ_i^b-**PIND** scheme:

$$\big(\varphi(0) \wedge \forall x(\varphi(\lfloor \frac{1}{2}x \rfloor) \supset \varphi(x))\big) \supset \forall x \varphi(x)$$

where $\varphi(x)$ is any Σ_i^b formula. Note that this axiom scheme is true in $\underline{\mathbb{N}}$. Also for $i \geq 1$, $\mathbf{T}_2^i$ proves the Σ_i^b-**PIND** axiom scheme, and $\mathbf{S}_2^{i+1}$ proves the Σ_i^b-**IND** axiom scheme. (Thus for $i \geq 1$, $\mathbf{S}_2^i \subseteq \mathbf{T}_2^i \subseteq \mathbf{S}_2^{i+1}$.)

For $i \geq 1$, the functions Σ_i^b-definable in $\mathbf{S}_2^i$ are precisely those polytime reducible to relations in Σ_{i-1}^p (level $i-1$ of the polynomial hierarchy). In particular, the functions Σ_1^b-definable in $\mathbf{S}_2^1$ are precisely the polynomial time functions.

Since $\mathbf{S}_2$ is a polynomial-bounded theory, Parikh's Theorem 3.20 can be applied to show that all Σ_1-definable functions in $\mathbf{S}_2$ are polynomial time reducible to $\mathbf{PH}$. To show that the Σ_1^b-definable functions in $\mathbf{S}_2^1$ are polynomial-time computable requires a more sophisticated "witnessing" argument introduced by Buss. We shall present this argument later in the context of the two-sorted first-order theory $\mathbf{V}^1$.

In Chapters 5, 6 and 8 we will present two-sorted versions $\langle \mathbf{V}^i \rangle$ of $\langle \mathbf{S}_2^i \rangle$ and $\langle \mathbf{TV}^i \rangle$ of $\langle \mathbf{T}_2^i \rangle$. With the exception of $\mathbf{V}^0$ and $\mathbf{TV}^0$ (which have no corresponding theories in the $\mathbf{S}_2^i$ hierarchy), the two-sorted versions are essentially equivalent to the originals, but are simpler and naturally represent complexity classes on strings as opposed to numbers.

3F. Notes

The main references for this chapter are [19, 20] and [41, pp 277–293].

Parikh's Theorem originally appears in [67], and the proof there is based in the Herbrand Theorem, and resembles our "Alternative Proof" given at the end of Section 3C.2. Buss [12] gives a proof based on cut elimination which is closer to our first proof.

James Bennett [10] was the first to show that the relation $y = z^x$ can be defined by Δ_0 formulas. Hájek and Pudlák [41] give a different definition and show how to prove its basic properties in $\mathbf{I}\Delta_0$, and give a history of such definitions and proofs. Our treatment of the relations $y = 2^x$ and $BIT(i, x)$ in Section 3C.3 follows that of Buss in [20], simplified with an idea from earlier proofs.

Bennett's Trick, described in the proof of Lemma 3.56, is due to Bennett [10] Section 1.7, where it is used to show that the rudimentary functions are closed under a form of bounded recursion on notation.

Theorem 3.58, stating $\mathbf{LTH} = \Delta_0^{\mathbb{N}}$, is due to Wrathall [84]. Nepomnjaščij's Theorem 3.60 appears in [62].

Chapter 4

TWO-SORTED LOGIC AND COMPLEXITY CLASSES

In this chapter we introduce two-sorted first-order logic (sometimes called second-order logic), an extension of the (single-sorted) first-order logic that we use in the previous chapters. The reason for using two-sorted logic is that our theories capture complexity classes defined in terms of Turing machines or Boolean circuits. The inputs to these devices are bit strings, whereas the objects in the universe of discourse in our single-sorted theories are numbers. Although we can code numbers by bit strings using binary notation, this indirection is sometimes awkward, especially for low-level complexity classes. In particular, our single-sorted theories all include multiplication as a primitive operation, but binary multiplication is not in the complexity class $\mathbf{AC}^0$, whose theory $\mathbf{V}^0$ serves as the basis for all our two-sorted theories. Our complexity reductions and completeness notions are generally defined using $\mathbf{AC}^0$ functions.

The two-sorted theories retain the natural numbers as the first sort, and the objects in the second sort are bit strings (precisely, finite sets of natural numbers, whose characteristic vectors are bit strings). We need the first sort (numbers) in order to reason about the second sort. The numbers involved for this reasoning are small; they are used to index bit positions in the second sort (strings). In defining two-sorted complexity classes, the number inputs to the devices are coded in unary notation, and are treated as auxiliary to the main (second) sort, whose elements are coded by binary strings. In particular we use these conventions to define the two-sorted complexity class $\mathbf{AC}^0$. We prove the Σ_0^B Representation Theorem 4.18, which states that the set Σ_0^B of two-sorted formulas represent precisely the $\mathbf{AC}^0$ relations.

In later chapters we show how to translate bounded theorems in our theories into families of propositional proofs. This translation is made especially simple and elegant by using two-sorted theories.

The historical basis for using two-sorted logic to represent complexity classes is descriptive complexity theory, where each object (a language or a relation) in a complexity class is described by a logical formula whose set of finite models corresponds to the object. In the two-sorted logic setting, each object corresponds to the set of interpretations of a variable in the formula satisfying the formula in the standard model.

In the first part of this chapter we present a brief introduction to descriptive complexity theory. (A comprehensive treatment can be found in [45].) Then we introduce two-sorted first-order logic, describe two-sorted complexity classes, and explain how relations in these classes are represented by certain classes of formulas. We revisit the **LTH** theorem for two-sorted logic. We present the sequent calculus $\mathbf{LK}^2$, the two-sorted version of **LK**. Finally we show how to interpret two-sorted logic into single-sorted logic.

4A. Basic Descriptive Complexity Theory

In descriptive complexity theory, an object (e.g. a set of graphs) in a complexity class is specified as the set of all finite models of a given formula. Here we consider the case in which the object is a language $L \subseteq \{0,1\}^*$, and the formula is a formula of the first-order predicate calculus. We assume that the underlying vocabulary consists of

$$\mathcal{L}_{FO} = [0, max; X, BIT, \leq, =], \tag{41}$$

where 0, max are constants, X is a unary predicate symbol, and BIT, $\leq$, $=$ are binary predicate symbols. We consider finite $\mathcal{L}_{FO}$-structures $\mathcal{M}$ in which the universe $M = \{0, \ldots, n-1\}$ for some natural number $n \geq 1$, and max is interpreted by $n-1$. The symbols 0, $=$, $\leq$, and BIT receive their standard interpretations. (Recall that $BIT(i, x)$ holds iff the i-th bit in the binary representation of x is 1. In the previous chapter we showed how to define BIT in $\mathbf{I}\Delta_0$, but note that here it is a primitive symbol in $\mathcal{L}_{FO}$.)

Thus the only symbol without a fixed interpretation is the unary predicate symbol X, and to specify a structure it suffices to specify the tuple of truth values $\langle X(0), X(1), ..., X(n-1)\rangle$. By identifying $\top$ with 1 and $\bot$ with 0, we see that there is a natural bijection between the set of structures and the set $\{0,1\}^+$ of nonempty binary strings.

The class **FO** (First-Order) of languages describable by $\mathcal{L}_{FO}$ formulas is defined as follows. First, for each binary string X, we denote by $\mathcal{M}[X]$ the structure which is specified by the binary string X. Then the language $L(\varphi)$ associated with an $\mathcal{L}_{FO}$ sentence φ is the set of strings whose associated structures satisfy φ:

$$L(\varphi) = \{X \in \{0,1\}^+ \mid \mathcal{M}[X] \models \varphi\}.$$

Definition 4.1 (The Class **FO**).

$$\mathbf{FO} = \{L(\varphi) \mid \varphi \textit{ is an } \mathcal{L}_{FO}\textit{-sentence}\}$$

For example, let L_{even} be the set of strings whose even positions (starting from the right at position 0) have 1. Then $L_{\text{even}} \in \mathbf{FO}$, since

$L_{\text{even}} = L(\varphi)$, where

$$\varphi \equiv \forall y(\neg BIT(0, y) \supset X(y)).$$

To give a more interesting example, we use the fact [45, page 14] that the relation $x + y = z$ can be expressed by a first-order formula $\varphi_+(x, y, z)$ in the vocabulary $\mathcal{L}_{FO}$. Then the set PAL of binary palindromes is represented by the sentence

$$\forall x \forall y,\ \varphi_+(x, y, max) \supset (X(x) \leftrightarrow X(y)).$$

Thus $PAL \in \mathbf{FO}$.

Immerman showed that the class **FO** is the same as a uniform version of $\mathbf{AC}^0$ (see Appendix D.1). Originally $\mathbf{AC}^0$ was defined in its nonuniform version, which we shall refer to as $\mathbf{AC}^0/poly$. A language in $\mathbf{AC}^0/poly$ is specified by a polynomial size bounded depth family $\langle C_n \rangle$ of Boolean circuits, where each circuit C_n has n input bits, and is allowed to have $\neg$-gates, as well as unbounded fan-in $\wedge$-gates and $\vee$-gates. In the uniform version, the circuit C_n must be specified in a uniform way; for example one could require that $\langle C_n \rangle$ is in **FO**.

Immerman showed that this definition of uniform $\mathbf{AC}^0$ is robust, in the sense that it has several quite different characterizations. For example, the logtime hierarchy **LH** consists of all languages recognizable by an ATM (Alternating Turing Machine) in time $O(\log n)$ with a constant number of alternations. Also **CRAM**[1] consists of all languages recognizable in constant time on a so-called Concurrent Random Access Machine. The following theorem is from [45, Corollary 5.32].

Theorem 4.2.

$$\mathbf{FO} \ = \ \mathbf{AC}^0 \ = \ \mathbf{CRAM}[1] \ = \ \mathbf{LH}.$$

Of course the nonuniform class $\mathbf{AC}^0/poly$ contains non-computable sets, and hence it properly contains the uniform class $\mathbf{AC}^0$. Nevertheless in 1983 Ajtai [2] (and independently Furst, Saxe, and Sipser [40]) proved that even such a simple set as $PARITY$ (the set of all strings with an odd number of 1's) is not in $\mathbf{AC}^0/poly$ (and hence not in **FO**).

On the positive side, we pointed out that the set PAL of palindromes is in **FO**, and hence in $\mathbf{AC}^0$. If we code a triple $\langle U, V, W \rangle$ of strings as a single string in some reasonable way then it is easy to see using a carry look-ahead adder that binary addition (the set $\langle U, V, U+V \rangle$) is in $\mathbf{AC}^0$ (see page 83). Do not confuse this with the result of [45, page 14] mentioned above that some first-order formula $\phi_+(x, y, z)$ represents $x + y = z$, since here x, y, z represent elements in the model $\mathcal{M}$, which have nothing much to do with the input string X.

In fact $PARITY$ is efficiently reducible to binary multiplication, so Ajtai's result implies that the set $\langle U, V, U \cdot V \rangle$ is *not* in $\mathbf{AC}^0$. In contrast, there is a first-order formula in the vocabulary $\mathcal{L}_{FO}$ which represents $x \cdot y = z$ in standard model with universe $M = \{0, ..., n-1\}$.

4B. Two-Sorted First-Order Logic

4B.1. Syntax. Our two-sorted first-order logic is an extension of the (single-sorted) first-order logic introduced in Chapter 2. Here there are two kinds of variables: the variables $x, y, z, ...$ of the first sort are called *number variables*, and are intended to range over the natural numbers; and the variables $X, Y, Z, ...$ of the second sort are called *set* (or also *string*) *variables*, and are intended to range over finite subsets of natural numbers (which represent binary strings). Also the function and predicate symbols are now over both sorts.

Definition 4.3 (Two-sorted First-order Vocabularies). *A two-sorted first-order language (or just two-sorted language, or language, or vocabulary) $\mathcal{L}$ is specified by a set of function symbols and predicate symbols, just as in the case of a single-sorted language (Section 2B.1), except that the functions and predicates now can take arguments of both sorts, and there are two kinds of functions: the* number-valued functions *(or just* number functions*) and the* string-valued functions *(or just* string functions*).*

In particular, for each $n, m \in \mathbb{N}$, there is a set of (n, m)-ary number function symbols, a set of (n, m)-ary string function symbols, and a set of (n, m)-ary predicate symbols. An $(0, 0)$-ary function symbol is called a constant symbol, which can be either a number constant or a string constant.

We use $f, g, h, \ldots$ as meta-symbols for number function symbols; $F, G, H, \ldots$ for string function symbols; and $P, Q, R, \ldots$ for predicate symbols.

For example, consider the following two-sorted extension of $\mathcal{L}_A$ (Definition 2.22):

Definition 4.4. $\mathcal{L}_A^2 = [0, 1, +, \cdot, |\ |\ ;\ =_1,\ =_2, \leq, \in]$.

Here the symbols $0, 1, +, \cdot, =_1$ and $\leq$ are from $\mathcal{L}_A$; they are function and predicate symbols over the first sort ($=_1$ corresponds to $=$ of $\mathcal{L}_A$). The function $|X|$ (the "length of X") is a number-valued function and is intended to denote the least upper bound of the set X (roughly the length of the corresponding string). The binary predicate $\in$ takes a number and a set as arguments, and is intended to denote set membership. Finally, $=_2$ is the equality predicate for the second-sort objects. We will write $=$ for both $=_1$ and $=_2$, its exact meaning will be clear from the context.

We will use the abbreviation

$$X(t) =_{\text{def}} t \in X$$

where t is a number term (Definition 4.5 below). Thus we think of $X(i)$ as the i-th bit of the binary string X.

Note that in $\mathcal{L}_A^2$ the function symbols $+, \cdot$ each has arity $(2,0)$, while $|\ |$ has arity $(0,1)$ and the predicate symbol $\in$ has arity $(1,1)$.

For a two-sorted language $\mathcal{L}$, the notions of $\mathcal{L}$-terms and $\mathcal{L}$-formulas generalize the corresponding notions in the single-sorted case (Definitions 2.20 and 2.21). Here we have two kinds of terms: *number terms* and *string terms*. As before, we will drop mention of $\mathcal{L}$ when it is not important, or clear from the context. Also, we are interested only in vocabularies $\mathcal{L}$ that extend $\mathcal{L}_A^2$, and we may list only the elements of the set $\mathcal{L} - \mathcal{L}_A^2$ (sometimes without the braces $\{,\}$ for set). In such cases, the notations $\mathcal{L}$-terms, $\mathcal{L}$-formulas, $\Sigma_i^B(\mathcal{L})$, etc. refer really to the corresponding notions for $\mathcal{L} \cup \mathcal{L}_A^2$.

Definition 4.5 ($\mathcal{L}$-Terms). *Let $\mathcal{L}$ be a two-sorted vocabulary:*

1) *Every number variable is an $\mathcal{L}$-number term.*
2) *Every string variable is an $\mathcal{L}$-string term.*
3) *If f is an (n,m)-ary number function symbol of $\mathcal{L}$, $t_1, \dots, t_n$ are $\mathcal{L}$-number terms, and $T_1, \dots, T_m$ are $\mathcal{L}$-string terms, then $ft_1 \dots t_n T_1 \dots T_m$ is an $\mathcal{L}$-number term.*
4) *If F is an (n,m)-ary string function symbol of $\mathcal{L}$, and $t_1, \dots, t_n$ and $T_1, \dots, T_m$ are as above, then $Ft_1 \dots t_n T_1 \dots T_m$ is an $\mathcal{L}$-string term.*

Note that all constants in $\mathcal{L}$ are $\mathcal{L}$-terms.

We often denote number terms by $r, s, t, \dots$, and string terms by $S, T, \dots$.

The formulas over a two-sorted language $\mathcal{L}$ are defined as in the single-sorted case (Definition 2.21), with the addition of quantifiers over string variables. These are called *string quantifiers*, and the quantifiers over number variables are called *number quantifiers*. Also note that a predicate symbol in general may have arguments from both sorts.

Definition 4.6 ($\mathcal{L}$-Formulas). *Let $\mathcal{L}$ be a two-sorted first-order language. Then a* two-sorted first-order formula *in $\mathcal{L}$ (or* $\mathcal{L}$-formula*, or just* formula*) are defined inductively as follows:*

1) *If P is an (n,m)-ary predicate symbol of $\mathcal{L}$, $t_1, \dots, t_n$ are $\mathcal{L}$-number terms and $T_1, \dots, T_m$ are $\mathcal{L}$-string terms, then*
$$Pt_1 \dots t_n T_1 \dots T_m$$
is an atomic $\mathcal{L}$-formula. Also, each of the logical constants $\bot$, $\top$ is an atomic formula.
2) *If φ, ψ are $\mathcal{L}$-formulas, so are $\neg\varphi$, $(\varphi \wedge \psi)$, and $(\varphi \vee \psi)$.*
3) *If φ is an $\mathcal{L}$-formula, x is a number variable and X is a string variable, then $\forall x\varphi$, $\exists x\varphi$, $\forall X\varphi$ and $\exists X\varphi$ are $\mathcal{L}$-formulas.*

We often denote formulas by $\varphi, \psi, \dots$.

Recall that in $\mathcal{L}_A^2$ we write $X(t)$ for $t \in X$.

Example 4.7 ($\mathcal{L}_A^2$-Terms and $\mathcal{L}_A^2$-Formulas).

1) *The only string terms of* $\mathcal{L}_A^2$ *are the string variables* $X, Y, Z, \ldots$.
2) *The number terms of* $\mathcal{L}_A^2$ *are obtained from the constants 0, 1, number variables* $x, y, z, \ldots$, *and the lengths of the string variables* $|X|, |Y|, |Z|, \ldots$ *using the binary function symbols* $+, \cdot$.
3) *The only atomic formulas of* $\mathcal{L}_A^2$ *are* $\bot$, $\top$ *or those of the form* $s = t$, $X = Y$, $s \leq t$ *and* $X(t)$ *for string variables* X, Y *and number terms* s, t.

4B.2. Semantics. As for single-sorted first-order logic, the semantics of a two-sorted language is given by structures and object assignments. Here the universe of a structure contains two sorts of objects, one for the number variables and one for the string variables. As in the single-sorted case, we also require that the predicate symbols $=_1$ and $=_2$ must be interpreted as the true equality in the respective sort. The following definition generalizes the notion of a (single-sorted) structure given in Definition 2.25.

Definition 4.8 (Two-sorted Structures). *Let* $\mathcal{L}$ *be a two-sorted language. Then an* $\mathcal{L}$*-structure* $\mathcal{M}$ *consists of the following:*

1) *A pair of two nonempty sets* U_1 *and* U_2, *which together are called the* universe. *Number (resp. string) variables in an* $\mathcal{L}$*-formulas are intended to range over* U_1 *(resp.* U_2*).*
2) *For each* (n, m)*-ary number function symbol* f *of* $\mathcal{L}$ *an associated function* $f^{\mathcal{M}} : U_1^n \times U_2^m \to U_1$.
3) *For each* (n, m)*-ary string function symbol* F *of* $\mathcal{L}$ *an associated function* $F^{\mathcal{M}} : U_1^n \times U_2^m \to U_2$.
4) *For each* (n, m)*-ary predicate symbol* P *of* $\mathcal{L}$ *an associated relation* $P^{\mathcal{M}} \subseteq U_1^n \times U_2^m$.

Thus, for our "base" language $\mathcal{L}_A^2$, an $\mathcal{L}_A^2$-structure with universe $\langle U_1, U_2 \rangle$ contains the following interpretations of $\mathcal{L}_A^2$:

- Elements $0^{\mathcal{M}}, 1^{\mathcal{M}} \in U_1$ to interpret 0 and 1, respectively;
- Binary functions $+^{\mathcal{M}}, \cdot^{\mathcal{M}} : U_1 \times U_1 \to U_1$ to interpret $+$ and $\cdot$, respectively;
- A binary predicate $\leq^{\mathcal{M}} \subseteq U_1^2$ interpreting $\leq$;
- A function $|\ |^{\mathcal{M}} : U_2 \to U_1$;
- A binary relation $\in^{\mathcal{M}} \subseteq U_1 \times U_2$.

Note that in an $\mathcal{L}_A^2$-structure $\mathcal{M}$ as above, an element $\alpha \in U_2$ can be specified by the pair $(|\alpha|, S_\alpha)$, where $S_\alpha = \{u \in U_1 | u \in^{\mathcal{M}} \alpha\}$. Technically many different elements of U_2 could be represented by the same such pair. However, if we define an equivalence class on U_2 by stating two elements are equivalent if they have the same pair, then the structure and object assignment (see definition below) obtained by passing to equivalence classes satisfies exactly the same formulas as the original structure and object assignment. Therefore without loss of generality, we assume that every element α of U_2 is uniquely specified by $(|\alpha|, S_\alpha)$.

Example 4.9 (The Standard Two-sorted Model $\underline{\mathbb{N}}_2$). *The* standard model $\underline{\mathbb{N}}_2$ *has* $U_1 = \mathbb{N}$ *and* U_2 *the set of finite subsets of* $\mathbb{N}$*. The number part of the structure is the standard single-sorted first-order structure* $\underline{\mathbb{N}}$*. The relation* $\in$ *gets its usual interpretation (membership), and for each finite subset* $S \subseteq \mathbb{N}$*,* $|S|$ *is interpreted as one plus the largest element in* S*, or 0 if* S *is empty.*

As in the single-sorted case, the truth value of a formula in a structure is defined based on the interpretations of free variables occurring in it. Here we need to generalize the notion of an object assignment (Definition 2.26):

Definition 4.10 (Two-sorted Object Assignment). *A two-sorted object assignment (or just an object assignment)* σ *for a two-sorted structure* $\mathcal{M}$ *is a mapping from the number variables to* U_1 *together with a mapping from the string variables to* U_2*.*

Notation. We will write $\sigma(x)$ for the first-sort object assigned to the number variable x by σ, and $\sigma(X)$ for the second-sort object assigned to the string variable X by σ. Also as in the single-sorted case, if x is a variable and $m \in U_1$, then the object assignment $\sigma(m/x)$ is the same as σ except it maps x to m, and if X is a variable and $M \in U_2$, then the object assignment $\sigma(M/X)$ is the same as σ except it maps X to M.

Now the Basic Semantic Definition 2.27 generalizes in the obvious way.

Definition 4.11 (Basic Semantic Definition, Two-sorted Case). *Let* $\mathcal{L}$ *be a two-sorted first-order language, let* $\mathcal{M}$ *be an* $\mathcal{L}$*-structure with universe* $\langle U_1, U_2\rangle$*, and let* σ *be an object assignment for* $\mathcal{M}$*. Each* $\mathcal{L}$*-number term* t *is assigned an element* $t^{\mathcal{M}}[\sigma]$ *in* U_1*, and each* $\mathcal{L}$*-string term* T *is assigned an element* $T^{\mathcal{M}}[\sigma]$ *in* U_2*, defined by structural induction on terms* t *and* T*, as follows (refer to Definition 4.5 for the definition of* $\mathcal{L}$*-term):*

(a) $x^{\mathcal{M}}[\sigma]$ *is* $\sigma(x)$*, for each number variable* x
(b) $X^{\mathcal{M}}[\sigma]$ *is* $\sigma(X)$*, for each string variable* X
(c) $(ft_1 \cdots t_n T_1 \ldots T_m)^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma], T_1^{\mathcal{M}}[\sigma], \ldots, T_m^{\mathcal{M}}[\sigma])$
(d) $(Ft_1 \cdots t_n T_1 \ldots T_m)^{\mathcal{M}}[\sigma] = F^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma], T_1^{\mathcal{M}}[\sigma], \ldots, T_m^{\mathcal{M}}[\sigma])$

Definition 4.12. *For* φ *an* $\mathcal{L}$*-formula, the notion* $\mathcal{M} \models \varphi[\sigma]$ *(*$\mathcal{M}$ satisfies φ under σ*) is defined by structural induction on formulas* φ *as follows (refer to Definition 4.6 for the definition of a formula):*

(a) $\mathcal{M} \models \top$ *and* $\mathcal{M} \not\models \bot$
(b) $\mathcal{M} \models (Pt_1 \cdots t_n T_1 \ldots T_m)[\sigma]$ *iff*

$$\langle t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma], T_1^{\mathcal{M}}[\sigma], \ldots, T_m^{\mathcal{M}}[\sigma]\rangle \in P^{\mathcal{M}}$$

(c1) *If* $\mathcal{L}$ *contains* $=_1$*, then* $\mathcal{M} \models (s = t)[\sigma]$ *iff* $s^{\mathcal{M}}[\sigma] = t^{\mathcal{M}}[\sigma]$
(c2) *If* $\mathcal{L}$ *contains* $=_2$*, then* $\mathcal{M} \models (S = T)[\sigma]$ *iff* $S^{\mathcal{M}}[\sigma] = T^{\mathcal{M}}[\sigma]$

(d) $\mathcal{M} \models \neg\varphi[\sigma]$ *iff* $\mathcal{M} \not\models \varphi[\sigma]$.
(e) $\mathcal{M} \models (\varphi \vee \psi)[\sigma]$ *iff* $\mathcal{M} \models \varphi[\sigma]$ *or* $\mathcal{M} \models \psi[\sigma]$.
(f) $\mathcal{M} \models (\varphi \wedge \psi)[\sigma]$ *iff* $\mathcal{M} \models \varphi[\sigma]$ *and* $\mathcal{M} \models \psi[\sigma]$.
(g1) $\mathcal{M} \models (\forall x\varphi)[\sigma]$ *iff* $\mathcal{M} \models \varphi[\sigma(m/x)]$ *for all* $m \in U_1$
(g2) $\mathcal{M} \models (\forall X\varphi)[\sigma]$ *iff* $\mathcal{M} \models \varphi[\sigma(M/X)]$ *for all* $M \in U_2$
(h1) $\mathcal{M} \models (\exists x\varphi)[\sigma]$ *iff* $\mathcal{M} \models \varphi[\sigma(m/x)]$ *for some* $m \in U_1$
(h2) $\mathcal{M} \models (\exists X\varphi)[\sigma]$ *iff* $\mathcal{M} \models \varphi[\sigma(M/X)]$ *for some* $M \in U_2$

Note that items c1) and c2) in the definition of $\mathcal{M} \models A[\sigma]$ follow from b) and the fact that $=_1^{\mathcal{M}}$ and $=_2^{\mathcal{M}}$ are always the equality relations in the respective sorts.

The notions of "$\mathcal{M} \models \varphi$", "logical consequence", "validity", etc., are defined as before (Definition 2.29), and we do not repeat them here. Also, the Substitution Theorem (2.33) generalizes to the current context, and the Formula Replacement Theorem (2.34) continues to hold, and we will not restate them.

4C. Two-sorted Complexity Classes

4C.1. Notation for Numbers and Finite Sets. In Section 3D we explained how to interpret an element of a complexity class, such as **P** (polynomial time) and **LTH** (Linear Time Hierarchy) as a relation over $\mathbb{N}$. In this context the numerical inputs $x_1, \dots, x_k$ of a relation $R(x_1, \dots, x_k)$ are presented in binary to the accepting machine. In the two-sorted context, however, the relations $R(x_1, \dots, x_k, X_1, \dots, X_m)$ in question have arguments of both sorts, and now the numbers x_i are presented to the accepting machines using unary notation (n is represented by a string of n 1's) instead of binary. The elements X_i of the second sort are finite subsets of $\mathbb{N}$, and below we explain exactly how we represent them as binary strings for the purpose of presenting them as inputs to the accepting machine. The intuitive reason that we represent the numerical arguments in unary is that now they play an auxiliary role as indices to the string arguments, and hence their values are comparable in size to the length of the string arguments.

Thus a numerical relation $R(x)$ with no string argument is in two-sorted polynomial time iff it is computed in time $2^{O(n)}$ on some Turing machine, where n is the binary length of the input x. In particular, the relation $Prime(x)$ is easily seen to be in this class, using a "brute force" algorithm that tries all possible divisors between 1 and x.

The binary string representation of a finite subset of $\mathbb{N}$ is defined as follows. Recall that we write $S(i)$ for $i \in S$ (for $i \in \mathbb{N}$ and $S \subseteq \mathbb{N}$). Thus if we write 0 for $\perp$ and 1 for $\top$, then we can use the binary string

$$w(S) = S(n)S(n-1)\ \dots\ S(1)S(0) \tag{42}$$

to interpret the finite nonempty subset S of $\mathbb{N}$, where n is the largest member of S. We define $w(\varnothing)$ to be the empty string. For example,

$$w(\{0, 2, 3\}) = 1101$$

Thus w is an injective map from finite subsets of $\mathbb{N}$ to $\{0, 1\}^*$, but it is not surjective, since the string $w(S)$ begins with 1 for all nonempty S. Nevertheless $w(S)$ is a useful way to represent S as an input to a Turing machine or circuit.

Using the method just described of representing numbers and strings, we can define two-sorted complexity classes as sets of relations. For example two-sorted **P** consists of the set of all relations $R(\vec{x}, \vec{X})$ which are accepted in polynomial time by some deterministic Turing machine, where each numerical argument x_i is represented in unary as an input, and each subset argument X_i is represented by the string $w(X_i)$ as an input. Similar definitions specify the two-sorted polynomial hierarchy **PH**, and the two-sorted complexity classes $\mathbf{AC}^0$ and **LTH**.

4C.2. Representation Theorems. Notation. If $\vec{T} = T_1, \ldots T_n$, is a sequence of string terms, then $|\vec{T}|$ denotes the sequence $|T_1|, \ldots, |T_n|$ of number terms.

Bounded number quantifiers are defined as in the single-sorted case (Definition 3.6). To define bounded string quantifiers, we need the length function $|X|$ of $\mathcal{L}_A^2$.

Notation. A two-sorted language $\mathcal{L}$ is always assumed to be an extension of $\mathcal{L}_A^2$.

Definition 4.13 (Bounded Formulas). *Let $\mathcal{L}$ be a two-sorted language. If x is a number variable and X a string variable that do not occur in the $\mathcal{L}$-number term t, then $\exists x \le t\varphi$ stands for $\exists x(x \le t \wedge \varphi)$, $\forall x \le t\varphi$ stands for $\forall x(x \le t \supset \varphi)$, $\exists X \le t\varphi$ stands for $\exists X(|X| \le t \wedge \varphi)$, and $\forall X \le t\varphi$ stands for $\forall X(|X| \le t \supset \varphi)$. Quantifiers that occur in this form are said to be* bounded, *and a* bounded formula *is one in which every quantifier is bounded.*

Notation. $\exists \vec{x} \le \vec{t}\varphi$ stands for $\exists x_1 \le t_1 \ldots \exists x_k \le t_k\varphi$ for some k, where no x_i occurs in any t_j (even if $i < j$). Similarly for $\forall \vec{x} \le \vec{t}$, $\exists \vec{X} \le \vec{t}$, and $\forall \vec{X} \le \vec{t}$.

If the above convention is violated in the sense that x_i occurs in t_j for $i < j$, and the terms $\vec{t}$ are $\mathcal{L}_A^2$-terms, then new bounding terms $\overrightarrow{t'}$ in $\mathcal{L}_A^2$ can be found which satisfy the convention. For example $\exists x_1 \le t_1 \exists x_2 \le t_2(x_1)\varphi$ is equivalent to

$$\exists x_1 \le t_1 \exists x_2 \le t_2(t_1)(x_2 \le t_2(x_1) \wedge \varphi)$$

We will now define the following important classes of formulas.

Definition 4.14 (The $\Sigma_1^1(\mathcal{L})$, $\Sigma_i^B(\mathcal{L})$ and $\Pi_i^B(\mathcal{L})$ Formulas). ¿ ¿ *Let $\mathcal{L} \supseteq \mathcal{L}_A^2$ be a two-sorted language. Then $\Sigma_0^B(\mathcal{L}) = \Pi_0^B(\mathcal{L})$ is the set of $\mathcal{L}$-formulas whose only quantifiers are bounded number quantifiers (there can be free string variables). For $i \geq 0$, $\Sigma_{i+1}^B(\mathcal{L})$ (resp. $\Pi_{i+1}^B(\mathcal{L})$) is the set of formulas of the form $\exists \vec{X} \leq \vec{t}\varphi(\vec{X})$ (resp. $\forall \vec{X} \leq \vec{t}\varphi(\vec{X})$), where φ is a $\Pi_i^B(\mathcal{L})$ formula (resp. a $\Sigma_i^B(\mathcal{L})$ formula), and $\vec{t}$ is a sequence of $\mathcal{L}_A^2$-terms not involving any variable in $\vec{X}$. Also, a $\Sigma_1^1(\mathcal{L})$ formula is one of the form $\exists \vec{X}\varphi$, where $\vec{X}$ is a vector of zero or more string variables, and φ is a $\Sigma_0^B(\mathcal{L})$ formula.*

We will drop mention of $\mathcal{L}$ when it is clear from the context (the default is $\mathcal{L} = \mathcal{L}_A^2$). Thus

$$\Sigma_0^B \subseteq \Sigma_1^B \subseteq \Sigma_2^B \subseteq \cdots$$
$$\Sigma_0^B \subseteq \Pi_1^B \subseteq \Pi_2^B \subseteq \cdots$$

and for $i \geq 0$

$$\Sigma_i^B \subseteq \Pi_{i+1}^B \quad \text{and} \quad \Pi_i^B \subseteq \Sigma_{i+1}^B$$

Notice the "strict" requirements on $\Sigma_i^B(\mathcal{L})$ and $\Pi_i^B(\mathcal{L})$: all string quantifiers must occur in front. For example, $\Sigma_1^B(\mathcal{L}_A^2)$ is sometimes called *strict* $\Sigma_1^{1,b}$ in the literature. (Also notice that the bounding terms $\vec{t}$ must be in the basic language $\mathcal{L}_A^2$.) We will show that some theories prove *replacement* theorems, which assert the equivalence of a non-strict Σ_i^B formula (for certain values if i) with its strict counterpart.

In Section 3C.1 we discussed the definability of predicates (i.e., relations) and functions in a single-sorted theory. In the case of relations, the notion is purely semantic, and does not depend on the theory, but only the underlying language and the standard model. The situation is the same for the two-sorted case, and so we will define the notion of a relation $R(\vec{x}, \vec{X})$ *represented* by a formula, without reference to a theory. As in the single-sorted case, we assume that each relation symbol has a standard interpretation in an expansion of the standard model, in this case $\underline{\mathbb{N}}_2$, and formulas in the following definition are interpreted in the same model.

Definition 4.15 (Representable/Definable Relations). *Let $\mathcal{L} \supseteq \mathcal{L}_A^2$ be a two-sorted vocabulary, and let φ be an $\mathcal{L}$-formula. Then we say that $\varphi(\vec{x}, \vec{X})$* represents *(or* defines*) a relation $R(\vec{x}, \vec{X})$ if*

$$R(\vec{x}, \vec{X}) \leftrightarrow \varphi(\vec{x}, \vec{X}) \tag{43}$$

If Φ is a set of $\mathcal{L}$-formulas, then we say that $R(\vec{x}, \vec{X})$ is Φ-representable *(or* Φ-definable*) if it is represented by some $\varphi \in \Phi$.*

If we want to precisely represent a language $L \subseteq \{0,1\}^*$, then we need to consider strings that do not necessarily begin with 1. Thus the relation $R_L(X)$ corresponding to L is defined by

$$R_L(X) \leftrightarrow w'(X) \in L$$

where the string $w'(X)$ is obtained from $w(X)$ (42) by deleting the initial 1 (and $w'(\varnothing)$ and $w'(\{1\})$ both are the empty string).

Example 4.16. *The language PAL (page 73) of binary palindromes is represented by the formula*

$$\varphi_{PAL}(X) \leftrightarrow |X| \leq 1 \vee \forall x, y < |X|,\ x + y + 2 = |X| \supset (X(x) \leftrightarrow X(y))$$

Despite this example, we emphasize that the objects of the second sort in our complexity classes are finite sets of natural numbers, and we will not be much concerned by the fact that the corresponding strings (for nonempty sets) all begin with 1.

We define two-sorted $\mathbf{AC}^0$ using the log time hierarchy **LH**. We could define **LH** using alternating Turing machines (those relations accepted in log time with a constant number of alternations), but we choose instead to define the levels of the hierarchy using a recurrence analogous to our defintion of **LTH** in Section 3D.1. Thus we define **NLogTime** to be the class of relations $R(\vec{x}, \vec{X})$ accepted by a nondeterministic index Turing machine M in time $O(\log n)$.

As explained before, normally inputs $\vec{x}$ are presented in unary and $\vec{X}$ are presented in binary. However in defining **LH** it is convenient to change this convention and assume that the number inputs $\vec{x}$ are presented in binary (string inputs $\vec{X}$ are also presented in binary as before). To keep the meaning of "log time" unchanged, we define the length of a number input x_i to be x_i, even though the actual length of the binary notation is $|x_i|$. The reason for using binary notation is that in time $O(\log x_i)$ a Turing machine M can read the entire binary notation for x_i.

The machine M accesses its string inputs using index tapes; one such tape for each string argument X_i of $R(\vec{x}, \vec{X})$. When M enters the query state for an input X_i, if the index tape contains the number j written in binary, then j-th bit of X_i is returned. The index tape is not erased between input queries. Since M runs in log time, only $O(\log |X_i|)$ bits of X_i can be accessed during any one computation.

Now define

$$\Sigma_1^{log} = \mathbf{NLogTime} \tag{44}$$

and for $i \geq 1$

$$\Sigma_{i+1}^{log} = \mathbf{NLogTime}^{\Sigma_i^{log}} \tag{45}$$

Then

$$\mathbf{LH} = \bigcup_i \Sigma_i^{log}$$

Definition 4.17 (Two-sorted $\mathbf{AC}^0$).

$$\mathbf{AC}^0 = \mathbf{LH}$$

The notation $\mathbf{NLogTime}(\Sigma_i^{log})$ in (45) refers to a nondeterministic log time Turing machine M as above, except now M has access to an oracle for a relation $S(\vec{y}, \vec{Y})$ in Σ_i^{log}. In order to explain how M in log time accesses an arbitrary input $(\vec{y}, \vec{Y})$ to S, we simplify things by requiring that $\vec{X} = \vec{Y}$; that is the string inputs to S are the same as the string inputs to M. However the number inputs $\vec{y}$ to S are arbitrary: M has time to write them in binary on a special query tape.

Two-sorted $\mathbf{AC}^0$ restricted to numerical relations $R(\vec{x})$ is exactly the same as single-sorted $\mathbf{LTH}$ as defined in Section 3D.1. The amount of time alloted for the Turing machines under the two definitions for an input $\vec{x}$ is the same, namely $O(\log(\Sigma x_i))$.

Thus for numerical relations, the following representation theorem is the same as the $\mathbf{LTH}$ Theorem 3.58 ($\mathbf{LTH} = \Delta_0^{\mathbb{N}}$). For string relations, it can be considered a restatement of Theorem 4.2 ($\mathbf{FO} = \mathbf{AC}^0$).

Theorem 4.18 (Σ_0^B Representation Theorem). *A relation $R(\vec{x}, \vec{X})$ is in $\mathbf{AC}^0$ iff it is represented by some Σ_0^B formula $\varphi(\vec{x}, \vec{X})$.*

Proof sketch. In light of the above discussion, the proof is essentially the same as for Theorem 3.58. To show that every relation $R(\vec{x}, \vec{X})$ in $\mathbf{AC}^0$ (i.e. $\mathbf{LH}$) is representable by a Σ_0^B formula $\varphi(\vec{x}, \vec{X})$ we use the recurrence (44), (45). The proof is almost the same as showing $\mathbf{LTH} \subseteq \Delta_0^{\mathbb{N}}$. There is an extra consideration in the base case, showing how the formula $\varphi(\vec{x}, \vec{X})$ represents the computation of a log time nondeterministic Turing machine M that now accesses its string inputs using index tapes. The computation is represented as before, except now $\varphi(\vec{x}, \vec{X})$ uses an extra number variable j_i for each string input variable X_i. Here j_i holds the current numerical value of the index tape for X_i.

The proof of the converse, that every relation representable by a Σ_0^B formula is in $\mathbf{LH}$, is straightforward and similar to the proof that $\Delta_0^{\mathbb{N}} \subseteq \mathbf{LTH}$. $\dashv$

Notation. For X a finite subset of $\mathbb{N}$, let $bin(X)$ be the number whose binary notation is $w(X)$ (see (42)). Thus

$$bin(X) = \sum_i X(i)2^i \tag{46}$$

where here we treat the predicate $X(i)$ as a 0–1-valued function. For example, $bin(\{0, 2, 3\}) = 2^2 + 2^3 = 12$.

Define the relations R_+ and $R_\times$ by

$$R_+(X, Y, Z) \leftrightarrow bin(X) + bin(Y) = bin(Z)$$
$$R_\times(X, Y, Z) \leftrightarrow bin(X) \cdot bin(Y) = bin(Z)$$

As mentioned earlier, $PARITY$ is efficiently reducible to $R_\times$, and hence $R_\times$ is not in $\mathbf{AC}^0$, and cannot be represented by any Σ_0^B formula. However R_+ is in $\mathbf{AC}^0$. To represent it as a Σ_0^B formula, we first define the relation $Carry(i, X, Y)$ to mean that there is a carry into bit position i when computing $bin(X) + bin(Y)$. Then (using the idea behind a carry-lookahead adder)

(47)
$$Carry(i, X, Y) \leftrightarrow \exists k < i\big(X(k) \wedge Y(k) \wedge \forall j < i(k < j \supset (X(j) \vee Y(j)))\big)$$

Thus

$$\begin{aligned} R_+(X, Y, Z) \leftrightarrow \big(&|Z| \le |X| + |Y| \wedge \\ &\forall i < |X| + |Y|(Z(i) \leftrightarrow (X(i) \oplus Y(i) \oplus Carry(i, X, Y)))\big) \end{aligned}$$

where $\oplus$ represents exclusive or.

Note that the Σ_0^B Representation Theorem can be alternatively proved by using the characterization $\mathbf{AC}^0 = \mathbf{FO}$. Here we need the fact that

$$\mathbf{FO}[BIT] = \mathbf{FO}[PLUS, TIMES]$$

i.e., the vocabulary $\mathcal{L}_{FO}$ in (41) can be equivalently defined as

$$[0, max, +, \ \cdot; X, \le, =]$$

Note also that in $\mathcal{L}_{FO}$ we have only one "free" unary predicate symbol X, so technically speaking, $\mathcal{L}_{FO}$ formulas can describe only unary relations (i.e., languages). In order to describe a k-ary relation, one way is to extend the vocabulary $\mathcal{L}_{FO}$ to include additional "free" unary predicates. Then Theorem 4.2 continues to hold. Now the Σ_0^B Representation Theorem can be proved by translating any Σ_0^B formula φ into an $\mathbf{FO}$ formula φ' that *describes* the relation *represented* by φ, and vice versa.

We use Σ_i^p to denote level $i \ge 1$ of the two-sorted polynomial hierarchy. In particular, Σ_1^P denotes two-sorted $\mathbf{NP}$. Thus a relation $R(\vec{x}, \vec{X})$ is in Σ_i^p iff it is accepted by some polynomial time ATM with at most i alternations, starting with existential, using the input conventions described in Section 4C.1.

Theorem 4.19 (Σ_i^B and Σ_1^1 Representation Theorem). *For $i \ge 1$, a relation $R(\vec{x}, \vec{X})$ is in Σ_i^p iff it is represented by some Σ_i^B formula. The relation is recursively enumerable iff it is represented by some Σ_1^1 formula.*

Proof. We show that a relation $R(\vec{x}, \vec{X})$ is in $\mathbf{NP}$ iff it is represented by a Σ_1^B formula. (The other cases are proved similarly.) First suppose that $R(\vec{x}, \vec{X})$ is accepted by a nondeterministic polytime Turing machine M. Then the Σ_1^B formula that represents R has the form

$$\exists Y \le t(\vec{x}, \vec{X})\ \varphi(\vec{x}, \vec{X}, Y)$$

where Y codes an accepting computation of M on input $\langle\vec{x},\vec{X}\rangle$, t represents the upper bound on the length of such computation, and φ is a Σ_0^B formula that verifies the correctness of Y. Here the bounding term t exists by the assumption that M works in polynomial time, and the formula φ can be easily constructed given the transition function of M.

On the other hand, suppose that $R(\vec{x},\vec{X})$ is represented by the Σ_1^B formula

$$\exists\vec{Y}\leq\vec{t}(\vec{x},\vec{X})\ \varphi(\vec{x},\vec{X},\vec{Y})$$

Then the polytime NTM M that accepts R works as follows. On input $\langle\vec{x},\vec{X}\rangle$ M simply guesses the values of $\vec{Y}$, and then verifies that $\varphi(\vec{x},\vec{X},\vec{Y})$ holds. The verification can be easily done in polytime (it is in fact in $\mathbf{AC}^0$ as shown by the Σ_0^B Representation Theorem). ⊣

4C.3. The LTH Revisited. Consider **LTH** (Linear Time Hierarchy, Section 3D) as a two-sorted complexity class. Here we can define the relations in this class by *linearly bounded formulas*, a concept defined below.

Definition 4.20. *A formula φ over $\mathcal{L}_A^2$ is called a* linearly bounded formula *if all of its quantifiers are bounded by terms not involving $\cdot$.*

Theorem 4.21 (Two-sorted **LTH** Theorem). *A relation is in* **LTH** *if and only if it is represented by some linearly bounded formula.*

The proof of this theorem is similar to the proof of Theorem 3.58. Here the ($\Longleftarrow$) direction is simpler: For the base case, we need to calculate the number terms $t(x_1,\ldots,x_k,|X_1|,\ldots,|X_m|)$ in time *linear in* $(\sum x_i+\sum|X_j|)$, and this is straightforward.

For the other direction, as in the proof of the single-sorted **LTH** Theorem, the interesting part is to show that relations in **NLinTime** can be represented by linearly bounded formulas. Here we do not need to define the relation $y=2^x$ as in the single-sorted case, since the relation $X(i)$ (which stands for $i\in X$) is already in our vocabulary. We still need to "count" the number of 1-bits in a string, i.e., we need to define the two-sorted version of *Numones*: $\mathit{Numones}_2(a,i,X)$ is true iff a is the number of 1-bits in the first i low-order bits of X. Again, $\mathit{Numones}_2$ can be defined using Bennett's Trick.

Exercise 4.22. (a) *Define using linearly bounded formula the relation $m=\lceil\sqrt{i}\rceil$.*
(b) *Define using linearly bounded formula the relation "k = the number of 1-bits in the substring $X(im)\ \ldots\ X(im+m-1)$".*
(c) *Now define $\mathit{Numones}_2(a,i,X)$ using linearly bounded formula.*

Exercise 4.23. *Complete the proof of the Two-Sorted* **LTH** *Theorem.*

In [86], Zambella considers the subset of $\mathcal{L}_A^2$ without the number function $\cdot$, denoted here by $\mathcal{L}_A^{2-}$, and introduces the notion of *linear* *formulas*, which are the bounded formulas in the language $\mathcal{L}_A^{2-}$. Then **LTH** is also characterized as the class of relations representable by linear formulas. In order to prove this claim from the Two-Sorted **LTH** Theorem above, we need to show that the relation $x \cdot y = z$ is definable by some *linear formula*.

Exercise 4.24. *Define the relation $x \cdot y = z$ using a linear formula. (Hint: First define the relation "z is a multiple of y".)*

We have shown how to define the relation $y = 2^x$ using Δ_0 formula in Section 3C.3. Here it is much easier to define this relation using linearly bounded formulas.

Exercise 4.25. *Show how to express $y = 2^x$ using linearly bounded formula. (Hint: Use $Numones_2$ from Exercise 4.22.)*

4D. The Proof System $\mathbf{LK}^2$

Now we extend the sequent system **LK** (Section 2B.3) to a system $\mathbf{LK}^2$ for a two-sorted language $\mathcal{L}^2$. As for **LK**, here we introduce the *free string variables* denoted by $\alpha, \beta, \gamma, \ldots$, and the *bound string variables* $X, Y, Z, \ldots$ in addition to the *free number variables* denoted by $a, b, c, \ldots$, and the *bound number variables* denoted by $x, y, z, \ldots$.

Also, in $\mathbf{LK}^2$ the terms (of both sorts) do not involve any bound variable, and the formulas do not have any free occurrence of any bound variable.

The system $\mathbf{LK}^2$ includes all axioms and rules for **LK** as described in Section 2B.3, where the term t is a number term respecting our convention for free and bound variables above. In addition $\mathbf{LK}^2$ has the following four rules introducing string quantifiers, here T is any string term that does not contain any bound string variable $X, Y, Z, \ldots$:

String $\forall$ introduction rules

$$\text{left: } \frac{\varphi(T), \Gamma \longrightarrow \Delta}{\forall X \varphi(X), \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, \varphi(\beta)}{\Gamma \longrightarrow \Delta, \forall X \varphi(X)}$$

String $\exists$ introduction rules

$$\text{left: } \frac{\varphi(\beta), \Gamma \longrightarrow \Delta}{\exists X \varphi(X), \Gamma \longrightarrow \Delta} \qquad \text{right: } \frac{\Gamma \longrightarrow \Delta, \varphi(T)}{\Gamma \longrightarrow \Delta, \exists X \varphi(X)}$$

Restriction. The free variable β must not occur in the conclusion of $\forall$-right and $\exists$-left.

The notions of $\mathbf{LK}^2$ *proofs*generalize the notion of **LK** proofs and anchored **LK** proofs. Then the Derivational Soundness, the Completeness Theorem (2.42), and the Anchored Completeness Theorem (2.47) continue to hold for $\mathbf{LK}^2$ (without equality).

In general, when the vocabulary $\mathcal{L}$ does not contain either of the equality predicate symbols, then the notion of $\mathbf{LK}^2$-Φ proof is defined as in Definition 2.40. In the sequel our two-sorted vocabularies will all contain both of the equality predicates, so we will restrict our attention to this case. Here we need to generalize the Equality Axioms given in Definition 2.54. Recall that we write $=$ for both $=_1$ and $=_2$.

Definition 4.26 ($\mathbf{LK}^2$ Equality Axioms for $\mathcal{L}$). *Suppose that $\mathcal{L}$ is a two-sorted vocabulary containing both $=_1$ and $=_2$. The* $\mathbf{LK}^2$ Equality Axioms for $\mathcal{L}$ *consists of the following axioms. (We let Λ stand for*

$$t_1 = u_1, \dots, t_n = u_n, T_1 = U_1, \dots, T_m = U_m$$

in $\mathbf{E}4'$, $\mathbf{E}4''$ and $\mathbf{E}5'$.) Here t, u, t_i, u_i are number terms, and T, U, T_i, U_i are string terms.
$\mathbf{E}1'$. $\longrightarrow t = t$
$\mathbf{E}1''$. $\longrightarrow T = T$
$\mathbf{E}2'$. $t = u \longrightarrow u = t$
$\mathbf{E}2''$. $T = U \longrightarrow U = T$
$\mathbf{E}3'$. $t = u, u = v \longrightarrow t = v$
$\mathbf{E}3''$. $T = U, U = V \longrightarrow T = V$
$\mathbf{E}4'$. $\Lambda \longrightarrow ft_1 \dots t_n T_1 \dots T_m = fu_1 \dots u_n U_1 \dots U_m$ *for each f in $\mathcal{L}$*
$\mathbf{E}4''$. $\Lambda \longrightarrow Ft_1 \dots t_n T_1 \dots T_m = Fu_1 \dots u_n U_1 \dots U_m$ *for each F in $\mathcal{L}$*
$\mathbf{E}5'$. $\Lambda, Pt_1 \dots t_n T_1 \dots T_m \longrightarrow Pu_1 \dots u_n U_1 \dots U_m$ *for each P in $\mathcal{L}$ (here P is not $=_1$ or $=_2$).*

Definition 4.27 ($\mathbf{LK}^2$-Φ Proofs). *Suppose that $\mathcal{L}$ is a two-sorted vocabulary containing both $=_1$ and $=_2$, and Φ is a set of $\mathcal{L}$-formulas. Then an $\mathbf{LK}^2$-Φ proof (or a Φ-proof) is an $\mathbf{LK}^2$-Ψ proof in the sense of Definition 2.40, where Ψ is Φ together with all instances of the $\mathbf{LK}^2$ Equality Axioms $\mathbf{E}1'$, $\mathbf{E}1''$, ..., $\mathbf{E}4'$, $\mathbf{E}4''$, $\mathbf{E}5'$ for $\mathcal{L}$. If Φ is empty, we simply refer to an $\mathbf{LK}^2$-proof (but allow $\mathbf{E}1'$, ..., $\mathbf{E}5'$ as axioms).*

Recall that if φ is a formula with free variables $a_1, \dots, a_n, \alpha_1, \dots, \alpha_m$, then $\forall\varphi$, the universal closure of φ, is the sentence

$$\forall x_1 \dots \forall x_n \forall X_1 \dots \forall X_m \varphi(x_1/a_1, \dots, x_n/a_n, X_1/\alpha_1, \dots, X_m/\alpha_m)$$

where $x_1, \dots, x_n, X_1, \dots, X_m$ is a list of new bound variables. Also recall that if Φ is a set of formulas, then $\forall\Phi$ is the set of all sentences $\forall\varphi$, for $\varphi \in \Phi$.

The following Soundness and Completeness Theorem for the two-sorted system $\mathbf{LK}^2$ is the analogue of Theorem 2.56, and is proved in the same way.

Theorem 4.28 (Soundness and Completeness of $\mathbf{LK}^2$). *For any set Φ of formulas and sequent S,*

$$\forall\Phi \models S \text{ iff } S \text{ has an } \mathbf{LK}^2\text{-}\Phi \text{ proof}$$

Below we will state the two-sorted analogue of the Anchored **LK** Completeness Theorem and the Subformula Property of Anchored **LK** Proofs (Theorems 2.58 and 2.59). They can be proved just as in the case of **LK**.

Definition 4.29 (Anchored $\mathbf{LK}^2$ Proof). *An* $\mathbf{LK}^2$-Φ *proof* π *is* anchored *provided every cut formula in* π *is a formula in some non-logical axiom of* π *(including possibly* $\mathbf{E}1'$, $\mathbf{E}1''$, ... , $\mathbf{E}5'$*).*

Theorem 4.30 (Anchored $\mathbf{LK}^2$ Completeness). *Suppose that* Φ *is a set of formulas closed under substitution of terms for variables and that the sequent* S *is a logical consequence of* $\forall\Phi$*. Then there is an anchored* $\mathbf{LK}^2$-Φ *proof of* S*.*

Theorem 4.31 (Subformula Property of Anchored $\mathbf{LK}^2$ Proofs). *If* π *is an anchored* $\mathbf{LK}^2$-Φ *proof of a sequent* S*, then every formula in every sequent of* π *is a term substitution instance of a sub-formula of a formula either in* S *or in a non-logical axiom of* π *(including* $\mathbf{E}1'$, ... , $\mathbf{E}4''$, $\mathbf{E}5'$*).*

As in the case for **LK** where the Anchored **LK** Completeness Theorem is used to prove the Compactness Theorem (Theorem 2.61), the above Anchored $\mathbf{LK}^2$ Completeness Theorem can be used to prove the following (two-sorted) Compactness Theorem.

Theorem 4.32 (Compactness Theorem). *If* Φ *is an unsatisfiable set of (two-sorted) formulas, then some finite subset of* Φ *is unsatisfiable.*

(See also the three alternative forms in Theorem 2.16.)

Form 1 of the Herbrand Theorem (Theorem 2.67) can also be extended to the two-sorted logic, with the set of (single-sorted) equality axioms $\mathcal{E}_{\mathcal{L}}$ now replaced by the set of two-sorted equality axioms $\mathbf{E}1'$, $\mathbf{E}1''$, ..., $\mathbf{E}4''$, $\mathbf{E}5'$ above. Below we will state only Form 2 of the Herbrand Theorem for the two-sorted logics. Note that it also follows from Form 1, just as in the single-sorted case.

A *two-sorted theory* (or just *theory*, when it is clear) is defined as in Definition 3.1, where now it is understood that the underlying language $\mathcal{L}$ is a two-sorted language. Also, a universal theory is a theory which can be axiomatized by universal formulas, (i.e., formulas in prenex form, in which all quantifiers are universal).

Theorem 4.33 (Herbrand Theorem for Two-sorted Logic). (a) *Let* $\mathcal{T}$ *be a universal (two-sorted) theory, and let* $\varphi(x_1, \dots, x_k, X_1, \dots, X_m, Z)$ *be a quantifier-free formula with all free variables displayed such that*

$$\mathcal{T} \vdash \forall x_1 \dots \forall x_k \forall X_1 \dots \forall X_m \exists Z \varphi(\vec{x}, \vec{X}, Z).$$

Then there exist finitely many string terms $T_1(\vec{x}, \vec{X}), \dots, T_n(\vec{x}, \vec{X})$ *such that*

$$\mathcal{T} \vdash \forall x_1 \dots \forall x_k \forall X_1 \dots \forall X_m \big(\varphi(\vec{x}, \vec{X}, T_1(\vec{x}, \vec{X})) \vee \dots \vee \varphi(\vec{x}, \vec{X}, T_n(\vec{x}, \vec{X}))\big)$$

(b) *Similarly, let the theory* $\mathcal{T}$ *be as above, and let*

$$\varphi(x_1, \ldots, x_k, z, X_1, \ldots, X_m)$$

be a quantifier-free formula with all free variables displayed such that

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_k \forall X_1 \ldots \forall X_m \exists z \varphi(\vec{x}, z, \vec{X}).$$

Then there exist finitely many number terms $t_1(\vec{x}, \vec{X}), \ldots, t_n(\vec{x}, \vec{X})$ *such that*

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_k \forall X_1 \ldots \forall X_m \left(\varphi(\vec{x}, t_1(\vec{x}, \vec{X}), \vec{X}) \vee \cdots \vee \varphi(\vec{x}, t_n(\vec{x}, \vec{X}), \vec{X})\right)$$

The theorem easily extends to the cases where

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists z_1 \ldots \exists z_m \exists Z_1 \ldots \exists Z_n \varphi(\vec{x}, \vec{z}, \vec{X}, \vec{Z}).$$

4D.1. Two-Sorted Free Variable Normal Form. The notion of free variable normal form (Section 2B.4) generalizes naturally to $\mathbf{LK}^2$ proofs, where now the term *free variable* refers to free variables of both sorts. Again there is a simple procedure for putting any $\mathbf{LK}^2$ proof into free variable normal form (with the same endsequent), provided that the underlying language has constant symbols of both sorts. This procedure preserves the size and shape of the proof, and takes an anchored $\mathbf{LK}^2$-Φ proof to an anchored $\mathbf{LK}^2$-Φ proof, provided that the set Φ of formulas is closed under substitution of terms for free variables.

In the case of $\mathcal{L}^2_A$, there is no string constant symbol, so we expand the notion of a $\mathbf{LK}^2$-Φ proof over $\mathcal{L}^2_A$ by allowing the constant symbol $\varnothing$ (for the empty string) and assume that Φ contains the following axiom:

E. $|\varnothing| = 0$

Adding this symbol and axiom to any theory $\mathcal{T}$ over $\mathcal{L}^2_A$ we consider will result in a conservative extension of $\mathcal{T}$, since every model for $\mathcal{T}$ can trivially be expanded to a model of $\mathcal{T} \cup \{\mathbf{E}\}$. Now any $\mathbf{LK}^2$ proof over $\mathcal{L}^2_A$ can be transformed to one in free variable normal form with the same endsequent, and similarly for $\mathbf{LK}^2$-Φ for suitable Φ.

4E. Single-Sorted Logic Interpretation

In this section we will briefly discuss how the Compactness Theorem and Herbrand Theorem in the two-sorted logic follow from the analogous results for the single-sorted logic that we have seen in Chapter 2. This section is independent with the rest of the book, and it is the approach that we follow to prove the above theorems in Section 4D that will be useful in later chapters, not the approach that we present here.

Although a two-sorted logic is a generalization of a single-sorted logic by having one more sort, it can be interpreted as a single-sorted logic by merging both sorts and using 2 extra unary predicate symbols to identify elements of the 2 sorts.

More precisely, for each two-sorted vocabulary $\mathcal{L}$, w.l.o.g., we can assume that it does not contain the unary predicate symbols FS (for first sort) and SS (for second sort). Let $\mathcal{L}^1 = \{\mathsf{FS}, \mathsf{SS}\} \cup \mathcal{L}$, where it is understood that the functions and predicates in $\mathcal{L}_1$ take arguments from a single sort.

In addition, let $\Phi_{\mathcal{L}}$ be the set of $\mathcal{L}^1$-formulas which consists of

1) $\forall x,\ \mathsf{FS}(x) \vee \mathsf{SS}(x)$.
2) For each function symbol f of $\mathcal{L}^1$ (where f has arity (n, m) in $\mathcal{L}$) the formula

$$\forall \vec{x} \forall \vec{y},\ (\mathsf{FS}(x_1) \wedge \ldots \mathsf{FS}(x_n) \wedge \mathsf{SS}(y_1) \cdots \wedge \mathsf{SS}(y_m)) \supset \mathsf{FS}(f(\vec{x}, \vec{y}))$$

   (If f is a number constant c, the above formula is just $\mathsf{FS}(c)$.)
3) For each function symbol F of $\mathcal{L}^1$ (where F has arity (n, m) in $\mathcal{L}$) the formula

$$\forall \vec{x} \forall \vec{y},\ (\mathsf{FS}(x_1) \wedge \ldots \mathsf{FS}(x_n) \wedge \mathsf{SS}(y_1) \cdots \wedge \mathsf{SS}(y_m)) \supset \mathsf{SS}(F(\vec{x}, \vec{y}))$$

   (If F is a string constant α, the above formula is just $\mathsf{SS}(\alpha)$.)
4) For each predicate symbol P of $\mathcal{L}^1$ (where P has arity (n, m) in $\mathcal{L}$) the formula

$$\forall \vec{x} \forall \vec{y},\ P(\vec{x}, \vec{y}) \supset (\mathsf{FS}(x_1) \wedge \ldots \mathsf{FS}(x_n) \wedge \mathsf{SS}(y_1) \cdots \wedge \mathsf{SS}(y_m))$$

Lemma 4.34. *For each nonempty two-sorted language $\mathcal{L}$, the set $\Phi_{\mathcal{L}}$ is satisfiable.*

Proof. The proof is straightforward: For an arbitrary (two-sorted) $\mathcal{L}$-structure $\mathcal{M}$ with universe $\langle U_1, U_2 \rangle$, we construct a (single-sorted) $\mathcal{L}_1$-structure $\mathcal{M}_1$ that has universe $\langle U_1, U_2 \rangle$, $\mathsf{FS}^{\mathcal{M}_1} = U_1, \mathsf{SS}^{\mathcal{M}_1} = U_2$, and the same interpretation as in $\mathcal{M}$ for each symbol of $\mathcal{L}$. It is easy to verify that $\mathcal{M}_1 \models \Phi_{\mathcal{L}}$. ⊣

It is also evident from the above proof that any model $\mathcal{M}_1$ of $\Phi_{\mathcal{L}}$ can be interpreted as a two-sorted $\mathcal{L}$-structure $\mathcal{M}$.

Now we construct for each $\mathcal{L}$-formula φ an $\mathcal{L}^1$-formula φ^1 inductively as follows.

1) If φ is an atomic sentence, then $\varphi^1 =_{\text{def}} \varphi$.
2) If $\varphi \equiv \varphi_1 \wedge \varphi_2$ (or $\varphi \equiv \varphi_1 \vee \varphi_2$, or $\varphi \equiv \neg\psi$), then $\varphi^1 =_{\text{def}} \varphi_1^1 \wedge \varphi_2^1$ (or $\varphi^1 \equiv \varphi_1^1 \vee \varphi_2^1$, or $\varphi^1 \equiv \neg\psi^1$, respectively).
3) If $\varphi \equiv \exists x \psi(x)$, then $\varphi^1 =_{\text{def}} \exists x(\mathsf{FS}(x) \wedge \psi^1(x))$.
4) If $\varphi \equiv \forall x \psi(x)$, then $\varphi^1 =_{\text{def}} \forall x(\mathsf{FS}(x) \supset \psi^1(x))$.
5) If $\varphi \equiv \exists X \psi(X)$, then $\varphi^1 =_{\text{def}} \exists x(\mathsf{SS}(x) \wedge \psi^1(x))$.
6) If $\varphi \equiv \forall X \psi(X)$, then $\varphi^1 =_{\text{def}} \forall x(\mathsf{SS}(x) \supset \psi^1(x))$.

Note that when φ is a sentence, then φ^1 is also a sentence.

For a set Ψ of $\mathcal{L}$-formulas, let Ψ^1 denote the set $\{\varphi^1 : \varphi \in \Psi\}$. The lemma above can strengthened as follows.

Theorem 4.35. *A set Ψ of $\mathcal{L}$-sentences φ is satisfiable iff the set of $\Phi_{\mathcal{L}} \cup \Psi^1$ of $\mathcal{L}^1$-sentences is satisfiable.*

Notice that in the statement of the theorem, Ψ is a set of *sentences*. In general, the theorem may not be true if Ψ is a set of formulas.

Proof. For simplicity, we will prove the theorem when Ψ is the set of a single sentence φ. The proof for the general case is similar.

For the ONLY IF direction, for any model $\mathcal{M}$ of φ we construct a $\mathcal{L}_1$-structure $\mathcal{M}_1$ as in the proof of Lemma 4.34. It can be proved by structural induction on φ that $\mathcal{M}_1 \models \varphi^1$. By the lemma, $\mathcal{M}_1 \models \Phi_{\mathcal{L}}$. Hence $\mathcal{M}_1 \models \Phi_{\mathcal{L}} \cup \{\varphi^1\}$.

For the other direction, suppose that $\mathcal{M}_1$ is a model for $\Phi_{\mathcal{L}} \cup \{\varphi^1\}$. Construct the two-sorted $\mathcal{L}$-structure $\mathcal{M}$ from $\mathcal{M}_1$ as in the remark following the proof of Lemma 4.34. Now we can prove by structural induction on φ that $\mathcal{M}$ is a model for φ. Therefore φ is also satisfiable. $\dashv$

Exercise 4.36. *Prove the Compactness Theorem for the two-sorted logic (4.32) from the Compactness Theorem for single-sorted logic (2.61).*

Exercise 4.37. *Prove the Herbrand Theorem for the two-sorted logic (4.33) from Form 2 of the Herbrand Theorem for single-sorted logic (3.38).*

4F. Notes

Historically, Buss [12] was the first to use multi-sorted theories to capture complexity classes such as polynomial space and exponential time.

The main reference for Section 4A is [45] Sections 1.1, 1.2, 5.5. Our two-sorted language $\mathcal{L}_A^2$ is from Zambella [85, 86]. Zambella [85] states the representation theorems 4.18 and 4.19, although Theorem 4.19 essentially goes back to [84], [39], and [79].

Chapter 5

THE THEORY $\mathbf{V}^0$ AND $\mathbf{AC}^0$

In this chapter we introduce the family of two-sorted theories $\mathbf{V}^0 \subset \mathbf{V}^1 \subseteq \mathbf{V}^2 \subseteq \cdots$. For $i \geq 1$, $\mathbf{V}^i$ corresponds to Buss's single-sorted theory $\mathbf{S}_2^i$ (Section 3E). The theory $\mathbf{V}^0$ characterizes $\mathbf{AC}^0$ in the same way that $\mathbf{I\Delta}_0$ characterizes **LTH**. Similarly $\mathbf{V}^1$ characterizes $\mathbf{P}$, and in general for $i > 1$, $\mathbf{V}^i$ is related to the i-th level of the polynomial time hierarchy.

Here we concentrate on the theory $\mathbf{V}^0$, which will serve as the base theory: all two-sorted theories introduced in this book are extensions of $\mathbf{V}^0$. It is axiomatized by the set 2-**BASIC** of the defining axioms for the symbols in $\mathcal{L}_A^2$, together with Σ_0^B-**COMP** (the comprehension axiom scheme for Σ_0^B formulas). For $i \geq 1$, $\mathbf{V}^i$ is the same as $\mathbf{V}^0$ except that Σ_0^B-**COMP** is replaced by Σ_i^B-**COMP**. We generalize Parikh's Theorem, and show that it applies to each of the theories $\mathbf{V}^i$.

The main result of this chapter is that $\mathbf{V}^0$ characterizes $\mathbf{AC}^0$: The provably total functions in $\mathbf{V}^0$ are precisely the $\mathbf{AC}^0$ functions. The proof of this characterization is somewhat more involved than the proof of the analogous characterization of **LTH** by $\mathbf{I\Delta}_0$ (Theorem 3.64). The hard part here is the *Witnessing Theorem for* $\mathbf{V}^0$, which is proved by analyzing anchored $\mathbf{LK}^2$-$\mathbf{V}^0$ proofs. We also give an alternative proof of the witnessing theorem based on the universal conservative extension $\overline{\mathbf{V}}^0$ of $\mathbf{V}^0$, using the Herbrand Theorem.

5A. Definition and Basic Properties of $\mathbf{V}^i$

The set 2-**BASIC** of axioms is given in Figure 2. Recall that $t < u$ stands for $(t \leq u \wedge t \neq u)$.

Axioms **B**1, ... , **B**8 are taken from the axioms in 1-**BASIC** for $\mathbf{I\Delta}_0$, and **B**9, ... , **B**12 are theorems of $\mathbf{I\Delta}_0$ (see Examples 3.8 and 3.9). Axioms **L**1 and **L**2 characterize $|X|$ to be one more than the largest element of X, or 0 if X is empty. Axiom **SE** (extensionality) specifies that sets X and Y are the same if they have the same elements. Note that the converse

$$X = Y \supset (|X| = |Y| \wedge \forall i < |X|(X(i) \leftrightarrow Y(i)))$$

B1. $x + 1 \neq 0$	**B**7. $(x \leq y \wedge y \leq x) \supset x = y$
B2. $x + 1 = y + 1 \supset x = y$	**B**8. $x \leq x + y$
B3. $x + 0 = x$	**B**9. $0 \leq x$
B4. $x + (y + 1) = (x + y) + 1$	**B**10. $x \leq y \vee y \leq x$
B5. $x \cdot 0 = 0$	**B**11. $x \leq y \leftrightarrow x < y + 1$
B6. $x \cdot (y + 1) = (x \cdot y) + x$	**B**12. $x \neq 0 \supset \exists y \leq x(y + 1 = x)$
L1. $X(y) \supset y < \lvert X\rvert$	**L**2. $y + 1 = \lvert X\rvert \supset X(y)$
SE. $\big(\lvert X\rvert = \lvert Y\rvert \wedge \forall i < \lvert X\rvert(X(i) \leftrightarrow Y(i))\big) \supset X = Y$	

FIGURE 2. 2-**BASIC**

is valid because in every $\mathcal{L}_A^2$-structure, $=_2$ must be interpreted as true equality over the strings.

EXERCISE 5.1. *Show that the following formulas are provable from* 2-**BASIC**.

(a) $\neg x < 0$.
(b) $x < x + 1$.
(c) $0 < x + 1$.
(d) $x < y \supset x + 1 \leq y$. *(Use* **B**10*,* **B**11*,* **B**7*.)*
(e) $x < y \supset x + 1 < y + 1$.

DEFINITION 5.2 (Comprehension Axiom). *If Φ is a set of formulas, then the* comprehension axiom scheme *for Φ, denoted by Φ-***COMP**, *is the set of all formulas*

$$\exists X \leq y \forall z < y(X(z) \leftrightarrow \varphi(z)), \tag{48}$$

where $\varphi(z)$ is any formula in Φ, and X does not occur free in $\varphi(z)$.

In the above definition $\varphi(z)$ may have free variables of both sorts, in addition to z. We are mainly interested in the cases in which Φ is one of the formula classes Σ_i^B.

Notation. Since (48) states the existence of a finite set X of numbers, we will sometimes use standard set-theoretic notation in defining X:

$$X = \{z \mid z < y \wedge \varphi(z)\} \tag{49}$$

DEFINITION 5.3 ($\mathbf{V}^i$). *For $i \geq 0$, the theory $\mathbf{V}^i$ has the vocabulary $\mathcal{L}_A^2$ and is axiomatized by* 2-**BASIC** *and* Σ_i^B-**COMP**.

Notation. Since now there are two sorts of variables, there are two different types of *induction axioms*: One is on numbers, and is defined as in Definition 3.4 (where now Φ is a set of two-sorted formulas), and one is on strings, which we will discuss later. For this reason, we will speak of *number induction axioms* and *string induction axioms*. Similarly, we will use the notion of *number minimization axioms*, which is different from the *string minimization axioms* (to be introduced later).

For convenience we repeat the definitions of the axiom schemes for numbers below.

DEFINITION 5.4 (Number Induction Axiom). *If Φ is a set of two-sorted formulas, then Φ-**IND** axioms are the formulas*

$$\big(\varphi(0) \wedge \forall x(\varphi(x) \supset \varphi(x+1))\big) \supset \forall z \varphi(z)$$

where φ is a formula in Φ.

DEFINITION 5.5 (Number Minimization and Maximization Axioms). *The number minimization axioms (or* least number principle *axioms) for a set Φ of two-sorted formulas are denoted Φ-**MIN** and consist of the formulas*

$$\varphi(y) \supset \exists x \leq y\big(\varphi(x) \wedge \neg\exists z < x \varphi(z)\big)$$

*where φ is a formula in Φ. Similarly the number maximization axioms for Φ are denoted Φ-**MAX** and consist of the formulas*

$$\varphi(0) \supset \exists x \leq y\big(\varphi(x) \wedge \neg\exists z \leq y(x < z \wedge \varphi(z))\big)$$

where φ is a formula in Φ.

In the above definitions, $\varphi(x)$ is permitted to have free variables of both sorts, in addition to x.

Notice that all axioms of $\mathbf{V}^0$ hold in the standard model $\underline{\mathbb{N}}_2$ (page 77). In particular, all theorems of $\mathbf{V}^0$ about numbers are true in $\underline{\mathbb{N}}$. Indeed we will show that $\mathbf{V}^0$ is a conservative extension of $\mathbf{I\Delta}_0$: all theorems of $\mathbf{I\Delta}_0$ are theorems of $\mathbf{V}^0$, and all theorems of $\mathbf{V}^0$ over $\mathcal{L}_A$ are theorems of $\mathbf{I\Delta}_0$.

For the first direction, note that the above axiomatization of $\mathbf{V}^0$ contains no explicit induction axioms, so we need to show that it proves the number induction axioms for the Δ_0 formulas. In fact, we will show that it proves Σ_0^B-**IND** by showing first that it proves the X-**MIN** axiom, where

$$X\text{-}\mathbf{MIN} \;\equiv\; 0 < |X| \supset \exists x < |X|(X(x) \wedge \forall y < x \,\neg X(y))$$

LEMMA 5.6. $\mathbf{V}^0 \vdash X$-**MIN**.

PROOF. We reason in $\mathbf{V}^0$: By Σ_0^B-**COMP** there is a set Y such that $|Y| \leq |X|$ and for all $z < |X|$

$$Y(z) \leftrightarrow \forall y \leq z \,\neg X(y) \tag{50}$$

Thus the set Y consists of the numbers smaller than every element in X. Assuming $0 < |X|$, we will show that $|Y|$ is the least member of X. Intuitively, this is because $|Y|$ is the least number that is larger than any member of Y. Formally, we need to show: (i) $X(|Y|)$, and (ii) $\forall y < |Y| \neg X(y)$. Details are as follows.

First suppose that Y is empty. Then $|Y| = 0$ by **B**12 and **L**2, hence (ii) holds vacuously by Exercise 5.1 (a). Also, $X(0)$ holds, since otherwise $Y(0)$ holds by **B**7 and **B**9. Thus we have proved (i).

Now suppose that Y is not empty, i.e., $Y(y)$ holds for some y. Then $y < |Y|$ by **L**1, and thus $|Y| \neq 0$ by Exercise 5.1 (a). By **B**12, $|Y| = z+1$ for some z and hence $(Y(z) \wedge \neg Y(z+1))$ by **L**1 and **L**2. Hence by (50) we have

$$\forall y \leq z\ \neg X(y)\ \wedge\ \exists i \leq z+1\ X(i)$$

It follows that $i = z+1$ in the second conjunct, since if $i < z+1$ then $i \leq z$ by **B**11, which contradicts the first conjunct. This establishes (i) and (ii), since $i = z+1 = |Y|$. ⊣

Consider the following instance of Σ_0^B-**IND**:

$$X\text{-}\mathbf{IND}\ \equiv\ \big(X(0) \wedge \forall y < z(X(y) \supset X(y+1))\big) \supset X(z)$$

COROLLARY 5.7. $\mathbf{V}^0 \vdash X$-**IND**.

PROOF. We prove by contradiction. Assume $\neg X$-**IND**, then we have for some z:

$$X(0) \wedge \neg X(z) \wedge \forall y < z(X(y) \supset X(y+1))$$

By Σ_0^B-**COMP**, there is a set Y with $|Y| \leq z+1$ such that

$$\forall y < z+1\ (Y(y) \leftrightarrow \neg X(y))$$

Then $Y(z)$ holds by Exercise 5.1 (b), so $0 < |Y|$ by (a) and **L**1. By Y-**MIN**, Y has a least element y_0. Then $y_0 \neq 0$ because $X(0)$, hence $y_0 = x_0 + 1$ for some x_0, by **B**12. But then we must have $X(x_0)$ and $\neg X(x_0+1)$, which contradicts our assumption. ⊣

COROLLARY 5.8. *Let $\mathcal{T}$ be an extension of $\mathbf{V}^0$ and Φ be a set of formulas in $\mathcal{T}$. Suppose that $\mathcal{T}$ proves the Φ-**COMP** axiom scheme. Then $\mathcal{T}$ also proves the Φ-**IND** axiom scheme, the Φ-**MIN** axiom scheme, and the Φ-**MAX** axiom scheme.*

PROOF. We show that $\mathcal{T}$ proves the Φ-**IND** axiom scheme. This will show that $\mathbf{V}^0$ proves Σ_0^B-**IND**, and hence extends $\mathbf{I}\Delta_0$ and proves the arithmetic properties in Examples 3.8 and 3.9. The proof for the Φ-**MIN** and Φ-**MAX** axiom schemes is similar to that for Φ-**IND**, but easier since these properties are now available.

Let $\varphi(x) \in \Phi$. We need to show that

$$\mathcal{T} \vdash\ \big(\varphi(0) \wedge \forall y(\varphi(y) \supset \varphi(y+1))\big) \supset \varphi(z)$$

Reasoning in $\mathbf{V}^0$, assume

$$\varphi(0) \wedge \forall y(\varphi(y) \supset \varphi(y+1)) \tag{51}$$

By Φ-**COMP**, there exists X such that $|X| \leq z+1$ and

$$\forall y < z+1\ (X(y) \leftrightarrow \varphi(y)) \tag{52}$$

By **B**11, Exercise 5.1 (c,e) and (51) we conclude from this

$$X(0) \wedge \forall y < z(X(y) \supset X(y+1))$$

Finally $X(z)$ follows from this and X-**IND**, and so $\varphi(z)$ follows from (52) and Exercise 5.1 (b). ⊣

It follows from the corollary that for all $i \geq 0$, $\mathbf{V}^i$ proves Σ_i^B-**IND**, Σ_i^B-**MIN**, and Σ_i^B-**MAX**.

Theorem 5.9. *$\mathbf{V}^0$ is a conservative extension of* $\mathbf{I}\Delta_0$.

Proof. The axioms for $\mathbf{I}\Delta_0$ consist of $\mathbf{B}1, \ldots, \mathbf{B}8$ and the Δ_0-**IND** axioms. Since $\mathbf{B}1, \ldots, \mathbf{B}8$ are also axioms of $\mathbf{V}^0$, and we have just shown that $\mathbf{V}^0$ proves the Σ_0^B-**IND** axioms (which include the Δ_0-**IND** axioms), it follows that $\mathbf{V}^0$ extends $\mathbf{I}\Delta_0$. To show that $\mathbf{V}^0$ is conservative over $\mathbf{I}\Delta_0$ (i.e. theorems of $\mathbf{V}^0$ in the language of $\mathbf{I}\Delta_0$ are also theorems of $\mathbf{I}\Delta_0$), we prove the following lemma.

Lemma 5.10. *Any model $\mathcal{M}$ for $\mathbf{I}\Delta_0$ can be expanded to a model $\mathcal{M}'$ for $\mathbf{V}^0$, where the "number" part of $\mathcal{M}'$ is $\mathcal{M}$.*

Note that Theorem 5.9 follows immediately from the above lemma, because if φ is in the language of $\mathbf{I}\Delta_0$, then the truth of φ in $\mathcal{M}'$ depends only on the truth of φ in $\mathcal{M}$. (See the proof of the Extension by Definition Theorem 3.30.) $\dashv$

Proof of Lemma 5.10. Suppose that $\mathcal{M}$ is a model of $\mathbf{I}\Delta_0$ with universe $M = U_1$. Recall that $\mathbf{I}\Delta_0$ proves $\mathbf{B}1, \ldots, \mathbf{B}12$, so $\mathcal{M}$ satisfies these axioms. According to the semantics for $\mathcal{L}_A^2$ (Section 4B.2), to expand $\mathcal{M}$ to a model $\mathcal{M}'$ for $\mathbf{V}^0$ we must construct a suitable universe U_2 whose elements are determined by pairs (m, S), where $S \subseteq M$ and $m = |S|$. In order to satisfy axioms $\mathbf{L}1$ and $\mathbf{L}2$, if $S \in U_2$ is empty, then $|S| = 0$, and if S is nonempty, then S must have a largest element s and $|S| = s + 1$. Since $S \subseteq M$ and $|S|$ is determined by S, it follows that the extensionality axiom **SE** is satisfied.

The other requirement for U_2 is that the Σ_0^B-**COMP** axioms must be satisfied. We will construct U_2 to consist of all bounded subsets of M defined by Δ_0-formulas with parameters in M. We use the following conventional notation: If $\varphi(x)$ is a formula and c is an element in M, then $\varphi(c)$ represents $\varphi(x)$ with a constant symbol (also denoted c) substituted for x in φ, where it is understood that the symbol c is interpreted as the element c in M. If $\varphi(x, \vec{y})$ is a formula and $c, \vec{d}$ are elements of M, we use the notation

$$S(c, \varphi(x, \vec{d})) = \{e \in M | e < c \text{ and } \mathcal{M} \text{ satisfies } \varphi(e, \vec{d})\}.$$

Then we define

(53)

$$U_2 = \{S(c, \varphi(x, \vec{d})) \mid c, d_1, ..., d_k \in M \text{ and } \varphi(x, \vec{y}) \text{ is a } \Delta_0(\mathcal{L}_A) \text{ formula}\}$$

We must show that every nonempty element S of U_2 has a largest element, so that $|S|$ can be defined to satisfy $\mathbf{L}1$ and $\mathbf{L}2$. The largest element exists because the differences between the upper bound c for S and elements of S have a minimum element, by Δ_0-**MIN**. Specifically, if $S = S(c, \varphi(x, \vec{d}))$ is nonempty and m is the least z satisfying

$\varphi(c \dot- 1 \dot- z, \vec{d})$, then define $|S| = \ell_\varphi(c, \vec{d})$ where $\ell_\varphi(c, \vec{d}) = c \dot- m$. Then

$$\ell_\varphi(c, \vec{d}) = \begin{cases} \sup(S(c, \varphi(x, \vec{d})) + 1 & \text{if } S \neq \varnothing \\ 0 & \text{otherwise} \end{cases}$$

The preceding argument shows that the function $\ell_\varphi(z, \vec{y})$ is provably total in $\mathbf{I}\Delta_0$.

It remains to show that Σ_0^B-**COMP** holds in $\mathcal{M}'$. This means that for every Σ_0^B formula $\psi(z, \vec{x}, \vec{Y})$ (with all free variables indicated) and for every vector $\vec{d}$ of elements of M interpreting $\vec{x}$ and every vector $\vec{S}$ of elements in U_2 interpreting $\vec{Y}$ and for every $c \in M$, the set

$$T = \{e \in M \mid e < c \text{ and } \mathcal{M}' \models \psi(e, \vec{d}, \vec{S})\} \tag{54}$$

must be in U_2. Suppose that

$$S_i = S(c_i, \varphi_i(u, \vec{d_i}))$$

for some Δ_0 formulas $\varphi_i(x, \vec{y_i})$. Let $\theta(z, \vec{x}, \vec{y}_1, \vec{y}_2, \dots, w_1, w_2, \dots)$ be the result of replacing every sub-formula of the form $Y_i(t)$ in $\psi(z, \vec{x}, \vec{Y})$ by $(\varphi_i(t, \vec{y_i}) \wedge t < w_i)$ and every occurrence of $|Y_i|$ by $\ell_{\varphi_i}(w_i, \vec{y_i})$. (We may assume that ψ has no occurrence of $=_2$ by replacing every equation $X =_2 Z$ by a Σ_0^B formula using the extensionality axiom **SE**.) Finally let

$$T = S(c, \theta(z, \vec{d}, \vec{d}_1, \vec{d}_2, \dots, c_1, c_2, \dots)).$$

Then T satisfies (54). Since the functions ℓ_{φ_i} are Σ_1-definable in $\mathbf{I}\Delta_0$, by the Conservative Extension Lemma 3.35, θ can be transformed into an equivalent $\Delta_0(\mathcal{L}_A)$ formula. Thus $T \in U_2$. $\dashv$

EXERCISE 5.11. *Suppose that instead of defining U_2 according to (53), we defined U_2 to consist of all subsets of M which have a largest element, together with $\varnothing$. Then for each set $S \subset U_1$ in U_2 we define $|S|$ in the obvious way to satisfy axioms* **L***1 and* **L***2. Prove that if $\mathcal{M}$ is a nonstandard model of $\mathbf{I}\Delta_0$, then the resulting two-sorted structure (U_1, U_2) is not a model of $\mathbf{V}^0$.*

EXERCISE 5.12. *Suppose that we want to prove that $\mathbf{V}^0$ is conservative over $\mathbf{I}\Delta_0$ by considering an anchored $\mathbf{LK}^2$ proof instead of the above model-theoretic argument. Here we consider a small part of such an argument. Suppose that φ is an $\mathbf{I}\Delta_0$ formula and π is an anchored $\mathbf{LK}^2$-$\mathbf{V}^0$ proof of $\longrightarrow \varphi$. Suppose (to make things easy) that no formula in π contains a string quantifier. Show explicitly how to convert π to an $\mathbf{LK}$-$\mathbf{I}\Delta_0$ proof π' of $\longrightarrow \varphi$.*

Since according to Theorem 5.9 $\mathbf{V}^0$ extends $\mathbf{I}\Delta_0$, we will freely use the results in Chapter 3 when reasoning in $\mathbf{V}^0$ in the sequel.

5B. Two-Sorted Functions

Complexity classes of two-sorted relations were discussed in Section 4C Now we associate with each two-sorted complexity class $\mathbf{C}$ of relations a two-sorted function class $\mathbf{FC}$. Two-sorted functions are either *number functions* or *string functions*. A number function $f(\vec{x}, \vec{Y})$ takes values in $\mathbb{N}$, and a string function $F(\vec{x}, \vec{Y})$ takes finite subsets of $\mathbb{N}$ as values.

Definition 5.13. *A function f or F is* polynomially bounded *(or* p-bounded*) if there is a polynomial $p(\vec{x}, \vec{y})$ such that $f(\vec{x}, \vec{Y}) \le p(\vec{x}, |\vec{Y}|)$ or $|F(\vec{x}, \vec{Y})| \le p(\vec{x}, |\vec{Y}|)$.*

All function complexity classes we consider here contain only p-bounded functions.

In defining the functions associated with a complexity class of relations the natural relation to use for a number function is its graph. However this does not work well for string functions. For example the function $F(X)$ which gives the prime factorization of X (considered as a binary number) is not known to be polynomial time computable, but its graph is a polynomial time relation. It turns out that the right relation to associate with a string function is its *bit graph*.

Definition 5.14 (Graph, Bit Graph). *The* graph G_f *of a number function $f(\vec{x}, \vec{Y})$ is defined by*

$$G_f(z, \vec{x}, \vec{Y}) \leftrightarrow z = f(\vec{x}, \vec{Y})$$

The bit graph B_F *of a string function $F(\vec{x}, \vec{Y})$ is defined by*

$$B_F(i, \vec{x}, \vec{Y}) \leftrightarrow F(\vec{x}, \vec{Y})(i).$$

Definition 5.15 (Function Class). *If $\mathbf{C}$ is a two-sorted complexity class of relations, then the corresponding function class $\mathbf{FC}$ consists of all p-bounded number functions whose graphs are in $\mathbf{C}$, together with all p-bounded string functions whose bit graphs are in $\mathbf{C}$.*

In particular, the string functions in $\mathbf{FAC}^0$ are those p-bounded functions whose bit graphs are in $\mathbf{AC}^0$. The nonuniform version $\mathbf{FAC}^0/poly$ has a nice circuit characterization like that of $\mathbf{AC}^0/poly$ (see page 73). Thus a string function $F(X)$ is in $\mathbf{FAC}^0/poly$ iff there is a polynomial size bounded depth family $\langle C_n \rangle$ of Boolean circuits (with unbounded fan-in $\wedge$-gates and $\vee$-gates) such that each C_n has n input bits specifying the input string X, and the output bits of C_n specify the string $F(X)$.

The following characterization of $\mathbf{FAC}^0$ follows from the above definitions and the Σ_0^B Representation Theorem (Theorem 4.18).

Corollary 5.16. *A string function is in* $\mathbf{FAC}^0$ *if and only if it is p-bounded, and its bit graph is represented by a* Σ_0^B *formula. The same holds for a number function, with graph replacing bit graph.*

An interesting example of a string function in $\mathbf{FAC}^0$ is binary addition. Note that as in (46) we can treat a finite subset $X \subset \mathbb{N}$ as the natural number

$$bin(X) = \sum_i X(i)2^i$$

where we write 0 for $\bot$ and 1 for $\top$. We will write $X + Y$ for the string function "binary addition", so $X + Y = bin(X) + bin(Y)$. Let $Carry(i, X, Y)$ hold iff there is a carry into bit position i when computing $X + Y$. Then $Carry(i, X, Y)$ is represented by the Σ_0^B formula given in (47).

The bit graph of $X + Y$ can be defined as follows.

Example 5.17 (Bit Graph of String Addition). *The bit graph of $X+Y$ is*

$$(55)\quad (X + Y)(i) \leftrightarrow \big(i < |X| + |Y| \wedge (X(i) \oplus Y(i) \oplus Carry(i, X, Y))\big)$$

where $p \oplus q \equiv ((p \wedge \neg q) \vee (\neg p \wedge q))$.

In general, the graph $G_F(\vec{x}, \vec{Y}, Z) \equiv (Z = F(\vec{x}, \vec{Y}))$ of a string function $F(\vec{x}, \vec{Y})$ can be defined from its bit graph as follows:

$$G_F(\vec{x}, \vec{Y}, Z) \;\leftrightarrow\; \forall i\,(Z(i) \leftrightarrow B_F(i, \vec{x}, \vec{Y}))$$

So if F is polynomially bounded and its bit graph is in $\mathbf{AC}^0$, then its graph is also in $\mathbf{AC}^0$, because

$$(56)\qquad G_F(\vec{x}, \vec{Y}, Z) \;\leftrightarrow\; \big(|Z| \leq t \;\wedge\; \forall i < t\,(Z(i) \leftrightarrow B_F(i, \vec{x}, \vec{Y}))\big)$$

where t is the bound on the length of F.

As we noted earlier (Section 4A), the relation $R_\times$ is not in $\mathbf{AC}^0$, where

$$R_\times(X, Y, Z) \leftrightarrow bin(X) \cdot bin(Y) = bin(Z)$$

(because PARITY, which is not in $\mathbf{AC}^0$, is reducible to it). As a result, the bit graph of $(X \times Y)(i)$ is not representable by any Σ_0^B formula, where $X \times Y = bin(X) \cdot bin(Y)$ is the string function "binary multiplication".

If a string function $F(X)$ is polynomially bounded, it is not enough to say that its graph is an $\mathbf{AC}^0$ relation in order to ensure that $F \in \mathbf{FAC}^0$. For example, let M be a fixed polynomial-time Turing machine, and define $F(X)$ to be a string coding the computation of M on input X. If the computation is nicely encoded then $F(X)$ is polynomially bounded and the graph $Y = F(X)$ is an $\mathbf{AC}^0$ relation, but if the Turing machine computes a function not in $\mathbf{AC}^0$ (such as the number of ones in X) then $F \notin \mathbf{FAC}^0$.

For the same reason that the numerical $\mathbf{AC}^0$ relations in the two-sorted setting are precisely the **LTH** relations in the single-sorted setting (see the proof of the Σ_0^B Representation Theorem, 4.18), number functions with no string arguments are $\mathbf{AC}^0$ functions iff they are single-sorted **LTH** functions.

The nonuniform version of $\mathbf{FAC}^0$ consists of functions computable by bounded-depth polynomial-size circuits, and it is clear from this definition that the class is closed under composition. It is also clear that nonuniform $\mathbf{AC}^0$ is closed under substitution of (nonuniform) $\mathbf{AC}^0$ functions for parameters. These are some of the natural properties that also hold for uniform $\mathbf{AC}^0$ and $\mathbf{FAC}^0$.

Exercise 5.18. *Show that a number function $f(\vec{x}, \vec{X})$ is in $\mathbf{FAC}^0$ if and only if*

$$f(\vec{x}, \vec{X}) = |F(\vec{x}, \vec{X})|$$

for some string function $F(\vec{x}, \vec{X})$ in $\mathbf{FAC}^0$.

Theorem 5.19. (a) *The $\mathbf{AC}^0$ relations are closed under substitution of $\mathbf{AC}^0$ functions for variables.*

(b) *The $\mathbf{AC}^0$ functions are closed under composition.*

(c) *The $\mathbf{AC}^0$ functions are closed under* definition by cases*, i.e., if φ is an $\mathbf{AC}^0$ relation, g, h and G, H are functions in $\mathbf{FAC}^0$, then the functions f and F defined by*

$$f = \begin{cases} g & \text{if } \varphi, \\ h & \text{otherwise} \end{cases} \qquad\qquad F = \begin{cases} G & \text{if } \varphi, \\ H & \text{otherwise} \end{cases}$$

are also in $\mathbf{FAC}^0$.

Proof. We will prove (a) for the case of substituting a string function for a string variable. The case of substituting a number function for a number variable is left as an easy exercise. Part (b) follows easily from part (a). We leave part (c) as an exercise.

Suppose that $R(\vec{x}, \vec{X}, Y)$ is an $\mathbf{AC}^0$ relation and $F(\vec{x}, \vec{X})$ an $\mathbf{AC}^0$ function. We need to show that the relation $Q(\vec{x}, \vec{X}) \equiv R(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$ is also an $\mathbf{AC}^0$ relation, i.e., it is representable by some Σ_0^B formula.

By the Σ_0^B Representation Theorem (4.18) there is a Σ_0^B formula $\varphi(\vec{x}, \vec{X}, Y)$ that represents R:

$$R(\vec{x}, \vec{X}, Y) \leftrightarrow \varphi(\vec{x}, \vec{X}, Y)$$

By Corollary 5.16 there is a Σ_0^B formula $\theta(i, \vec{x}, \vec{X})$ and a number term $t(\vec{x}, \vec{X})$ such that

$$F(\vec{x}, \vec{X})(i) \leftrightarrow i < t(\vec{x}, \vec{X}) \wedge \theta(i, \vec{x}, \vec{X}). \tag{57}$$

It follows from Exercise 5.18 that the relation $z = |F(\vec{x}, \vec{X})|$ is represented by a Σ_0^B formula η, so

$$z = |F(\vec{x}, \vec{X})| \leftrightarrow \eta(z, \vec{x}, \vec{X}) \tag{58}$$

The Σ_0^B formula that represents the relation $Q(\vec{x}, \vec{X})$ is obtained from $\varphi(\vec{x}, \vec{X}, Y)$ by successively eliminating each occurrence of Y using (57) and (58) as follows.

First eliminate all atomic formulas of the form $Y = Z$ (or $Z = Y$) in φ by replacing them with equivalent formulas using the extensionality axiom **SE**. Thus

$$Y = Z \leftrightarrow (|Y| = |Z|) \wedge \forall i < |Y|(Y(i) \leftrightarrow Z(i))$$

Now Y can only occur in the form $|Y|$ or $Y(r)$, for some term r. Any occurrence of $|Y|$ in $\varphi(\vec{x}, \vec{X}, Y)$ must be in the context of an atomic formula $\psi(\vec{x}, \vec{X}, |Y|)$, which we replace with

$$\exists z \leq t(\vec{x}, \vec{X})\ (\eta(z, \vec{x}, \vec{X}) \wedge \psi(\vec{x}, \vec{X}, z))$$

Finally we replace each occurrence of $Y(r)$ in $\varphi(\vec{x}, \vec{X}, Y)$ by

$$r < t(\vec{x}, \vec{X}) \wedge \theta(r, \vec{x}, \vec{X})$$

The result is a Σ_0^B formula which represents $Q(\vec{x}, \vec{X})$. ⊣

EXERCISE 5.20. *Prove part (a) of Theorem 5.19 for the case of substitution of number functions for variables. Also prove parts (b) and (c) of the theorem.*

5C. Parikh's Theorem for Two-Sorted Logic

Recall (Section 3B) that a term $t(\vec{x})$ is a bounding term for a function symbol f in a single-sorted theory $\mathcal{T}$ if

$$\mathcal{T} \vdash \forall \vec{x}\ f(\vec{x}) \leq t(\vec{x})$$

For a two-sorted theory $\mathcal{T}$ whose vocabulary is an extension of $\mathcal{L}_A^2$, we say that a number term $t(\vec{x}, \vec{X})$ is a bounding term for a number function f in $\mathcal{T}$ if

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ f(\vec{x}, \vec{X}) \leq t(\vec{x}, \vec{X})$$

Also, $t(\vec{x}, \vec{X})$ is a bounding term for a string function F in $\mathcal{T}$ if

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ |F(\vec{x}, \vec{X})| \leq t(\vec{x}, \vec{X})$$

DEFINITION 5.21. *A number function or a string function is* polynomially bounded *in $\mathcal{T}$ if it has a bounding term in the language $\mathcal{L}_A^2$.*

Exercise 5.22. *Let $\mathcal{T}$ be a two-sorted theory over the vocabulary $\mathcal{L} \supseteq \mathcal{L}_A^2$. Suppose that $\mathcal{T}$ extends $\mathbf{I}\Delta_0$. Show that if the functions of $\mathcal{L}$ are polynomially bounded in $\mathcal{T}$, then for each number term $s(\vec{x}, \vec{X})$ and string term $T(\vec{x}, \vec{X})$ of $\mathcal{L}$, there is an $\mathcal{L}_A^2$-number term $t(\vec{x}, \vec{X})$ such that*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ s(\vec{x}, \vec{X}) \leq t(\vec{x}, \vec{X}) \qquad \textit{and} \qquad \mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ |T(\vec{x}, \vec{X})| \leq t(\vec{x}, \vec{X})$$

Note that a bounded formula is one in which every quantifier (both string and number quantifiers) is bounded. Recall the definition of a polynomial-bounded single-sorted theory (Definition 3.19).

In two-sorted logic, a polynomial-bounded theory is required to extend $\mathbf{V}^0$. The formal definition follows.

Definition 5.23. *[Polynomial-bounded Two-sorted Theory] Let $\mathcal{T}$ be a two-sorted theory over the vocabulary $\mathcal{L}$. Then $\mathcal{T}$ is a* polynomial-bounded theory *if (i) it extends $\mathbf{V}^0$; (ii) it can be axiomatized by a set of bounded formulas; and (iii) each function f or F in $\mathcal{L}$ is polynomially bounded in $\mathcal{T}$.*

Note that each theory $\mathbf{V}^i, i \geq 0$, is a polynomial-bounded theory. In fact, all two-sorted theories considered in this book are polynomial-bounded.

Theorem 5.24 (Parikh's Theorem, Two-sorted Case). *Suppose that $\mathcal{T}$ is a polynomial-bounded theory and $\varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$ is a bounded formula with all free variables indicated such that*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \exists \vec{Y} \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y}) \tag{59}$$

Then

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \leq t \exists \vec{Y} \leq t \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y}) \tag{60}$$

for some $\mathcal{L}_A^2$-term $t = t(\vec{x}, \vec{X})$ containing only the variables $(\vec{x}, \vec{X})$.

It follows from Exercise 5.22 that the bounding term t can be taken to be a term in $\mathcal{L}_A^2$.

It suffices to prove the following simple form of the above theorem.

Lemma 5.25. *Suppose that $\mathcal{T}$ is a polynomial-bounded theory, and $\varphi(z, \vec{x}, \vec{X})$ is a bounded formula with all free variables indicated such that*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists z \varphi(z, \vec{x}, \vec{X})$$

Then

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists z \leq t(\vec{x}, \vec{X}) \varphi(z, \vec{x}, \vec{X})$$

for some term $t(\vec{x}, \vec{X})$ with all variables indicated.

Proof of Parikh's Theorem from Lemma 5.25. Define (omitting $\vec{x}$ and $\vec{X}$)

$$\psi(z) \equiv \exists \vec{y} \leq z \exists \vec{Y} \leq z \varphi(\vec{y}, \vec{Y})$$

From the assumption (59) we conclude that $\mathcal{T} \vdash \exists z \psi(z)$, since we can take

$$z = y_1 + \ldots + y_k + |Y_1| + \ldots + |Y_\ell|$$

Since φ is a bounded formula, ψ is also a bounded formula. By the lemma, we conclude that $\mathcal{T}$ proves $\exists z \le t\psi(z)$, where the variables in t satisfy Parikh's Theorem. Thus (60) follows. $\dashv$

PROOF OF LEMMA 5.25. The proof is the same as the proof of Parikh's Theorem in the single-sorted logic (page 44), with minor modifications. Refer to Section 4D for the system $\mathbf{LK}^2$. Here we consider an anchored $\mathbf{LK}^2$-$\mathbf{T}$ proof π of $\exists z\varphi(z, \vec{a}, \vec{\alpha})$, where $\mathbf{T}$ is the set of all term substitution instances of axioms of $\mathcal{T}$ (note that now we have both the substitution of number terms for number variables and string terms for string variables). We assume that π is in free variable normal form (see Section 4D.1).

We convert π to a proof π' by converting each sequent $\mathcal{S}$ in π into a sequent $\mathcal{S}'$ and providing an associated derivation $\mathbf{D}(\mathcal{S})$, where $\mathcal{S}'$ and $\mathbf{D}(\mathcal{S})$ are defined by induction on the depth of $\mathcal{S}$ in π so that the following is satisfied:

Induction Hypothesis: If $\mathcal{S}$ has no occurrence of $\exists y\varphi$, then $\mathcal{S}' = \mathcal{S}$. If $\mathcal{S}$ has one or more occurrences of $\exists y\varphi$, then $\mathcal{S}'$ is a sequent which is the same as $\mathcal{S}$ except all occurrences of $\exists y\varphi$ are replaced by a single occurrence of $\exists y \le t\varphi$, where t is an $\mathcal{L}_A^2$-number term that depends on $\mathcal{S}$ and the placement of $\mathcal{S}$ in π. Further every variable in t occurs free in the original sequent $\mathcal{S}$.

As discussed in Section 4D.1, if the underlying vocabulary has no string constant symbol (for example $\mathcal{L}_A^2$), then we allow the string constant $\varnothing$ to occur in π, in order to assume that it is in free variable normal form. Thus the bounding term t in the endsequent $\longrightarrow \exists y \le t\varphi$ may contain $\varnothing$. Since t is an $\mathcal{L}_A^2(\varnothing)$-term, each occurrence of $\varnothing$ is in the context $|\varnothing|$, and hence can be replaced by 0 using the axiom **E**: $|\varnothing| = 0$.

The **Cases I–V** are supplemented to consider the four string quantifier rules, which are treated in the same way as their **LK** counterparts. $\dashv$

5D. Definability in $\mathbf{V}^0$

Recall the notion of Φ-definable single-sorted function (Definition 3.27). For a two-sorted theory $\mathcal{T}$, this notion is defined in the same way for functions of each sort, and in particular $\mathcal{T}$ must be able to prove existence and uniqueness of function values.

DEFINITION 5.26 (Two-sorted Definability). *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L} \supseteq \mathcal{L}_A^2$, and let Φ be a set of $\mathcal{L}$-formulas. A number function f not in $\mathcal{L}$ is* Φ-definable in $\mathcal{T}$ *if there is a formula $\varphi(\vec{x}, y\vec{X})$ in Φ such that*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists! y \varphi(\vec{x}, y, \vec{X}) \tag{61}$$

and

$$y = f(\vec{x}, \vec{X}) \leftrightarrow \varphi(\vec{x}, y, \vec{X}) \tag{62}$$

A string function F not in $\mathcal{L}$ is Φ-definable in $\mathcal{T}$ *if there is a formula $\varphi(\vec{x}, \vec{X}, Y)$ in Φ such that*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists! Y \varphi(\vec{x}, \vec{X}, Y) \tag{63}$$

and

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \varphi(\vec{x}, \vec{X}, Y) \tag{64}$$

Then (62) is a defining axiom *for f and (64) is a* defining axiom *for F, and we write $\mathcal{T}(f)$ or $\mathcal{T}(F)$ for the theory extending $\mathcal{T}$ by adding f or F and its corresponding defining axiom to $\mathcal{T}$. We say that f or F is* definable in $\mathcal{T}$ *if it is Φ-definable in $\mathcal{T}$ for some Φ.*

THEOREM 5.27 (Two-sorted Extension by Definition). *$\mathcal{T}(f)$ and $\mathcal{T}(F)$ (as defined above) are conservative extensions of $\mathcal{T}$.*

PROOF. This is proved in the same way as its single-sorted version Theorem 3.30. $\dashv$

If Φ is the set of all $\mathcal{L}_A^2$-formulas, then every arithmetical function (that is, every function whose graph is represented by an $\mathcal{L}_A^2$-formula) is Φ-definable in $\mathbf{V}^0$. To see this, suppose that $F(\vec{x}, \vec{X})$ has defining axiom (64). Then the graph of F is also defined by the following formula $\varphi'(\vec{x}, \vec{X}, Y)$:

$$(\exists! Z \varphi(\vec{x}, \vec{X}, Z) \wedge \varphi(\vec{x}, \vec{X}, Y)) \vee (\neg \exists! Z \varphi(\vec{x}, \vec{X}, Z) \wedge Y = \varnothing)$$

Then (63) with φ' for φ is trivially provable in $\mathbf{V}^0$.

We want to choose a standard class Φ of formulas such that the class of Φ-definable functions in a theory $\mathcal{T}$ depends nicely on the proving power of $\mathcal{T}$, so that various complexity classes can be characterized by fixing Φ and varying $\mathcal{T}$. In single-sorted logic, our choice for Φ was Σ_1, and we defined the provably total functions of $\mathcal{T}$ to be the Σ_1-definable functions in $\mathcal{T}$. Here our choice for Φ is Σ_1^1 (recall (Definition 4.14) that a Σ_1^1 formula is a formula of the form $\exists \vec{X} \varphi$, where φ is a Σ_0^B formula). The notion of a *provably total function* in two-sorted logic is defined as follows.

DEFINITION 5.28 (Provably Total Function). *A function (which can be either a number function or a string function) is said to be* provably total *in a theory $\mathcal{T}$ iff it is Σ_1^1-definable in $\mathcal{T}$.*

If $\mathcal{T}$ consists of all formulas of $\mathcal{L}_A^2$ which are true in the standard model $\underline{\mathbb{N}}_2$, then the functions provably total in $\mathcal{T}$ are precisely all total functions computable on a Turing machine. The idea here is that the existential string quantifiers in a Σ_1^1 formula can be used to code the computation of a Turing machine computing the function. If $\mathcal{T}$ is a polynomially bounded theory, then both the function values and the computation must be polynomially bounded. In fact, the following result in a corollary of Parikh's Theorem.

Corollary 5.29. *Let $\mathcal{T}$ be a polynomial-bounded theory. Then all provably total functions in $\mathcal{T}$ are polynomially bounded. A function is provably total in $\mathcal{T}$ iff it is Σ_1^B-definable in $\mathcal{T}$.*

We will show that the provably total functions in $\mathbf{V}^0$ are precisely the functions in $\mathbf{FAC}^0$, and in the next chapter we will show that the provably total functions in $\mathbf{V}^1$ are precisely the polynomial time functions. Later we will give similar characterizations of other complexity classes.

Exercise 5.30. *Show that for any theory $\mathcal{T}$ whose vocabulary includes $\mathcal{L}_A^2$, the provably total functions of $\mathcal{T}$ are closed under composition.*

In two-sorted logic, for string functions we have the notion of a *bit-definable function* in addition to that of a definable function.

Definition 5.31 (Bit-definable Function). *Let Φ be a set of $\mathcal{L}$ formulas where $\mathcal{L} \supseteq \mathcal{L}_A^2$. We say that a string function symbol $F(\vec{x}, \vec{Y})$ not in $\mathcal{L}$ is* Φ-bit-definable *from $\mathcal{L}$ if there is a formula $\varphi(i, \vec{x}, \vec{Y})$ in Φ and an $\mathcal{L}_A^2$-number term $t(\vec{x}, \vec{Y})$ such that the bit graph of F satisfies*

$$F(\vec{x}, \vec{Y})(i) \leftrightarrow (i < t(\vec{x}, \vec{Y}) \wedge \varphi(i, \vec{x}, \vec{Y})) \tag{65}$$

We say that the formula on the RHS of (65) *is a* bit-defining axiom*, or* bit definition*, of F.*

The choice of φ and t in the above definition is not uniquely determined by F. However we will assume that a specific formula φ and a specific number term t has been chosen, so we will speak of *the bit-defining axiom*, or *the bit definition*, of F. Note also that such a F is polynomially bounded in $\mathcal{T}$, and t is a bounding term for F.

The following proposition follows easily from the above definition and Corollary 5.16.

Proposition 5.32. *A string function is Σ_0^B-bit-definable iff it is in* $\mathbf{FAC}^0$.

Exercise 5.33. *Let $\mathcal{T}$ be a theory which extends $\mathbf{V}^0$ and proves the bit-defining axiom (65) for a string function F, where φ is a Σ_0^B formula. Show that there is a Σ_0^B formula $\eta(z, \vec{x}, \vec{Y})$ such that $\mathcal{T}$ proves*

$$z = |F(\vec{x}, \vec{Y})| \leftrightarrow \eta(z, \vec{x}, \vec{Y})$$

It is important to distinguish between a "definable function" and a "bit-definable function". In particular, if a theory $\mathcal{T}_2$ is obtained from a theory $\mathcal{T}_1$ by adding a Φ-bit-definable function F together with its bit-defining axiom (65), then in general we cannot conclude that $\mathcal{T}_2$ is a conservative extension of $\mathcal{T}_1$. For example, it is easy to show that the string multiplication function $X \times Y$ has a Σ_1^B bit definition. However, as we noted earlier, this function is not Σ_1^B-definable in $\mathbf{V}^0$. The theory that results from adding this function together with its Σ_1^B-bit-definition to $\mathbf{V}^0$ is *not* a conservative extension of $\mathbf{V}^0$.

To get definability, and hence conservativity, it suffices to assume that $\mathcal{T}_1$ proves a comprehension axiom scheme. The following definition is useful here and in Chapter 6.

DEFINITION 5.34 (Σ_0^B-Closure). *Let Φ be a set of formulas over a language $\mathcal{L}$ which extends $\mathcal{L}_A^2$. Then $\Sigma_0^B(\Phi)$ is the closure of Φ under the operations $\neg, \wedge, \vee$ and bounded number quantification. That is, if φ and ψ are formulas in $\Sigma_0^B(\Phi)$ and t is an $\mathcal{L}_A^2$-term not containing x, then the following formulas are also in $\Sigma_0^B(\Phi)$: $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $\forall x \le t\varphi$, and $\exists x \le t\varphi$.*

LEMMA 5.35 (Extension by Bit Definition Lemma). *Let $\mathcal{T}$ be a theory over $\mathcal{L}$ that contains $\mathbf{V}^0$, and Φ be a set of $\mathcal{L}$-formulas such that $\Phi \supseteq \Sigma_0^B$. Suppose that $\mathcal{T}$ proves the Φ-**COMP** axiom scheme. Then any polynomially bounded number function whose graph is Φ-representable, or a polynomially bounded string function whose bit graph is Φ-representable, is $\Sigma_0^B(\Phi)$-definable in $\mathcal{T}$.*

PROOF. Consider the case of a string function. Suppose that F is a polynomially bounded string function with bit graph in Φ, so there are an $\mathcal{L}_A^2$-number term t and a formula $\varphi \in \Phi$ such that

$$F(\vec{x}, \vec{Y})(i) \leftrightarrow (i < t(\vec{x}, \vec{Y}) \wedge \varphi(i, \vec{x}, \vec{Y}))$$

As in (56), the graph G_F of F can be defined as follows:

$$G_F(\vec{x}, \vec{Y}, Z) \;\equiv\; |Z| \le t \;\wedge\; \forall i < t\,(Z(i) \leftrightarrow \varphi(i, \vec{x}, \vec{Y})) \tag{66}$$

Now since $\mathcal{T}$ proves the Φ-**COMP**, we have

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{Y}\exists Z\ G_F(\vec{x}, \vec{Y}, Z) \tag{67}$$

Also $\mathcal{T}$ proves that such Z is unique, by the extensionality axiom **SE** in 2-**BASIC**. Since the formula $G_F(\vec{x}, \vec{Y}, Z)$ is in $\Sigma_0^B(\Phi)$, it follows that F is $\Sigma_0^B(\Phi)$-definable in $\mathcal{T}$.

Next consider the case of a number function. Let f be a polynomially bounded number function whose graph is in Φ, so there are an $\mathcal{L}_A^2$-number term t and a formula $\varphi \in \Phi$ such that

$$y = f(\vec{x}, \vec{X}) \leftrightarrow (y < t(\vec{x}, \vec{X}) \wedge \varphi(y, \vec{x}, \vec{X}))$$

By Corollary 5.8, $\mathcal{T}$ proves the Φ-**MIN** axiom scheme. Therefore f is definable in $\mathcal{T}$ by using the following $\Sigma_0^B(\Phi)$ formula for its graph:

$$G_f(y, \vec{x}, \vec{X}) \equiv (\forall z < y \neg\varphi(z, \vec{x}, \vec{X})) \wedge (y < t \supset \varphi(y, \vec{x}, \vec{X})) \tag{68}$$

(i.e., y is the least number $< t$ such that $\varphi(y)$ holds, or t if no such y exists). ⊣

In this lemma, if we take $\mathcal{T} = \mathbf{V}^0$ and $\Phi = \Sigma_0^B$, then (since $\Sigma_0^B(\Sigma_0^B) = \Sigma_0^B$) we can apply Corollary 5.16 and Proposition 5.32 to obtain the following:

COROLLARY 5.36. *Every function in* $\mathbf{FAC}^0$ *is* Σ_0^B*-definable in* $\mathbf{V}^0$.

This result can be generalized, using the following definition.

DEFINITION 5.37. [3] *A string function is* Σ_0^B-definable *from a collection* $\mathcal{L}$ *of two-sorted functions and relations if it is p-bounded and its bit graph is represented by a* $\Sigma_0^B(\mathcal{L})$ *formula. Similarly, a number function is* Σ_0^B-definable *from* $\mathcal{L}$ *if it is p-bounded and its graph is represented by a* $\Sigma_0^B(\mathcal{L})$ *formula.*

This "semantic" notion of Σ_0^B-definability should not be confused with Σ_0^B-definability in a theory (Definition 5.26), which involves provabililty. The next result connects the two notions.

COROLLARY 5.38. *Let* $\mathcal{T}$ *be a theory over* $\mathcal{L}$ *that contains* $\mathbf{V}^0$*, and suppose that* $\mathcal{T}$ *proves the* $\Sigma_0^B(\mathcal{L})$-**COMP** *axiom scheme. Then a function which is* Σ_0^B*-definable from* $\mathcal{L}$ *is* $\Sigma_0^B(\mathcal{L})$*-definable in* $\mathcal{T}$.

Later we will prove the Witnessing Theorem for $\mathbf{V}^0$, which says that any Σ_1^1-definable function of $\mathbf{V}^0$ is in $\mathbf{FAC}^0$. This will complete our characterization of $\mathbf{FAC}^0$ by $\mathbf{V}^0$. (Compare this with Proposition 5.32, which characterizes $\mathbf{FAC}^0$ in terms of bit-definability, independent of any theory.)

COROLLARY 5.39. *Suppose that the theory* $\mathcal{T}$ *proves* $\Sigma_0^B(\mathcal{L})$-**COMP**, *where* $\mathcal{L}$ *is the vocabulary of* $\mathcal{T}$. *Then the theory resulting from* $\mathcal{T}$ *by adding the* $\Sigma_0^B(\mathcal{L})$*-defining axioms or the* $\Sigma_0^B(\mathcal{L})$*-bit-defining axioms for a collection of number functions and string functions is a conservative extension of* $\mathcal{T}$.

The following result shows in particular that if we extend $\mathbf{V}^0$ by a sequence of Σ_0^B defining axioms and bit-defining axioms, the resulting theory is not only conservative over $\mathbf{V}^0$, it also proves the $\Sigma_0^B(\mathcal{L})$-**COMP** and $\Sigma_0^B(\mathcal{L})$-**IND** axioms, where $\mathcal{L}$ is the resulting vocabulary. We state it generally for $\Sigma_i^B(\mathcal{L})$ formulas.

LEMMA 5.40 (Σ_0^B-Transformation Lemma). *Let* $\mathcal{T}$ *be a polynomial-bounded theory which extends* $\mathbf{V}^0$*, and assume that the vocabulary* $\mathcal{L}$ *of* $\mathcal{T}$ *has the same predicate symbols as* $\mathcal{L}_A^2$. *Suppose that for every*

[3] This notion is important for our definition of $\mathbf{AC}^0$ reduction, Definition 9.1.

number function f in $\mathcal{L}$, $\mathcal{T}$ proves a $\Sigma_0^B(\mathcal{L}_A^2)$ defining axiom for f, and for every string function F in $\mathcal{L}$, $\mathcal{T}$ proves a $\Sigma_0^B(\mathcal{L}_A^2)$ bit-defining axiom for F. Then for every $i \geq 0$ and every $\Sigma_i^B(\mathcal{L})$ formula φ^+ there is a $\Sigma_i^B(\mathcal{L}_A^2)$ formula φ such that

$$\mathcal{T} \vdash \varphi^+ \leftrightarrow \varphi$$

Proof. We prove the conclusion for the case $i = 0$. The case $i > 0$ follows immediately from this case. We may assume by the axiom **SE** that φ^+ does not contain $=_2$. We proceed by induction on the maximum nesting depth of any function symbol in φ^+, where in defining nesting depth we only count functions which are in $\mathcal{L}$ but not in $\mathcal{L}_A^2$. The base case is nesting depth 0, so φ^+ is already a $\Sigma_0^B(\mathcal{L}_A^2)$ formula, and there is nothing to prove.

For the induction step, assume that φ^+ has at least one occurrence of a function not in $\mathcal{L}_A^2$. It suffices to consider the case in which φ^+ is an atomic formula. Since by assumption the only predicate symbols in $\mathcal{L}$ are those in $\mathcal{L}_A^2$, the only predicate symbols we need consider are $\epsilon, =, \leq$. First consider the case $\in$, so φ^+ has the form $F(\vec{t}, \vec{T})(s)$. Then by assumption $\mathcal{T}$ proves a bit definition of the form

$$F(\vec{x}, \vec{X})(i) \;\leftrightarrow\; (i < r(\vec{x}, \vec{X}) \wedge \psi(i, \vec{x}, \vec{X}))$$

where r is an $\mathcal{L}_A^2$ term and ψ is a $\Sigma_0^B(\mathcal{L}_A^2)$ formula. Then $\mathcal{T}$ proves

$$\varphi^+ \;\leftrightarrow\; (s < r(\vec{t}, \vec{T}) \wedge \psi(s, \vec{t}, \vec{T}))$$

The RHS has nesting depth at most that of φ^+ and $\vec{t}, \vec{T}$ have smaller nesting depth, and hence we have reduced the induction step to the case that φ^+ has the form $\rho(\vec{s})$ where $\rho(\vec{x})$ is an atomic formula over $\mathcal{L}_A^2$ and each term s_i has one of the forms $f(\vec{t}, \vec{T})$, for f not in $\mathcal{L}_A^2$, or $|F(\vec{t}, \vec{T})|$. In either case, using the defining axiom for f or Exercise 5.33, for each term s_i there is a $\Sigma_0^B(\mathcal{L}_A^2)$ formula $\eta_i(z, \vec{x}, \vec{X})$ and a bounding term $r_i(\vec{x}, \vec{X})$ of $\mathcal{L}_A^2$ such that $\mathcal{T}$ proves

$$z = s_i \leftrightarrow (z < r_i(\vec{t}, \vec{T}) \wedge \eta_i(z, \vec{t}, \vec{T}))$$

Hence (since φ^+ is $\rho(\vec{s})$), $\mathcal{T}$ proves

$$\varphi^+ \leftrightarrow \exists \vec{z} < \vec{r}(\vec{t}, \vec{T})\big(\rho(\vec{z}) \wedge \bigwedge_i \eta_i(z_i, \vec{t}, \vec{T})\big)$$

Thus we have reduced the nesting depth of φ^+, and we can apply the induction hypothesis. ⊣

The following result is immediate from the preceding lemma, Definitions 5.37 and 5.15, and the Σ_0^B Representation Theorem 4.18.

Corollary 5.41 ($\mathbf{FAC}^0$ Closed under Σ_0^B-Definability). *Every function Σ_0^B-definable from a collection of $\mathbf{FAC}^0$ functions is in $\mathbf{FAC}^0$.*

Below we give Σ_0^B-bit-definitions of the string functions $\varnothing$ (zero, or empty string), $S(X)$ (successor), $X+Y$ and several other useful $\mathbf{AC}^0$ functions: *Row*, *seq*, *left* and *right*. Each of these functions is Σ_0^B-definable in $\mathbf{V}^0$, and the above lemmas and corollaries apply.

EXAMPLE 5.42 ($\varnothing, S, +$). *The string constant $\varnothing$ has bit defining axiom*

$$\varnothing(z) \leftrightarrow z < 0$$

Binary successor $S(X)$ has bit-defining axiom

$$S(X)(i) \;\leftrightarrow\; \big(i \le |X| \wedge ((X(i) \wedge \exists j < i \neg X(j)) \vee (\neg X(i) \wedge \forall j < i X(j)))\big)$$

Recall from (55) that binary addition $X+Y$ has the following bit-defining axiom:

$$(X+Y)(i) \leftrightarrow \big(i < |X|+|Y| \;\wedge\; (X(i) \oplus Y(i) \oplus \mathit{Carry}(i,X,Y))\big)$$

where $\oplus$ is exclusive OR, and

$$\mathit{Carry}(i,X,Y) \equiv \exists k < i\big(X(k) \wedge Y(k) \wedge \forall j < i(k < j \supset (X(j) \vee Y(j)))\big)$$

EXERCISE 5.43. *Show that $\mathbf{V}^0$ proves*

$$\neg\mathit{Carry}(0,X,Y) \wedge$$
$$\mathit{Carry}(i+1,X,Y) \leftrightarrow MAJ(\mathit{Carry}(i,X,Y), X(i), Y(i))$$

where the Boolean function $MAJ(P,Q,R)$ holds iff at least two of P,Q,R are true. This formula gives a recursive definition of Carry which is the binary analog to the school method for computing carries in decimal addition.

EXERCISE 5.44. *Let $\mathbf{V}^0(\varnothing,S,+)$ be $\mathbf{V}^0$ extended by $\varnothing,S,+$ and their bit-defining axioms. Show that the following are theorems of $\mathbf{V}^0(\varnothing,S,+)$:*

(a) $X+\varnothing = X$
(b) $X+S(Y) = S(X+Y)$ *Use the previous exercise, and the fact that in computing the successor of a binary number the lowest order 0 turns to 1, the 1's to the right turn to 0's, and the other bits remain the same. Compare the positions of this lowest order 0 in X and in $X+Y$.*
(c) $X+Y = Y+X$ *(Commutativity).*
(d) $(X+Y)+Z = X+(Y+Z)$ *(Associativity).*
For Associativity, first show in $\mathbf{V}^0(+)$ that

$$\mathit{Carry}(i,Y,Z) \oplus \mathit{Carry}(i,X,Y+Z) \leftrightarrow$$
$$\mathit{Carry}(i,X,Y) \oplus \mathit{Carry}(i,X+Y,Z)$$

Derive a stronger statement than this, and prove it by induction on i.

EXAMPLE 5.45 (The Pairing Function). *We define the pairing function $\langle x, y\rangle$ as the following term of $\mathbf{I\Delta}_0$:*

$$\langle x, y\rangle =_{\text{def}} (x+y)(x+y+1)+2y \tag{69}$$

EXERCISE 5.46. *Show using results in Section 3A that $\mathbf{I\Delta}_0$ proves $\langle x, y\rangle$ is a one-one function. That is*

$$\mathbf{I\Delta}_0 \vdash \ \langle x_1, y_1\rangle = \langle x_2, y_2\rangle \supset x_1 = x_2 \wedge y_1 = y_2 \tag{70}$$

(First show that the LHS implies $x_1 + y_1 = x_2 + y_2$.)

In general we can "pair" more than 2 numbers, e.g., define

$$\langle x_1, \dots, x_{k+1}\rangle = \langle\langle x_1, \dots, x_k\rangle, x_{k+1}\rangle$$

We will refer to the term $\langle x_1, \dots, x_{k+1}\rangle$ as *a tupling function.*

For any constant $k \in \mathbb{N}$, $k \geq 2$, we can use the tupling function to code a k-dimensional bit array by a single string Z by defining

Notation.

$$Z(x_1, \dots, x_k) =_{\text{def}} Z(\langle x_1, \dots, x_k\rangle) \tag{71}$$

EXAMPLE 5.47 (The Projection Functions). *Consider the (partial) projection functions:*

$$y = \mathit{left}(x) \leftrightarrow \exists z \leq x\, (x = \langle y, z\rangle) \qquad z = \mathit{right}(x) \leftrightarrow \exists y \leq x\, (x = \langle y, z\rangle)$$

To make these functions total, we define

$$\mathit{left}(x) = \mathit{right}(x) = 0 \qquad \text{if } \neg \mathit{Pair}(x)$$

where

$$\mathit{Pair}(x) \equiv \exists y \leq x \exists z \leq x\, (x = \langle y, z\rangle)$$

For constants n and $k \leq n$, if x codes an n-tuple, then the k-th component $\langle x\rangle^n_k$ of x can be extracted using left and right, e.g.,

$$\langle x\rangle^3_2 = \mathit{right}(\mathit{left}(x))$$

EXERCISE 5.48. *Use Exercise 5.46 to show that $\mathit{left}(x)$ and $\mathit{right}(x)$ are Σ^B_0-definable in $\mathbf{I\Delta}_0$. Show that $\mathbf{I\Delta}_0(\mathit{left},\mathit{right})$ proves the following properties of Pair and the projection functions:*

(a) $\forall y \forall z \mathit{Pair}(\langle y, z\rangle)$

(b) $\forall x(\mathit{Pair}(x) \supset x = \langle \mathit{left}(x), \mathit{right}(x)\rangle)$

(c) $x = \langle x_1, x_2\rangle \supset (x_1 = \mathit{left}(x) \wedge x_2 = \mathit{right}(x))$

Now we can generalize the Σ^B_0-comprehension axiom scheme to multiple dimensions.

DEFINITION 5.49 (Multiple Comprehension Axiom). *If Φ is a set of formulas, then the* multiple comprehension axiom scheme *for Φ, denoted by Φ-***MULTICOMP***, is the set of all formulas*

(72)
$$\exists X \leq \langle y_1, \dots, y_k\rangle \forall z_1 < y_1 \dots \forall z_k < y_k (X(z_1, \dots, z_k) \leftrightarrow \varphi(z_1, \dots, z_k))$$

where $k \geq 2$ and $\varphi(z)$ is any formula in Φ which may contain other free variables, but not X.

LEMMA 5.50 (Multiple Comprehension). *Suppose that $\mathcal{T} \supseteq \mathbf{V}^0$ is a theory with vocabulary $\mathcal{L}$ which proves the $\Sigma_0^B(\mathcal{L})$-***COMP** *axioms. Then $\mathcal{T}$ proves the $\Sigma_0^B(\mathcal{L})$-***MULTICOMP** *axioms.*

PROOF. For the case $\mathcal{L} = \mathcal{L}_A^2$ we could work in the conservative extension $\mathcal{T}(\mathit{left},\mathit{right})$ and apply Lemma 5.40 to prove this. However for general $\mathcal{L}$ we use another method.

For simplicity we prove the case $k = 2$. Define $\psi(z)$ by

$$\psi(z) \equiv \exists z_1 \leq z \exists z_2 \leq z (z = \langle z_1, z_2\rangle \wedge \varphi(z_1, z_2))$$

Now by Σ_0^B-**COMP**,

$$\mathcal{T} \vdash \exists X \leq \langle y_1, y_2\rangle \forall z < \langle y_1, y_2\rangle (X(z) \leftrightarrow \psi(z))$$

By Exercise 5.46, $\mathcal{T}$ proves that such X satisfies (72). $\dashv$

Notice that the string X in (72) is not unique, because there are numbers $z < \langle y_1, \dots, y_k\rangle$ which are not of the form $\langle z_1, \dots, z_k\rangle$ (the pairing function (69) is not surjective). This, however, is not important, since we will be using only the truth values of $X(z)$ where $z = \langle z_1, \dots, z_k\rangle$ for $z_i < y_i$, $1 \leq i \leq k$. (A unique such X can be defined as in the proof above.)

Now we introduce the string function $\mathit{Row}(x, Z)$ (or $Z^{[x]}$) in $\mathbf{FAC}^0$ to represent row x of the binary array Z.

DEFINITION 5.51 (Row and $\mathbf{V}^0(\mathit{Row})$). *The function $\mathit{Row}(x, Z)$ (also denoted $Z^{[x]}$) has the bit-defining axiom*

$$\mathit{Row}(x, Z)(i) \leftrightarrow (i < |Z| \wedge Z(x, i)) \tag{73}$$

$\mathbf{V}^0(\mathit{Row})$ is the extension of $\mathbf{V}^0$ obtained from $\mathbf{V}^0$ by adding to it the string function Row and its Σ_0^B-bit-definition (73).

Note that by Corollary 5.39, $\mathbf{V}^0(\mathit{Row})$ is a conservative extension of $\mathbf{V}^0$.

The next result follows immediately from Lemma 5.40.

LEMMA 5.52 (Row Elimination Lemma). *For every $\Sigma_0^B(\mathit{Row})$ formula φ, there is Σ_0^B formula φ' such that $\mathbf{V}^0(\mathit{Row}) \vdash \varphi \leftrightarrow \varphi'$. Hence $\mathbf{V}^0(\mathit{Row})$ proves the $\Sigma_0^B(\mathit{Row})$-***COMP** *axiom scheme.*

We can use Row to represent a tuple $X_1, \dots, X_k$ of strings by a single string Z, where $X_i = Z^{[i]}$. The following result follows immediately from the Multiple Comprehension Lemma.

LEMMA 5.53. *$\mathbf{V}^0(Row)$ proves*

$$\forall X_1...\forall X_k \exists Z \le t(X_1 = Z^{[1]} \wedge ... \wedge X_k = Z^{[k]}) \tag{74}$$

where $t = \langle k, |X_1| + ... + |X_k| \rangle$. □

DEFINITION 5.54. *A single-$\Sigma_1^B(\mathcal{L})$ formula is one of the form $\exists X \le t\varphi$, where φ is $\Sigma_0^B(\mathcal{L})$.*

EXERCISE 5.55. *Let $\mathcal{T}$ be a polynomial-bounded theory with vocabulary $\mathcal{L}$ such that $\mathcal{T}$ extends $\mathbf{V}^0(Row)$. Prove that for every $\Sigma_1^B(\mathcal{L})$ formula φ there is a single-$\Sigma_1^B(\mathcal{L})$ formula φ' such that $\mathcal{T} \vdash \varphi \leftrightarrow \varphi'$.*

Now use Lemma 5.52 to show that the same is true when $\mathcal{T}$ is $\mathbf{V}^0$ and $\mathcal{L}$ is $\mathcal{L}_A^2$.

Just as we use a "two-dimensional" string $Z(x, y)$ to code a sequence $Z^{[0]}$, $Z^{[1]}$, ... of strings, we use a similar idea to allow Z to code a sequence $y_0, y_1, \ldots$ of numbers. Now y_i is the smallest element of $Z^{[i]}$, or $|Z|$ if $Z^{[i]}$ is empty. We define an $\mathbf{AC}^0$ function $seq(i, Z)$ (also denoted $(Z)^i$) to extract y_i.

DEFINITION 5.56 (Coding a Bounded Sequence of Numbers). *The number function $seq(x, Z)$ (also denoted $(Z)^x$) has the defining axiom:*

$$y = seq(x, Z) \leftrightarrow$$
$$(y < |Z| \wedge Z(x, y) \wedge \forall z < y \neg Z(x, z)) \ \vee \ (\forall z < |Z| \neg Z(x, z) \wedge y = |Z|)$$

It is easy to check that $\mathbf{V}^0$ proves the existence and uniqueness of y satisfying the RHS of the above formula, and hence seq is Σ_0^B-definable in $\mathbf{V}^0$. As in the case of Row, it follows from Lemma 5.40 that any $\Sigma_0^B(seq)$ formula is provably equivalent in $\mathbf{V}^0(seq)$ to a $\Sigma_0^B(\mathcal{L}_A^2)$ formula. (See also the $\mathbf{AC}^0$ Elimination Lemma 5.74 for a more general result.)

5D.1. Δ_1^1-Definable Predicates. Recall the notion of a Φ-definable (or Φ-representable) predicate symbol, where Φ is a class of formulas (Definition 3.27). Recall also that we obtain a conservative extension of a theory $\mathcal{T}$ by adding to it a definable predicate symbol P and its defining axiom. Below we define the notions of a "$\Delta_1^1(\mathcal{L})$-definable predicate symbol" and a "$\Delta_1^B(\mathcal{L})$-definable predicate symbol". Note that here $\Delta_1^1(\mathcal{L})$ and $\Delta_1^B(\mathcal{L})$ depend on the theory $\mathcal{T}$, in contrast to Definition 3.27.

DEFINITION 5.57 ($\Delta_1^1(\mathcal{L})$ and $\Delta_1^B(\mathcal{L})$ Definable Predicate). *Let $\mathcal{T}$ be a theory over the vocabulary $\mathcal{L}$ and P a predicate symbol not in $\mathcal{L}$. We say that P is $\Delta_1^1(\mathcal{L})$-definable (or simply Δ_1^1-definable) in $\mathcal{T}$ if there are $\Sigma_1^1(\mathcal{L})$ formulas $\varphi(\vec{x}, \vec{Y})$ and $\psi(\vec{x}, \vec{Y})$ such that*

$$R(\vec{x}, \vec{Y}) \leftrightarrow \varphi(\vec{x}, \vec{Y}), \qquad \text{and} \qquad \mathcal{T} \vdash \varphi(\vec{x}, \vec{Y}) \leftrightarrow \neg\psi(\vec{x}, \vec{Y}). \tag{75}$$

We say that P is $\Delta_1^B(\mathcal{L})$-definable (or simply Δ_1^B-definable) in $\mathcal{T}$ if the formulas φ and ψ above are Σ_1^B formulas.

The following exercise can be proved using Parikh's Theorem.

EXERCISE 5.58. *Show that if $\mathcal{T}$ is a polynomial-bounded theory, then a predicate is Δ_1^1-definable in $\mathcal{T}$ iff it is Δ_1^B-definable in $\mathcal{T}$.*

DEFINITION 5.59 (Characteristic Function). *The* characteristic function *of a relation $R(\vec{x}, \vec{X})$, denoted by $f_R(\vec{x}, \vec{X})$, is defined as follows:*

$$f_R(\vec{x}, \vec{X}) = \begin{cases} 1 & \text{if } R(\vec{x}, \vec{X}) \\ 0 & \text{otherwise} \end{cases}$$

We will show that $\mathbf{FAC}^0$ coincides with the class of provably total functions in $\mathbf{V}^0$. It follows that $\mathbf{AC}^0$ relations are precisely the Δ_1^1 definable relations in $\mathbf{V}^0$. More generally we have the following theorem.

THEOREM 5.60. *If the language of a theory $\mathcal{T}$ includes $\mathcal{L}_A^2$, and a complexity class $\mathbf{C}$ has the property that for all relations R, $R \in \mathbf{C}$ iff $f_R \in \mathbf{FC}$, and the class of Σ_1^1-definable functions in $\mathcal{T}$ coincides with $\mathbf{FC}$, then the class of Δ_1^1-definable relations in $\mathcal{T}$ coincides with $\mathbf{C}$.*

PROOF. Assume the hypotheses of the theorem, and suppose that the relation $R(\vec{x}, \vec{X})$ is Δ_1^1-definable in $\mathcal{T}$. Then there are Σ_0^B formulas φ and ψ such that

$$R(\vec{x}, \vec{X}) \leftrightarrow \exists \vec{Y}\, \varphi(\vec{x}, \vec{X}, \vec{Y})$$

and

$$\mathcal{T} \vdash \exists \vec{Y}\, \varphi(\vec{x}, \vec{X}, \vec{Y}) \leftrightarrow \neg \exists \vec{Y}\, \psi(\vec{x}, \vec{X}, \vec{Y}) \tag{76}$$

Thus the characteristic function $f_R(\vec{x}, \vec{X})$ of R satisfies

$$y = f_R(\vec{x}, \vec{X}) \leftrightarrow \theta(y, \vec{x}, \vec{X}) \tag{77}$$

where

$$\theta(y, \vec{x}, \vec{X}) \equiv \exists \vec{Y}((y = 1 \wedge \varphi(\vec{x}, \vec{X}, \vec{Y})) \vee (y = 0 \wedge \psi(\vec{x}, \vec{X}, \vec{Y})))$$

Then $\mathcal{T}$ proves $\exists! y \theta(y, \vec{x}, \vec{X})$, where the existence of y and $\vec{Y}$ follows from the $\leftarrow$ direction of (76) and the uniqueness of y follows from the $\rightarrow$ direction of (76). Thus f_R is Σ_1^1-definable in $\mathcal{T}$, so f_R is in $\mathbf{FC}$, and therefore R is in $\mathbf{C}$.

Conversely, suppose that $R(\vec{x}, \vec{X})$ is in $\mathbf{C}$, so f_R is in $\mathbf{FC}$. Then f_R is Σ_1^1-definable in $\mathcal{T}$, so there is a Σ_1^1 formula $\theta(y, \vec{x}, \vec{X})$ such that (77) holds and

$$\mathcal{T} \vdash \exists! y \theta(y, \vec{x}, \vec{X})$$

Then $R(\vec{x}, \vec{X}) \leftrightarrow \exists y(y \neq 0 \wedge \theta(y, \vec{x}, \vec{X}))$ and

$$\mathcal{T} \vdash \quad \exists y(y \neq 0 \wedge \theta(y, \vec{x}, \vec{X})) \leftrightarrow \neg \theta(0, \vec{x}, \vec{X})$$

Since $\exists y(y \neq 0 \wedge \theta(y, \vec{x}, \vec{X}))$ is equivalent to a Σ_1^1 formula, it follows that R is Δ_1^1-definable in $\mathcal{T}$. $\dashv$

5E. The Witnessing Theorem for $\mathbf{V}^0$

Notation. For a theory $\mathcal{T}$ and a list $\mathcal{L}$ of functions that are definable/bit-definable in $\mathcal{T}$, we denote by $\mathcal{T}(\mathcal{L})$ the theory $\mathcal{T}$ extended by the defining/bit-defining axioms for the symbols in $\mathcal{L}$.

Recall that number functions in $\mathbf{FAC}^0$ are Σ_0^B-definable in $\mathbf{V}^0$, and string functions in $\mathbf{FAC}^0$ are Σ_0^B-bit-definable in $\mathbf{V}^0$ (see Proposition 5.32 and Corollary 5.36). It follows from Corollary 5.39 that $\mathbf{V}^0(\mathcal{L})$ is a conservative extension of $\mathbf{V}^0$, for any collection $\mathcal{L}$ of $\mathbf{FAC}^0$ functions.

Our goal now is to prove the following theorem.

Theorem 5.61 (Witnessing Theorem for $\mathbf{V}^0$). *Suppose that $\varphi(\vec{x},\vec{y},\vec{X},\vec{Y})$ is a Σ_0^B formula such that*

$$\mathbf{V}^0 \vdash \forall\vec{x}\forall\vec{X}\exists\vec{y}\exists\vec{Y}\ \varphi(\vec{x},\vec{y},\vec{X},\vec{Y})$$

Then there are $\mathbf{FAC}^0$ functions $f_1,\ldots,f_k,F_1,\ldots,F_m$ so that

$$\mathbf{V}^0(f_1,\ldots,f_k,F_1,\ldots,F_m) \vdash \forall\vec{x}\forall\vec{X}\varphi(\vec{x},\vec{f}(\vec{x},\vec{X}),\vec{X},\vec{F}(\vec{x},\vec{X}))$$

The functions f_i and F_j are called the *witnessing functions*, for y_i and Y_j, respectively.

We will prove the Witnessing Theorem for $\mathbf{V}^0$ in the next section. First, we list some of its corollaries.

The next corollary follows from the above theorem and Corollary 5.36.

Corollary 5.62 (Σ_1^1-Definability Theorem for $\mathbf{V}^0$). *A function is in $\mathbf{FAC}^0$ iff it is Σ_1^1-definable in $\mathbf{V}^0$ iff it is Σ_1^B-definable in $\mathbf{V}^0$ iff it is Σ_0^B-definable in $\mathbf{V}^0$.*

Corollary 5.63. *A relation is in $\mathbf{AC}^0$ iff it is Δ_1^1 definable in $\mathbf{V}^0$ iff it is Δ_1^B definable in $\mathbf{V}^0$.*

It follows from the Σ_0^B-Representation Theorem 4.18 that a relation is in $\mathbf{AC}^0$ iff its characteristic function is in $\mathbf{AC}^0$. Therefore Corollary 5.63 follows from the Σ_1^1-Definability Theorem for $\mathbf{V}^0$ and Theorem 5.60. Alternatively, it can be proved using the Witnessing Theorem for $\mathbf{V}^0$ as follows.

Proof. Since each $\mathbf{AC}^0$ relation R is represented by a Σ_0^B formula θ, it is obvious that they are Δ_1^B (and hence Δ_1^1) definable in $\mathbf{V}^0$: In (75) simply let φ be θ, and ψ be $\neg\theta$.

On the other hand, suppose that R is a Δ_1^1-definable relation of $\mathbf{V}^0$. In other words, there are Σ_0^B formulas $\varphi(\vec{x},\vec{X},\vec{Y})$ and $\psi(\vec{x},\vec{X},\vec{Y})$ so that

$$R(\vec{x},\vec{X}) \leftrightarrow \exists\vec{Y}\varphi(\vec{x},\vec{X},\vec{Y})$$

and

$$\mathbf{V}^0 \vdash \exists\vec{Y}\varphi(\vec{x},\vec{X},\vec{Y}) \leftrightarrow \neg\exists\vec{Y}\psi(\vec{x},\vec{X},\vec{Y}) \tag{78}$$

In particular,

$$\mathbf{V}^0 \vdash \exists \vec{Y}(\varphi(\vec{x}, \vec{X}, \vec{Y}) \vee \psi(\vec{x}, \vec{X}, \vec{Y}))$$

By the Witnessing Theorem for $\mathbf{V}^0$, there are $\mathbf{AC}^0$ functions $F_1, \ldots, F_k$ so that

$$\text{(79)} \quad \mathbf{V}^0(F_1, \ldots, F_k) \vdash \forall \vec{x} \forall \vec{X}(\varphi(\vec{x}, \vec{X}, \vec{F}(\vec{x}, \vec{X})) \vee \psi(\vec{x}, \vec{X}, \vec{F}(\vec{x}, \vec{X})))$$

We claim that $\mathbf{V}^0(F_1, \ldots, F_k)$ proves

$$\forall \vec{x} \forall \vec{X}(\exists \vec{Y} \varphi(\vec{x}, \vec{X}, \vec{Y}) \leftrightarrow \varphi(\vec{x}, \vec{X}, \vec{F}(\vec{x}, \vec{X})))$$

The $\leftarrow$ direction is trivial. The other direction follows from (78) and (79).

Consequently $\varphi(\vec{x}, \vec{X}, \vec{F}(\vec{x}, \vec{X}))$ also represents $R(\vec{x}, \vec{X})$. Here R is obtained from the relation represented by $\varphi(\vec{x}, \vec{X}, \vec{Y})$ by substituting the $\mathbf{AC}^0$ functions $\vec{F}$ for $\vec{Y}$. By Theorem 5.19 (a), R is also an $\mathbf{AC}^0$ relation. $\dashv$

5E.1. Independence follows from the Witnessing Theorem for $\mathbf{V}^0$. We can use the Witnessing Theorem to show the unprovability in $\mathbf{V}^0$ of $\exists Z\ \varphi(Z)$ by showing that no $\mathbf{AC}^0$ function can witness the quantifier $\exists Z$. Recall that the relation $PARITY(X)$ is defined by

$$PARITY(X) \leftrightarrow \text{ the set } X \text{ has an odd number of elements}$$

Then a well known result in complexity theory states:

Proposition 5.64. *$PARITY \notin \mathbf{AC}^0$.*

First, it follows that the characteristic function $parity(X)$ of $PARITY(X)$ is not in $\mathbf{FAC}^0$. Therefore *parity* is not Σ_1^1-definable in $\mathbf{V}^0$. In the next chapter we will show that *parity* is Σ_1^1-definable in the theory $\mathbf{V}^1$. This will show that $\mathbf{V}^0$ is a proper sub-theory of $\mathbf{V}^1$.

Now consider the Σ_0^B formula $\varphi_{parity}(X, Y)$:

$$\text{(80)} \qquad \neg Y(0) \wedge \forall i < |X|(Y(i+1) \leftrightarrow (X(i) \oplus Y(i)))$$

where $\oplus$ is exclusive OR. Thus $\varphi_{parity}(X, Y)$ asserts that for $0 \leq i < |X|$, bit $Y(i+1)$ is 1 iff the number of 1's among bits $X(0), ..., X(i)$ is odd. Define

$$\varphi(X) \equiv \exists Y \leq (|X|+1)\ \varphi_{parity}(X, Y)$$

Then $\forall X \varphi(X)$ is true in the standard model $\underline{\mathbb{N}}_2$, but by the above proposition, no function $F(X)$ satisfying $\forall X \varphi_{parity}(X, F(X))$ can be in $\mathbf{FAC}^0$. Hence by the Witnessing Theorem for $\mathbf{V}^0$,

$$\mathbf{V}^0 \nvdash \forall X \exists Y \leq (|X|+1)\ \varphi_{parity}(X, Y)$$

Note that this independence result does not follow from Parikh's Theorem.

5E.2. Proof of the Witnessing Theorem for $\mathbf{V}^0$. Recall the analogous statement in single-sorted logic for $\mathbf{I\Delta}_0$ (i.e., that a Σ_1 theorem of $\mathbf{I\Delta}_0$ can be "witnessed" by a single-sorted **LTH** function) which is proved in Theorem 3.64. There we use the Bounded Definability Theorem 3.33 (which follows from Parikh's Theorem) to show that the graph of any Σ_1-definable function of $\mathbf{I\Delta}_0$ is actually definable by a Δ_0 formula, and hence an **LTH** relation.

Unfortunately, a similar method does not work here. We can also use Parikh's Theorem to show that the graph of a Σ_1^1-definable function of $\mathbf{V}^0$ is representable by a Σ_1^B formula. However this does not suffice, since there are string functions whose graphs are in $\mathbf{AC}^0$ (i.e., representable by Σ_0^B formulas), but which do not belong to $\mathbf{FAC}^0$. An example is the counting function whose graph is given by the Σ_0^B formula $\delta_{NUM}(x, X, Y)$ (224).

Our first proof is by the Anchored $\mathbf{LK}^2$ Completeness Theorem 4.30. This proof is important because the same method can be used to prove the witnessing theorem for $\mathbf{V}^1$ (Theorem 6.28). Our second proof method (see Section 5F.1) is based on the Herbrand Theorem and does not work for $\mathbf{V}^1$.

We will prove the following simple form of the theorem, since it implies the general form.

Lemma 5.65. *Suppose that $\varphi(\vec{x}, \vec{X}, Y)$ is a Σ_0^B formula such that*

$$\mathbf{V}^0 \vdash \forall \vec{x} \forall \vec{X} \exists Z \varphi(\vec{x}, \vec{X}, Z)$$

Then there is an $\mathbf{FAC}^0$ function F so that

$$\mathbf{V}^0(F) \vdash \forall \vec{x} \forall \vec{X} \varphi(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

Proof of Theorem 5.61 from Lemma 5.65. The idea is to use the function *Row* to encode the tuple $\langle \vec{y}, \vec{Y} \rangle$ by a single string variable Z, as in Lemma 5.53. Then by the above lemma, Z is witnessed by an $\mathbf{AC}^0$ function F. The witnessing functions for $y_1, \dots, y_k, Y_1, \dots, Y_m$ will then be extracted from F using the function *Row*. Details are as follows.

Assume the hypothesis of the Witnessing Theorem for $\mathbf{V}^0$, i.e.,

$$\mathbf{V}^0 \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \exists \vec{Y} \; \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$$

for a Σ_0^B formula $\varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$. Then since $\mathbf{V}^0(Row)$ extends $\mathbf{V}^0$, we have also

$$\mathbf{V}^0(Row) \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \exists \vec{Y} \; \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$$

Note that

$$\mathbf{V}^0(Row) \vdash \forall y_1 \dots \forall y_k \forall Y_1 \dots \forall Y_m \exists Z \big(\bigwedge_{i=1}^{k} |Z^{[i]}| = y_i \wedge \bigwedge_{j=1}^{m} Z^{[k+j]} = Y_j \big)$$

(See also Lemma 5.53.) Thus

$$\mathbf{V}^0(Row) \vdash \forall\vec{x}\forall\vec{X}\exists Z\ \varphi(\vec{x}, |Z^{[1]}|, \dots, |Z^{[k]}|, \vec{X}, Z^{[k+1]}, \dots, Z^{[k+m]})$$

i.e.,

$$\mathbf{V}^0(Row) \vdash \forall\vec{x}\forall\vec{X}\exists Z\psi(\vec{x}, \vec{X}, Z)$$

where

$$\psi(\vec{x}, \vec{X}, Z) \equiv \varphi(\vec{x}, |Z^{[1]}|, \dots, |Z^{[k]}|, \vec{X}, Z^{[k+1]}, \dots, Z^{[k+m]})$$

is a $\Sigma_0^B(\mathcal{L}_A^2 \cup \{Row\})$ formula.

Now by Lemma 5.52, there is a $\Sigma_0^B(\mathcal{L}_A^2)$ formula $\psi'(\vec{x}, \vec{X}, Z)$ so that

$$\mathbf{V}^0(Row) \vdash \forall\vec{x}\forall\vec{X}\forall Z(\psi(\vec{x}, \vec{X}, Z) \leftrightarrow \psi'(\vec{x}, \vec{X}, Z))$$

As a result, since $\mathbf{V}^0(Row)$ is conservative over $\mathbf{V}^0$, we also have

$$\mathbf{V}^0 \vdash \forall\vec{x}\forall\vec{X}\exists Z\psi'(\vec{x}, \vec{X}, Z)$$

Applying Lemma 5.65, there is an $\mathbf{AC}^0$ function F so that

$$\mathbf{V}^0(F) \vdash \forall\vec{x}\forall\vec{X}\psi'(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

Therefore

$$\mathbf{V}^0(Row, F) \vdash \forall\vec{x}\forall\vec{X}\psi(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

i.e.,

$$\mathbf{V}^0(Row, F) \vdash \forall\vec{x}\forall\vec{X}\ \varphi(\vec{x}, |F^{[1]}|, \dots, |F^{[k]}|, \vec{X}, F^{[k+1]}, \dots, F^{[k+m]})$$

where we write F for $F(\vec{x}, \vec{X})$.

Let $f_i(\vec{x}, \vec{X}) = |(F(\vec{x}, \vec{X}))^{[i]}|$ for $1 \le i \le k$ and $F_j(\vec{x}, \vec{X}) = (F(\vec{x}, \vec{X}))^{[k+j]}$ for $1 \le j \le m$ and denote $\{f_1, \dots, f_k, F_1, \dots, F_m\}$ by $\mathcal{L}$, we have

$$\mathbf{V}^0(\{Row, F\} \cup \mathcal{L}) \vdash \forall\vec{x}\forall\vec{X}\ \varphi(\vec{x}, \vec{f}, \vec{X}, \vec{F})$$

By Corollary 5.39, $\mathbf{V}^0(\{Row, F\} \cup \mathcal{L})$ is a conservative extension of $\mathbf{V}^0(\mathcal{L})$. Consequently,

$$\mathbf{V}^0(\mathcal{L}) \vdash \forall\vec{x}\forall\vec{X}\ \varphi(\vec{x}, \vec{f}, \vec{X}, \vec{F})$$

$\dashv$

The rest of this section is devoted to the proof of Lemma 5.65.

Proof of Lemma 5.65. The proof method is similar to that of Lemma 5.25 (for Parikh's Theorem). Suppose that $\exists Z\varphi(\vec{a}, \vec{\alpha}, Z)$ is a theorem of $\mathbf{V}^0$. By the Anchored $\mathbf{LK}^2$ Completeness Theorem, there is an anchored $\mathbf{LK}^2$-$\mathbf{T}$ proof π of

$$\longrightarrow \exists Z\varphi(\vec{a}, \vec{\alpha}, Z)$$

where $\mathbf{T}$ is the set of all term substitution instances of the axioms for $\mathbf{V}^0$. We assume that π is in free variable normal form (see Section 4D.1).

Note that all instances of the Σ_0^B-$\mathbf{COMP}$ axioms (48) are Σ_1^1 formulas (they are in fact Σ_1^B formulas). Since the endsequent of π is also a Σ_1^1 formula, by the Subformula Property (Theorem 4.31), all formulas in π are Σ_1^1 formulas, and in fact they contain at most one string quantifier $\exists X$ in front. In particular, every sequent in π has the form

(81)
$$\exists X_1\theta_1(X_1), \ldots, \exists X_m\theta_m(X_m), \Gamma \longrightarrow \Delta, \exists Y_1\psi_1(Y_1), \ldots, \exists Y_n\psi_n(Y_n)$$

for $m, n \geq 0$, where θ_i and ψ_j and all formulas in Γ and Δ are Σ_0^B.

We will prove by induction on the depth in π of a sequent $\mathcal{S}$ of the form (81) that there are Σ_0^B-bit-definable string functions $F_1, ..., F_n$ (i.e., the witnessing functions) such that there is a collection of Σ_0^B-bit-definable functions $\mathcal{L}$ including $F_1, ..., F_n$ and an $\mathbf{LK}^2$-$\mathbf{V}^0(\mathcal{L})$ proof of

(82) $$\mathcal{S}' =_{\text{def}} \theta_1(\beta_1), \ldots, \theta_m(\beta_m), \Gamma \longrightarrow \Delta, \psi_1(F_1), \ldots, \psi_n(F_n)$$

where F_i stands for $F_i(\vec{a}, \vec{\alpha}, \vec{\beta})$, and $\vec{a}, \vec{\alpha}$ is a list of exactly those variables with free occurrences in $\mathcal{S}$. (This list may be different for different sequents.) Here $\beta_1, ..., \beta_m$ are distinct new free variables corresponding to the bound variables $X_1, ..., X_m$, although the latter variables may not be distinct.

It follows that for the endsequent $\longrightarrow \exists Z\varphi(\vec{a}, \vec{\alpha}, Z)$ of π, there is a finite collection $\mathcal{L}$ of $\mathbf{FAC}^0$ functions, and an $F \in \mathcal{L}$ so that

$$\mathbf{V}^0(\mathcal{L}) \vdash \varphi(\vec{a}, \vec{\alpha}, F(\vec{a}, \vec{\alpha}))$$

Note that by Corollary 5.39, $\mathbf{V}^0(\mathcal{L})$ is a conservative extension of $\mathbf{V}^0(F)$. Consequently we have

$$\mathbf{V}^0(F) \vdash \varphi(\vec{a}, \vec{\alpha}, F(\vec{a}, \vec{\alpha}))$$

and we are done.

Our inductive proof has several cases, depending on whether $\mathcal{S}$ is a $\mathbf{V}^0$ axiom, or which rule is used to generate $\mathcal{S}$. In each case we will introduce suitable witnessing functions when required, and it is an easy exercise to check that in each of the functions introduced has a $\Sigma_0^B(\mathcal{L}_A^2)$-bit-definition.

To show that the arguments $\vec{a}, \vec{\alpha}$ of previously-introduced witnessing functions continue to include only those variables with free occurrences in the sequent $\mathcal{S}$, we use the fact that the proof π is in free variable normal form, and hence no free variable is eliminated by any rule in the proof except $\forall$-right and $\exists$-left. (We made a similar argument concerning the free variables in the bounding terms t in the proof of Lemma 5.25).

In general we will show that $\mathcal{S}'$ has an $\mathbf{LK}^2$-$\mathbf{V}^0(\mathcal{L})$ proof not by constructing the proof, but rather by arguing that the formula giving the semantics of $\mathcal{S}'$ (Definition 2.35) is provable in $\mathbf{V}^0$ from the bit-defining axioms of the functions $\mathcal{L}$, and invoking the $\mathbf{LK}^2$ Completeness Theorem. However in each case the $\mathbf{LK}^2$-$\mathbf{V}^0(\mathcal{L})$ proof is not hard to find.

Specifically, if we write (82) in the form

$$\mathcal{S}' = \quad A_1, ..., A_k \longrightarrow B_1, ..., B_\ell$$

then we assert

(83) $$\mathbf{V}^0(\mathcal{L}) \vdash \forall \vec{x} \forall \vec{X} \forall \vec{Y} \big((A_1 \wedge ... \wedge A_k) \supset (B_1 \vee ... \vee B_\ell)\big)$$

Case I: S is an axiom of $\mathbf{V}^0$. If the axiom only involves Σ_0^B formulas, then no witnessing functions are needed. Otherwise $\mathcal{S}$ comes from a Σ_0^B-**COMP** axiom, i.e.,

$$\mathcal{S} =_{\text{def}} \longrightarrow \exists X \le b \forall z < b(X(z) \leftrightarrow \psi(z, b, \vec{a}, \vec{\alpha}))$$

Then a function witnessing X has bit-defining axiom

$$F(b, \vec{a}, \vec{\alpha})(z) \leftrightarrow z < b \wedge \psi(z, b, \vec{a}, \vec{\alpha})$$

Case II: $\mathcal{S}$ is obtained by an application of the rule string $\exists$-right. Then $\mathcal{S}$ is the bottom of the inference

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda \longrightarrow \Pi, \psi(T)}{\Lambda \longrightarrow \Pi, \exists X \psi(X)}$$

where the string term T is either a variable γ or the constant $\varnothing$ introduced when putting π in free variable normal form. In the former case, γ must have a free occurrence in $\mathcal{S}$, and we may witness the new quantifier $\exists X$ by the function F with bit-defining axiom

$$F(\vec{a}, \gamma, \vec{\alpha}, \vec{\beta})(z) \leftrightarrow z < |\gamma| \wedge \gamma(z)$$

In the latter case T is $\varnothing$, and we define

$$F(\vec{a}, \vec{\alpha}, \vec{\beta})(z) \leftrightarrow z < 0$$

Case III: $\mathcal{S}$ is obtained by an application of the rule string $\exists$-left. Then $\mathcal{S}$ is the bottom of the inference

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\theta(\gamma), \Lambda \longrightarrow \Pi}{\exists X \theta(X), \Lambda \longrightarrow \Pi}$$

Note that γ cannot occur in $\mathcal{S}$, by the restriction for this rule, but $\mathcal{S}'$ has a new variable β' available corresponding to $\exists X$ (see (82)). No new witnessing function is required. Each witnessing function $F_j(\vec{a}, \gamma, \vec{\alpha}, \vec{\beta})$ for the top sequent is replaced by the witnessing function

$$F'_j(\vec{a}, \vec{\alpha}, \beta', \vec{\beta}) = F_j(\vec{a}, \beta', \vec{\alpha}, \vec{\beta})$$

for $\mathcal{S}'$.

Case IV: $\mathcal{S}$ is obtained by an application of the rule number $\exists$-right or number $\forall$-left. No new witnessing functions are required.

Case V: $\mathcal{S}$ follows from an application of rule number $\exists$-left or number $\forall$-right. We consider number $\exists$-left, since number $\forall$-right is similar. Then $\mathcal{S}$ is the bottom sequent in the inference

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{b \le t \wedge \theta(b), \Lambda \longrightarrow \Pi}{\exists x \le t\theta(x), \Lambda \longrightarrow \Pi}$$

No new witnessing function is needed, but the free variable b is eliminated as an argument to the existing witnessing functions, and it must be given a value. We give it a value which satisfies the new existential quantifier, if one exists. Thus define the $\mathbf{FAC}^0$ number function

$$g(\vec{a}, \vec{\alpha}) = \min b \le t\ \theta(b)$$

For each witnessing function $F_j(b, \vec{a}, \vec{\alpha}, \vec{\beta})$ for the top sequent define the corresponding witnessing function for the bottom sequent by

$$F_j'(\vec{a}, \vec{\alpha}, \vec{\beta}) = F_j(g(\vec{a}, \vec{\alpha}), \vec{a}, \vec{\alpha}, \vec{\beta})$$

Case VI: $\mathcal{S}$ is obtained by the cut rule. Then $\mathcal{S}$ is the bottom of the inference

$$\frac{\mathcal{S}_1 \qquad \mathcal{S}_2}{\mathcal{S}} = \frac{\Lambda \longrightarrow \Pi, \psi \qquad \psi, \Lambda \longrightarrow \Pi}{\Lambda \longrightarrow \Pi}$$

Assume first that ψ is Σ_0^B. For $i = 1, 2$, let $F_1^i(\vec{a}, \vec{\alpha}), \dots, F_n^i(\vec{a}, \vec{\alpha})$ be the witnessing functions for Π in $\mathcal{S}_i'$. Then we define witnessing functions $F_1, \dots, F_n$ for these formulas in the conclusion $\mathcal{S}'$ by the bit-defining axioms

$$F_j(\vec{a}, \vec{\alpha})(z) \leftrightarrow ((\neg\psi \wedge F_j^1(\vec{a}, \vec{\alpha})(z)) \ \vee \ (\psi \wedge F_j^2(\vec{a}, \vec{\alpha})(z)))$$

Now assume that ψ is not Σ_0^B, so ψ has the form

$$\psi \equiv \exists X \theta(X) \tag{84}$$

where $\theta(X)$ is Σ_0^B. Let $G(\vec{a}, \vec{\alpha})$ be the witnessing function for $\exists X$ in S_1' and let β be the variable in S_2' corresponding to X. Let $F_1^1(\vec{a}, \vec{\alpha}), \dots, F_n^1(\vec{a}, \vec{\alpha})$ be the other witnessing functions for Π in $\mathcal{S}_1'$, and $F_1^2(\vec{a}, \vec{\alpha}, \beta), \dots, F_n^2(\vec{a}, \vec{\alpha}, \beta)$ be the witnessing functions for Π in $\mathcal{S}_2'$. The corresponding witnessing function F_j in S' has defining axiom (replace $\dots$ by $\vec{a}, \vec{\alpha}$)

$$F_j(\dots)(z) \leftrightarrow (\neg\theta(G(\dots)) \wedge F_j^1(\dots)(z)) \vee (\theta(G(\dots)) \wedge F_j^2(\dots, G(\dots))(z))$$

EXERCISE 5.66. *Show correctness of this definition of F in the special case where the cut formula ψ has the form* (84), *and* Π *has only one* Σ_1^1 *formula, by arguing that* $\mathbf{V}^0(\mathcal{L})$ *can prove the semantic translation (83) of* $\mathcal{S}'$ *from the semantic translations of* $\mathcal{S}_1'$ *and* $\mathcal{S}_2'$.

Case VII: $\mathcal{S}$ is obtained from an instance of the rule $\wedge$-left or $\vee$-right. These are both handled in the same manner. Consider $\wedge$-right.

$$\frac{\mathcal{S}_1 \qquad \mathcal{S}_2}{\mathcal{S}} = \frac{\Lambda \longrightarrow \Pi, A \qquad \Lambda \longrightarrow \Pi, B}{\Lambda \to \Pi, (A \wedge B)}$$

Here, as in (81),

$$\begin{aligned} &\Lambda =_{\mathrm{def}} \exists X_1\theta_1(X_1), \dots, \exists X_m\theta_m(X_m), \Gamma \\ \text{and} \quad &\Pi =_{\mathrm{def}} \Delta, \exists Y_1\psi_1(Y_1), \dots, \exists Y_n\psi_n(Y_n) \end{aligned}$$

for $m, n \geq 0$, where θ_i and ψ_j and all formulas in Γ and Δ are Σ_0^B. Also, A and B are Σ_0^B formulas.

Let $F_j^1(\vec{a}, \vec{\alpha})$ and $F_j^2(\vec{a}, \vec{\alpha})$ witness Y_j in $\mathcal{S}_1'$ and $\mathcal{S}_2'$, respectively. Then we define the witness $F_j(\vec{a}, \vec{\alpha})$ for Y_j in $\mathcal{S}'$ to be $F_j^1(\vec{a}, \vec{\alpha})$ or $F_j^2(\vec{a}, \vec{\alpha})$, depending on whether $F_j^1(\vec{a}, \vec{\alpha})$ works as a witness. In particular (replace $\dots$ by $\vec{a}, \vec{\alpha}$):

$$F_j(\dots)(z) \leftrightarrow \big((\psi_j(F_j(\dots)) \wedge F_j^1(\dots)(z)) \vee (\neg\psi_j(F_j(\dots)) \wedge F_j^2(\dots)(z))\big)$$

Case VIII: $\mathcal{S}$ is obtained by any of the other rules. Weakening is easy. There is nothing to do for exchange and $\neg$ introduction. The contraction rules can be derived from cut and exchanges. ⊣

Exercise 5.67. *Show that in the* **Cases V**, **VI**, *and* **VII** *above, the new functions introduced have $\Sigma_0^B(\mathcal{L}_A^2)$-bit-definitions.*

5F. $\overline{\mathbf{V}}^0$: Universal Conservative Extension of $\mathbf{V}^0$

Recall that a universal formula is a formula in prenex form in which all quantifiers are universal, and a universal theory is a theory which can be axiomatized by universal formulas. Recall also the universal single-sorted theory $\overline{\mathbf{I}\Delta}_0$ introduced in Section 3C.2.

The universal theory $\overline{\mathbf{V}}^0$ extends $\overline{\mathbf{I}\Delta}_0$, and is defined in the same way as $\overline{\mathbf{I}\Delta}_0$. Here we show that $\overline{\mathbf{V}}^0$ is a conservative extension of $\mathbf{V}^0$, and that this gives us an alternative proof of the Witnessing Theorem for $\mathbf{V}^0$ by applying the Herbrand Theorem 4.33 for $\overline{\mathbf{V}}^0$.

The idea is to introduce number functions with universal defining axioms, and string functions with universal bit-defining axioms, which are provably total in $\mathbf{V}^0$. Thus we obtain a conservative extension of $\mathbf{V}^0$. Furthermore, the new functions are defined in such a way that the axioms of $\mathbf{V}^0$ with existential quantifiers (namely Σ_0^B-**COMP** and **B**12, **SE**) can be proved from other axioms, and hence can be deduced from our set of universal axioms for $\overline{\mathbf{V}}^0$.

We use the following notation. For any formula $\varphi(z, \vec{x}, \vec{X})$ and $\mathcal{L}_A^2$-term $t(\vec{x}, \vec{X})$, let $F_{\varphi(z),t}(\vec{x}, \vec{X})$ be the string function with bit definition

$$F_{\varphi(z),t}(\vec{x}, \vec{X})(z) \leftrightarrow \big(z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X})\big) \tag{85}$$

Also, let $f_{\varphi(z),t}(\vec{x}, \vec{X})$ be the number function defined as in (28) to be the least $y < t$ such that $\varphi(y, \vec{x}, \vec{X})$ holds, or t if no such y exists. Then

$f_{\varphi(z),t}$ has defining axiom (we write f for $f_{\varphi(z),t}$, t for $t(\vec{x},\vec{X})$, and $\dots$ for $\vec{x},\vec{X}$):

(86)

$$f(\dots) \le t \wedge (f(\dots) < t \supset \varphi(f(\dots),\dots)) \wedge (v < f(\dots) \supset \neg\varphi(v,\dots))$$

Recall that the predecessor function pd has the defining axioms:

(87) $\quad$ $\mathbf{B}12'$. $pd(0) = 0$ $\qquad$ $\mathbf{B}12''$. $x \ne 0 \supset pd(x) + 1 = x$

($\mathbf{B}12'$ and $\mathbf{B}12''$ are called respectively $\mathbf{D}1'$ and $\mathbf{D}2''$ in Section 3C.2.)

In two-sorted logic, the extensionality axiom **SE** contains an implicit existential quantifier $\exists i < |X|$. Therefore we introduce the function $f_{\mathbf{SE}}$ with the defining axiom (86), where $\varphi(z, X, Y) \equiv X(z) \not\leftrightarrow Y(z)$, and $t(X, Y) = |X|$. Intuitively, $f_{\mathbf{SE}}(X, Y)$ is the smallest number $< |X|$ that distinguishes X and Y, and $|X|$ if no such number exists.

(88)
$$\begin{aligned} &f_{\mathbf{SE}}(X,Y) \le |X| \ \wedge \\ &f_{\mathbf{SE}}(X,Y) < |X| \supset \neg(X(f_{\mathbf{SE}}(X,Y)) \leftrightarrow Y(f_{\mathbf{SE}}(X,Y))) \ \wedge \\ &z < f_{\mathbf{SE}}(X,Y) \supset (X(z) \leftrightarrow Y(z)). \end{aligned}$$

Let $\mathbf{SE}'$ be the following axiom

(89)
$$(|X| = |Y| \wedge f_{\mathbf{SE}}(X,Y) = |X|) \supset X = Y.$$

The language $\mathcal{L}_{\mathbf{FAC}^0}$ is defined below. It contains a function symbol for every $\mathbf{AC}^0$ function. Note that it extends $\mathcal{L}_{\Delta_0}$ (Definition 3.41).

DEFINITION 5.68. *$\mathcal{L}_{\mathbf{FAC}^0}$ is the smallest set that satisfies*

1) *$\mathcal{L}_{\mathbf{FAC}^0}$ includes $\mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}\}$.*
2) *For each open formula $\varphi(z,\vec{x},\vec{X})$ over $\mathcal{L}_{\mathbf{FAC}^0}$ and term $t = t(\vec{x},\vec{X})$ of $\mathcal{L}_A^2$ there is a string function $F_{\varphi(z),t}$ and a number function $f_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FAC}^0}$.*

DEFINITION 5.69. *$\overline{\mathbf{V}}^0$ is the theory over $\mathcal{L}_{\mathbf{FAC}^0}$ with the following set of axioms:* $\mathbf{B}1$–$\mathbf{B}11$, $\mathbf{L}1$, $\mathbf{L}2$ *(Figure 2),* $\mathbf{B}12'$ *and* $\mathbf{B}12''$ (87), (88), $\mathbf{SE}'$ (89), *and* (85) *for each function $F_{\varphi(z),t}$ and* (86) *for each function $f_{\varphi(z),t}$ of $\mathcal{L}_{\mathbf{FAC}^0}$.*

Thus $\overline{\mathbf{V}}^0$ extends $\overline{\mathbf{I}\Delta}_0$. Also, the axioms for $\overline{\mathbf{V}}^0$ do not include any comprehension axiom. However, we will show that $\overline{\mathbf{V}}^0$ proves the Σ_0^B-**COMP** axiom scheme, and hence $\overline{\mathbf{V}}^0$ extends $\mathbf{V}^0$.

Recall that an open formula is a formula without quantifier. The following lemma can be proved by structural induction on φ in the same way as Lemma 3.44.

LEMMA 5.70. *For every $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$ formula φ there is an open $\mathcal{L}_{\mathbf{FAC}^0}$-formula φ^+ such that $\overline{\mathbf{V}}^0 \vdash \varphi \leftrightarrow \varphi^+$.*

LEMMA 5.71. *$\overline{\mathbf{V}}^0$ proves the $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$-**COMP**, $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$-**IND**, and $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$-**MIN** axiom schemes.*

PROOF. For comprehension, we need to show, for each $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$ formula $\varphi(z, \vec{x}, \vec{X})$,

$$\overline{\mathbf{V}}^0 \vdash \exists Z \le y \forall z < y(Z(z) \leftrightarrow \varphi(z, \vec{x}, \vec{X}))$$

Simply take $Z = F_{\varphi,y}(\vec{x}, \vec{X})$ and apply (85). For induction and minimization we use Corollary 5.8. ⊣

THEOREM 5.72. *The theory* $\overline{\mathbf{V}}^0$ *is a conservative extension of* $\mathbf{V}^0$.

PROOF. To show that $\overline{\mathbf{V}}^0$ extends $\mathbf{V}^0$, we need to verify that $\overline{\mathbf{V}}^0$ proves **B**12, **SE** and Σ_0^B-**COMP**. First, **B**12 follows from **B**12″. We prove **SE** in $\overline{\mathbf{V}}^0$ as follows. Assume that

$$|X| = |Y| \wedge \forall z < |X|(X(z) \leftrightarrow Y(z))$$

Then from (88) we have $f_{\mathbf{SE}}(X, Y) = |X|$. Hence by (89) we obtain $X = Y$.

That $\overline{\mathbf{V}}^0$ proves Σ_0^B-**COMP** follows from Lemma 5.71.

Now we show that $\overline{\mathbf{V}}^0$ is conservative over $\mathbf{V}^0$. Let

$$pd, f_{\mathbf{SE}}, \ldots \tag{90}$$

be an enumeration of $\mathcal{L}_{\mathbf{FAC}^0}$ such that the n-th function is defined or bit-defined by an open formula using only the first $(n-1)$ functions. Let $\mathcal{L}_n$ denote the union of $\mathcal{L}_A^2$ and the set of the first n functions in the enumeration, and $\mathbf{V}^0(\mathcal{L}_n)$ denote $\mathbf{V}^0$ together with the defining axioms or bit-defining axioms for the functions of $\mathcal{L}_n$ $(n \ge 0)$. Then

$$\overline{\mathbf{V}}^0 = \bigcup_{n \ge 0} \mathbf{V}^0(\mathcal{L}_n)$$

First we prove:

Claim. For $n \ge 1$, $\mathbf{V}^0(\mathcal{L}_n)$ satisfies the hypothesis of Lemma 5.40.

From Lemma 5.40 and the claim we have

$$\mathbf{V}^0(\mathcal{L}_n) \vdash \Sigma_0^B(\mathcal{L}_n)\text{-}\mathbf{COMP}$$

Therefore by Corollary 5.39 $\mathbf{V}^0(\mathcal{L}_{n+1})$ is conservative over $\mathbf{V}^0(\mathcal{L}_n)$. Then by Compactness Theorem, it follows that $\overline{\mathbf{V}}^0$ is also conservative over $\mathbf{V}^0$. (See also Corollary 3.31.)

It remains to prove the claim.

First note that $\mathbf{V}^0(\mathcal{L}_n)$ extends $\mathbf{V}^0$ for all $n \ge 1$. Also $\mathcal{L}_{\mathbf{FAC}^0}$ has the same predicates as $\mathcal{L}_A^2$. We will prove by induction on n that each string function in $\mathcal{L}_n$ has a $\Sigma_0^B(\mathcal{L}_A^2)$-bit-defining axiom in $\mathbf{V}^0(\mathcal{L}_n)$, and each number function in $\mathcal{L}_n$ has a $\Sigma_0^B(\mathcal{L}_A^2)$-defining axiom in $\mathbf{V}^0(\mathcal{L}_n)$, and thus establishing the claim.

For the base case, $n = 1$, by **B**12′ and **B**12″ pd has a $\Sigma_0^B(\mathcal{L}_A^2)$-defining axiom in $\mathbf{V}^0$, therefore $\mathbf{V}^0(\mathcal{L}_1)$ (which is $\mathbf{V}^0(pd)$) satisfies the hypothesis of Lemma 5.40.

For the induction step we need to show that the $(n+1)$-st function f_{n+1} or F_{n+1} in (90) has a $\Sigma_0^B(\mathcal{L}_A^2)$-defining axiom or a $\Sigma_0^B(\mathcal{L}_A^2)$-bit-defining axiom in $\mathbf{V}^0(\mathcal{L}_{n+1})$. By definition, the function f_{n+1}/F_{n+1} already has an open defining/bit-defining axiom in the vocabulary $\mathcal{L}_n$. From the induction hypothesis, $\mathbf{V}^0(\mathcal{L}_n)$ satisfies the hypothesis of Lemma 5.40. Consequently the defining/bit-defining axiom for f_{n+1}/F_{n+1} is provably equivalent in $\mathbf{V}^0(\mathcal{L}_n)$ to a $\Sigma_0^B(\mathcal{L}_A^2)$ formula. Hence $\mathbf{V}^0(\mathcal{L}_{n+1})$ proves that f_{n+1}/F_{n+1} has a $\Sigma_0^B(\mathcal{L}_A^2)$ defining/bit-defining axiom, and this completes the proof of the claim. ⊣

Inspection of the above proof shows that each number function of $\mathcal{L}_{\mathbf{FAC}^0}$ has a $\Sigma_0^B(\mathcal{L}_A^2)$-defining axiom, and each string function of $\mathcal{L}_{\mathbf{FAC}^0}$ has a $\Sigma_0^B(\mathcal{L}_A^2)$-bit-defining axiom.

COROLLARY 5.73. *The $\mathcal{L}_{\mathbf{FAC}^0}$ functions are precisely the functions of* $\mathbf{FAC}^0$.

PROOF. By the above remark and the Σ_0^B-Representation Theorem 4.18, the $\mathcal{L}_{\mathbf{FAC}^0}$ functions are in $\mathbf{FAC}^0$. The other inclusion follows from the Σ_0^B-Representation Theorem 4.18 and Lemma 5.70. ⊣

The next lemma follows from Lemma 5.40 and the claim in the above proof of Theorem 5.72. It generalizes the *Row* Elimination Lemma 5.52.

LEMMA 5.74 ($\mathbf{FAC}^0$ Elimination Lemma). *Suppose that $\mathcal{L} \subseteq \mathcal{L}_{\mathbf{FAC}^0}$. Then for every $i \geq 0$ and every $\Sigma_i^B(\mathcal{L})$ formula φ^+ there is a $\Sigma_i^B(\mathcal{L}_A^2)$ formula φ so that $\mathbf{V}^0(\mathcal{L}) \vdash \varphi^+ \leftrightarrow \varphi$.*

5F.1. Alternative Proof of the Witnessing Theorem for $\mathbf{V}^0$. Here we show how to apply the Herbrand Theorem to $\overline{\mathbf{V}}^0$ to obtain a simple proof of Theorem 5.61. For notational simplicity, we consider the case of a single existential string quantifier, and prove Lemma 5.65.

Suppose that $\varphi(\vec{x}, \vec{X}, Z)$ is a Σ_0^B formula such that

$$\mathbf{V}^0 \vdash \forall\vec{x}\forall\vec{X}\exists Z\ \varphi(\vec{x}, \vec{X}, Z)$$

By Lemma 5.70 there is an open formula φ' over $\mathcal{L}_{\mathbf{FAC}^0}$ such that $\overline{\mathbf{V}}^0 \vdash \varphi \leftrightarrow \varphi'$. Since $\overline{\mathbf{V}}^0$ extends $\mathbf{V}^0$, we have

$$\overline{\mathbf{V}}^0 \vdash \forall\vec{x}\forall\vec{X}\exists Z\ \varphi'(\vec{x}, \vec{X}, Z)$$

Now $\overline{\mathbf{V}}^0$ is a universal theory, so by the Herbrand Theorem 4.33, there are terms $T_1(\vec{x}, \vec{X}), \ldots, T_n(\vec{x}, \vec{X})$ of $\overline{\mathbf{V}}^0$ such that

$$\overline{\mathbf{V}}^0 \vdash \forall\vec{x}\forall\vec{X}\big(\varphi'(\vec{x}, \vec{X}, T_1(\vec{x}, \vec{X})) \vee \cdots \vee \varphi'(\vec{x}, \vec{X}, T_n(\vec{x}, \vec{X}))\big)$$

Define $F(\vec{x}, \vec{X})$ by cases as follows:

$$F(\vec{x}, \vec{X}) = \begin{cases} T_1(\vec{x}, \vec{X}) & \text{if } \varphi'(\vec{x}, \vec{X}, T_1(\vec{x}, \vec{X})) \\ \vdots & \\ T_{n-1}(\vec{x}, \vec{X}) & \text{if } \varphi'(\vec{x}, \vec{X}, T_{n-1}(\vec{x}, \vec{X})) \\ T_n(\vec{x}, \vec{X}) & \text{otherwise} \end{cases}$$

It is easy to see that $F(\vec{x}, \vec{X})$ has a bit definition (85), and hence is a function in $\mathcal{L}_{\mathbf{FAC}^0}$, and

$$\overline{\mathbf{V}}^0 \vdash \forall\vec{x}\forall\vec{X}\varphi'(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

Now $\overline{\mathbf{V}}^0 \vdash \varphi \leftrightarrow \varphi'$, and also the proof of Theorem 5.72 shows that $\overline{\mathbf{V}}^0$ is conservative over $\mathbf{V}^0(F)$ (the extension of $\mathbf{V}^0$ resulting by adding the defining axioms for F). Hence

$$\mathbf{V}^0(F) \vdash \forall\vec{x}\forall\vec{X}\varphi(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

as required. □

The above proof shows that adding true Σ_0^B axioms to a theory does not increase the set of provably total functions in the theory. For example, let $\mathit{True}\Sigma_0^B$ be the set of all Σ_0^B formulas which are true in the standard model $\underline{\mathbb{N}}_2$. Let $\mathbf{V}^0(\mathit{True}\Sigma_0^B)$ be the result of adding $\mathit{True}\Sigma_0^B$ as axioms to $\mathbf{V}^0$, and let $\overline{\mathbf{V}}^0(\mathit{True}\Sigma_0^B)$ be the result of adding $\mathit{True}\Sigma_0^B$ as axioms to $\overline{\mathbf{V}}^0$. Then $\overline{\mathbf{V}}^0(\mathit{True}\Sigma_0^B)$ is a conservative extension of $\mathbf{V}^0(\mathit{True}\Sigma_0^B)$, and the above proof goes through to show that the same class $\mathbf{FAC}^0$ of functions serve to witness the Σ_1^1 theorems of $\mathbf{V}^0(\mathit{True}\Sigma_0^B)$. Thus we have shown

Corollary 5.75. *The provably total functions in* $\mathbf{V}^0(\mathit{True}\Sigma_0^B)$ *are precisely the functions in* $\mathbf{FAC}^0$.

5G. Finite Axiomatizability

Theorem 5.76. $\mathbf{V}^0$ *is finitely axiomatizable.*

Proof. It suffices to show that all Σ_0^B-**COMP** axioms follow from finitely many theorems of $\mathbf{V}^0$. Let 2-**BASIC**$^+$ (or simply B^+) denote the 2-**BASIC** axioms (Fig. 2) along with the finitely many theorems of $\mathbf{I}\Delta_0$ (and hence of $\mathbf{V}^0$) given in Examples 3.8 and 3.9 asserting that $+, \cdot, \leq$ satisfy the properties of a commutative discretely-ordered semiring.

We show more generally that both Σ_0^B-**COMP** and the multiple comprehension axioms (72) for all Σ_0^B formulas follow from B^+ and finitely many such comprehension instances. We use the notation $\varphi[\vec{a}, \vec{Q}](\vec{x})$ to indicate that the Σ_0^B formula φ can contain the free variables $\vec{a}, \vec{Q}$ in addition to $\vec{x} = x_1, ..., x_k$. Then for $k \geq 1$, $\mathbf{COMP}_\varphi(\vec{a}, \vec{Q}, \vec{b})$ denotes the comprehension formula

$$\exists Y \leq \langle b_1, ..., b_k\rangle \forall x_1 < b_1 ... \forall x_k < b_k (Y(\vec{x}) \leftrightarrow \varphi(\vec{x})) \tag{91}$$

We will show that $\mathbf{COMP}_\varphi$ for the following 12 formulas φ will suffice.

$\varphi_1(x_1, x_2) \equiv x_1 = x_2$
$\varphi_2(x_1, x_2, x_3) \equiv x_3 = x_1$
$\varphi_3(x_1, x_2, x_3) \equiv x_3 = x_2$

$$
\begin{array}{lcl}
\varphi_4[Q_1,Q_2](x_1,x_2) & \equiv & \exists y \le x_1(Q_1(x_1,y) \wedge Q_2(y,x_2)) \\
\varphi_5[a](x,y) & \equiv & y = a \\
\varphi_6[Q_1,Q_2](x,y) & \equiv & \exists z_1 \le y \exists z_2 \le y(Q_1(x,z_1) \wedge Q_2(x,z_2) \wedge y = z_1 + z_2) \\
\varphi_7[Q_1,Q_2](x,y) & \equiv & \exists z_1 \le y \exists z_2 \le y(Q_1(x,z_1) \wedge Q_2(x,z_2) \wedge y = z_1 \cdot z_2) \\
\varphi_8[Q_1,Q_2,c](x) & \equiv & \exists y_1 \le c \exists y_2 \le c(Q_1(x,y_1) \wedge Q_2(x,y_2) \wedge y_1 \le y_2) \\
\varphi_9[X,Q,c](x) & \equiv & \exists y \le c(Q(x,y) \wedge X(y)) \\
\varphi_{10}[Q](x) & \equiv & \neg Q(x) \\
\varphi_{11}[Q_1,Q_2](x) & \equiv & Q_1(x) \wedge Q_2(x) \\
\varphi_{12}[Q,c](x) & \equiv & \forall y \le c Q(x,y)
\end{array}
$$

In the following lemmas, we abbreviate $\mathbf{COMP}_{\varphi_i}(...)$ by C_i.

Lemma 5.77. *For each $k \ge 1$ and $1 \le i \le k$ let*

$$\psi_{ik}(x_1,\ldots,x_k,y) \equiv y = x_i$$

Then $B^+, C_1, C_2, C_3, C_4 \vdash \mathbf{COMP}_{\psi_{ik}}$.

Proof. We proceed by induction on k. For $k = 1$ we have $\psi_{1,1} \leftrightarrow \varphi_1(x_1,y)$ and for $k = 2$ we have $\psi_{2,1} \leftrightarrow \varphi_2(x_1,x_2,y)$ and $\psi_{2,2} \leftrightarrow \varphi_3(x_1,x_2,y)$. For $k > 2$, recall $\langle x_1,...,x_k\rangle = \langle\langle x_1,...,x_{k-1}\rangle, x_k\rangle$. Hence

$$B^+, C_3 \vdash \mathbf{COMP}_{\psi_{kk}}$$

For $1 \le i < k$ use C_4 with Q_1 defined by C_2 and Q_2 defined by $\mathbf{COMP}_{\psi_{i,k-1}}$. ⊣

Lemma 5.78. *Let $\vec{x} = x_1, \cdots, x_k$, $k \ge 1$, be a list of variables and let $t(\vec{x})$ be a term which in addition to possibly involving variables from $\vec{x}$ may involve other variables $\vec{a}, \vec{Q}$. Let $\psi_t[\vec{a},\vec{Q}](\vec{x},y) \equiv y = t(\vec{x})$. Then*

$$B^+, C_1, ..., C_7 \vdash \mathbf{COMP}_{\psi_t}(\vec{a},\vec{Q},\vec{b},d)$$

Proof. By using algebraic theorems in B^+ we may suppose that $t(\vec{x})$ is a sum of monomials in $x_1,...,x_k$, where the coefficients are terms involving $\vec{a},\vec{Q}$. The case $t \equiv u$, where u does not involve any x_i is obtained from C_5 with $a \leftarrow u$. The cases $t \equiv x_i$ are obtained from Lemma 5.77. We then build monomials using C_7 repeatedly, and build the general case by repeated use of C_6. ⊣

Lemma 5.79. *Let $t_1(\vec{x}), t_2(\vec{x})$ be terms with variables among $\vec{x},\vec{a},\vec{Q}$. Suppose*

$$
\begin{array}{lcl}
\psi_1[\vec{a},\vec{Q}](\vec{x}) & \equiv & t_1(\vec{x}) \le t_2(\vec{x}) \\
\psi_2[\vec{a},\vec{Q},X](\vec{x}) & \equiv & X(t_1(\vec{x}))
\end{array}
$$

Then $B^+, C_1, ..., C_9 \vdash \mathbf{COMP}_{\psi_i}$, for $i = 1,2$.

Proof. $\mathbf{COMP}_{\psi_1}(\vec{a},\vec{Q},\vec{b})$ follows from $\mathbf{COMP}_{\varphi_8}(Q_1,Q_2,c,b)$ with for $i = 1,2$, Q_i defined from $\mathbf{COMP}_{\psi_{t_i}}$ in Lemma 5.78 with $d \leftarrow t_1(\vec{b}) + t_2(\vec{b}) + 1$, so

$$\forall \vec{x} < \vec{b} \forall y < (t_1(\vec{b}) + t_2(\vec{b}) + 1)\,(Q_i(\vec{x},y) \leftrightarrow y = t_i(\vec{x}))$$

In $\mathbf{COMP}_{\varphi_8}$ we take $c \leftarrow t_1(\vec{b}) + t_2(\vec{b})$ and $b \leftarrow \langle b_1,...,b_k\rangle$.

For $\mathbf{COMP}_{\psi_2}(\vec{a},\vec{Q},X,\vec{b})$ we use $\mathbf{COMP}_{\varphi_9}(X,P,c,b)$ with $c \leftarrow t_1(\vec{b})$ and $b \leftarrow \langle b_1,...,b_k\rangle$ and P defined from Lemma 5.78 similarly to Q_1 above. $\dashv$

Now we can complete the proof of the theorem. Lemma 5.79 takes care of the case when φ is an atomic formula, since equations $t_1(\vec{x}) = t_2(\vec{x})$ can be initially replaced by $t_1(\vec{x}) \le t_2(\vec{x}) \wedge t_2(\vec{x}) \le t_1(\vec{x})$. Then by repeated applications of $\mathbf{COMP}_{\varphi_{10}}$ and $\mathbf{COMP}_{\varphi_{11}}$ we handle the case in which φ is quantifier-free.

Now suppose $\varphi(\vec{x}) \equiv \forall y \le t(\vec{x})\psi(\vec{x},y)$. We assume as an induction hypothesis that we can define Q satisfying

$$\forall \vec{x} < \vec{b} \forall y < t(\vec{b}) + 1\big(Q(\vec{x},y) \leftrightarrow (y \le t(\vec{x}) \supset \psi(\vec{x},y))\big)$$

Then $\mathbf{COMP}_{\varphi}(\vec{b})$ follows from $\mathbf{COMP}_{\varphi_{12}}(Q,c,b)$ with $c \leftarrow t(\vec{b})$ and $b \leftarrow \langle b_1,...,b_k\rangle$. $\dashv$

5H. Notes

The system $\mathbf{V}^0$ we introduce in this chapter is essentially Σ_0^p-*comp* in [85], and $\mathbf{I}\Sigma_0^{1,b}$ (without #) in [55]. Zambella [85] used $\mathcal{R}$ for $\mathbf{FAC}^0$ and called it the class of*rudimentary* functions. However there is danger here of confusion with Smullyan's rudimentary relations [78].

The set 2-**BASIC** is similar to the axioms for Zambella's theory Θ in [85], and forms the two-sorted analog of Buss's single-sorted axioms **BASIC** [12]. It is slightly different from that which are presented in [31] and [30].

The statement and proof of Theorem 5.61 (witnessing) are inspired by [12], although our treatment here is simplified because we only witness formulas in which all string quantifiers are in front.

The universal theory $\overline{\mathbf{V}}^0$ is taken from [30].

Theorem 5.76 (finite axiomatizability) is taken from Section 7 of [31].

Chapter 6

THE THEORY $\mathbf{V}^1$ AND POLYNOMIAL TIME

In this chapter we show that the theory $\mathbf{V}^1$ characterizes $\mathbf{P}$ in the same way that $\mathbf{V}^0$ characterizes $\mathbf{AC}^0$. This is stated in the Σ^1_1-Definability Theorem for $\mathbf{V}^1$: A function is Σ^1_1-definable (equivalently Σ^B_1-definable) in $\mathbf{V}^1$ if and only if it is in $\mathbf{FP}$. The "only if" direction follows from the *Witnessing Theorem for* $\mathbf{V}^1$.

The theory of algorithms can be viewed, to a large extent, as the study of polynomial time functions. All polytime algorithms can be described in $\mathbf{V}^1$, and experience has shown that proofs of their important properties can usually be formalized in $\mathbf{V}^1$. (See Example 6.30, prime recognition, for an apparent exception.) Razborov [75] has shown how to formalize lower bound proofs for Boolean complexity in $\mathbf{V}^1$. Standard theorems from graph theory, including Kuratowski's Theorem, Hall's Theorem, and Menger's Theorem can be formalized in $\mathbf{V}^1$.

In Chapter 8 we will introduce (apparently) weaker theories for polynomial time, and prove that they all have the same Σ^B_1-theorems (and hence the same Σ^1_1-theorems) as $\mathbf{V}^1$.

6A. Induction Schemes in $\mathbf{V}^i$

Recall (Definition 5.3) that $\mathbf{V}^i$ is axiomatized by 2-**BASIC** and Σ^B_i-**COMP**, where Σ^B_i-**COMP** consists of all formulas of the form

$$\exists X \le y \forall z < y (X(z) \leftrightarrow \varphi(z)), \tag{92}$$

where $\varphi(z)$ is a Σ^B_i formula, and X does not occur free in $\varphi(z)$.

The next result follows from Corollary 5.8.

Corollary 6.1. *For* $i \ge 0$, $\mathbf{V}^i$ *proves the* Σ^B_i-**IND**, Σ^B_i-**MIN**, *and* Σ^B_i-**MAX** *axiom schemes.*

It turns out that $\mathbf{V}^i$ proves these schemes for a wider class of formulas than just Σ^B_i. To show this, we start with a partial generalization of the Multiple Comprehension Lemma 5.50. Recall the projection functions *left* and *right* (Example 5.47).

LEMMA 6.2 (Multiple Comprehension Revisited). *Let $\mathcal{T}$ be a theory which extends $\mathbf{V}^0$ and has vocabulary $\mathcal{L}$, and suppose that either $\mathcal{L} = \mathcal{L}_A^2$ or $\mathcal{L}$ includes the projection functions left and right. For each $i \geq 0$, if $\mathcal{T}$ proves the $\Sigma_i^B(\mathcal{L})$-**COMP** axioms, then $\mathcal{T}$ proves the multiple comprehension axiom* (72)*:*

(93)
$$\exists X \leq \langle y_1, \dots, y_k \rangle \forall z_1 < y_1 \dots \forall z_k < y_k (X(z_1, \dots, z_k) \leftrightarrow \varphi(z_1, \dots, z_k))$$

*for any $k \geq 2$ and any $\varphi \in \Sigma_i^B(\mathcal{L})$. In particular, for all $i \geq 0$, $\mathbf{V}^i$ proves Σ_i^B-**MULTICOMP**.*

PROOF. The method used to prove the earlier version, Lemma 5.50, does not work here, because for $i \geq 1$ the $\Sigma_i^B(\mathcal{L})$-formulas are not closed under bounded number quantification.

For notational simplicity we prove the case $k = 2$. First we consider the case that $\mathcal{L}$ includes *left* and *right*. Assuming that $\varphi(z_1, z_2)$ is in $\Sigma_i^B(\mathcal{L})$ and $\mathcal{T}$ proves the $\Sigma_i^B(\mathcal{L})$-**COMP** axioms, it follows that $\mathcal{T}$ proves

$$\exists X \leq \langle y_1, y_2 \rangle \forall z < \langle y_1, y_2 \rangle (X(z) \leftrightarrow \varphi(\mathit{left}(z), \mathit{right}(z)))$$

Now (93) follows by the properties of *left*,*right* (Exercise 5.48) and the notation (71) stating that $X(z_1, z_2) \equiv X(\langle z_1, z_2 \rangle)$.

For the case $\mathcal{L} = \mathcal{L}_A^2$, we work in the conservative extension $\mathcal{T}(\mathit{left}, \mathit{right})$ of $\mathcal{T}$. By the $\mathbf{FAC}^0$ Elimination Lemma 5.74, if $\mathcal{T}$ proves the Σ_i^B-**COMP** axioms, it follows that $\mathcal{T}(\mathit{left}, \mathit{right})$ proves the $\Sigma_i^B(\mathit{left}, \mathit{right})$-**COMP** axioms and hence also (93) by the previous case. Hence also $\mathcal{T}$ proves (93) by conservativity. ⊣

The next result refers to the Σ_0^B-closure of a set of formulas (Definition 5.34).

THEOREM 6.3. *Let $\mathcal{T}$ be a theory over a vocabulary $\mathcal{L}$ which extends $\mathbf{V}^0$ and proves the multiple comprehension axioms* (93) *for every $k \geq 1$ and every φ in some class Φ of $\mathcal{L}$-formulas. Then $\mathcal{T}$ proves the $\Sigma_0^B(\Phi)$-**COMP** axioms.*

The following result is an immediate consequence of this theorem, Lemma 6.2, and Corollary 5.8, since every Π_i^B formula is equivalent to a negated Σ_i^B formula.

COROLLARY 6.4. *For $i \geq 0$ let Φ_i be $\Sigma_0^B(\Sigma_i^B \cup \Pi_i^B)$. Then $\mathbf{V}^i$ proves the Φ_i-**COMP**, Φ_i-**IND**, Φ_i-**MIN**, and Φ_i-**MAX** axiom schemes.*

PROOF OF THEOREM 6.3. We prove the stronger assertion that $\mathcal{T}$ proves the multiple comprehension axioms (93) for $\varphi \in \Sigma_0^B(\Phi)$, by structural induction on φ relative to Φ. We use the fact that $\mathcal{T}$ extends $\mathbf{V}^0$ and hence by Lemma 6.2 proves the multiple comprehension axioms for Σ_0^B-formulas.

The base case, $\varphi \in \Phi$, holds by hypothesis. For the induction step, consider the case that φ has the form $\neg\psi$. By the induction hypothesis $\mathcal{T}$ proves

$$\exists Y \leq \langle \vec{y} \rangle \forall \vec{z} < \vec{y}(Y(\vec{z}) \leftrightarrow \psi(\vec{z}))$$

and by Lemma 6.2, $\mathcal{T}$ proves

$$\exists X \leq \langle \vec{y} \rangle \forall \vec{z} < \vec{y}(X(\vec{z}) \leftrightarrow \neg Y(\vec{z}))$$

Thus $\mathcal{T}$ proves (93).

The cases $\wedge$ and $\vee$ are similar. Finally we consider the case that $\varphi(\vec{z})$ has the form $\forall x \leq t\psi(x, \vec{z})$. By the induction hypothesis $\mathcal{T}$ proves

$$\exists Y \leq \langle t+1, \vec{y} \rangle \forall x \leq t \forall \vec{z} < \vec{y}(Y(x, \vec{z}) \leftrightarrow \psi(x, \vec{z}))$$

By Lemma 5.50 $\mathbf{V}^0$ proves

$$\exists X \leq \langle \vec{y} \rangle \forall \vec{z} < \vec{y}(X(\vec{z}) \leftrightarrow \forall x \leq tY(x, \vec{z}))$$

Now (93) follows from these two formulas. $\dashv$

6B. Characterizing P by $\mathbf{V}^1$

The class (two-sorted) **P** consists of relations computable in polynomial time by a deterministic Turing machine (i.e., *polytime relations*), and **FP** is the class of functions computable in polynomial time by a deterministic Turing machine (i.e., *polytime functions*). Alternatively (Definition 5.15) **FP** is the class of the polynomially bounded number functions whose graphs are in **P**, and the polynomially bounded string functions whose bit graphs are in **P**.

Recall that a number input to the accepting machine is represented as a unary string, and a set input is represented as a binary string (page 78). (Thus a purely numerical function $f(\vec{x})$ is in **FP** iff it is computed in time $2^{O(n)}$, where n is the length of the *binary* notation for its arguments.)

The following proposition follows easily from the definitions involved.

Proposition 6.5. (a) *A number function $f(\vec{x}, \vec{X})$ is in* **FP** *iff there is a string function $F(\vec{x}, \vec{X})$ in* **FP** *so that $f(\vec{x}, \vec{X}) = |F(\vec{x}, \vec{X})|$.*
(b) *A relation is in* **P** *iff its characteristic function is in* **FP**.

We will prove that the theory $\mathbf{V}^1$ characterizes **P** in the same way that $\mathbf{V}^0$ characterizes $\mathbf{AC}^0$:

Theorem 6.6 (Σ_1^1-Definability Theorem for $\mathbf{V}^1$). *A function is Σ_1^1-definable in $\mathbf{V}^1$ iff it is in* **FP**.

The "if" direction is proved in Section 6B.1. The "only-if" direction follows immediately from the Witnessing Theorem for $\mathbf{V}^1$ (Theorem 6.28).

Note that $\mathbf{V}^1$ is a polynomial-bounded theory (Definition 5.23). The following corollary follows from the Σ_1^1-Definability Theorem for $\mathbf{V}^1$ above, and Parikh's Theorem (see Corollary 5.29).

COROLLARY 6.7. *A function is in* $\mathbf{FP}$ *iff it is* Σ_1^B*-definable in* $\mathbf{V}^1$.

The next corollary follows from the results above and Theorem 5.60.

COROLLARY 6.8. *A relation is in* $\mathbf{P}$ *iff it is is* Δ_1^1*-definable in* $\mathbf{V}^1$ *iff it is* Δ_1^B*-definable in* $\mathbf{V}^1$.

Recall (Theorem 4.19) that the Σ_1^B formulas represent precisely the $\mathbf{NP}$ relations, and hence by Definition 5.57 a relation is Δ_1^B definable in a theory $\mathcal{T}$ iff $\mathcal{T}$ proves that the relation is in both $\mathbf{NP}$ and *co*-$\mathbf{NP}$. Thus the above corollary says that a relation is in $\mathbf{P}$ iff $\mathbf{V}^1$ proves that it is in $\mathbf{NP} \cap$ *co*-$\mathbf{NP}$.

COROLLARY 6.9. $\mathbf{V}^1$ *is a proper extension of* $\mathbf{V}^0$.

PROOF. There are relations (such as $PARITY(X)$ — page 114) which are in $\mathbf{P}$ but not in $\mathbf{AC}^0$. $\dashv$

EXERCISE 6.10 ($parity(X)$ in $\mathbf{V}^1$). *Recall the formula* $\varphi_{parity}(X,Y)$ *((80) on page 114). Show that the function* $parity(X)$*, which is the characteristic function of* $PARITY$ *(page 114), is* Σ_1^1*-definable in* $\mathbf{V}^1$ *by showing that*

$$\mathbf{V}^1 \vdash \forall X \exists ! Y \varphi_{parity}(X,Y)$$

EXERCISE 6.11 (String Multiplication in $\mathbf{V}^1$). *Consider the string multiplication function* $X \times Y$ *where*

$$X \times Y = Z \leftrightarrow bin(Z) = bin(X) \cdot bin(Y)$$

and $bin(X)$ *is the integer value of the binary string* X *(see (46) on page 82). Consider the the* Σ_1^1 *defining axiom for* $X \times Y$ *in* $\mathbf{V}^1$ *that is based on the "school" algorithm for multiplying two integers written in binary notation. First, we construct the* table $X \otimes Y$ *that has* $|Y|$ *rows and whose* i*th row is either 0, if* $Y(i) = 0$ *(i.e.,* $\neg Y(i)$*), or a copy of* X *shifted left by* i *bits, if* $Y(i) = 1$*. Thus,* $X \otimes Y$ *can be defined by (see Definition 5.51 for row notation)*

$$X \otimes Y = Z \leftrightarrow |Z| \leq \langle |Y|, |X| + |Y| \rangle \wedge$$
$$\forall i < |Y| \forall z < i + |X| \big(Z^{[i]}(z) \leftrightarrow (Y(i) \wedge \exists u \leq z\, (u + i = z \wedge X(u)))\big)$$

(a) *Let* $Z = X \otimes Y$*. Show that* $\mathbf{V}^0$ *proves the existence and uniqueness of* Z.

(b) *Show that* $\mathbf{V}^1$ *proves the existence and uniqueness of* W*, where*

$$|W| \leq 1 + \langle |Y|, |X| + |Y| \rangle \wedge |W^{[0]}| = 0 \wedge \forall i < |Y| (W^{[i+1]} = W^{[i]} + Z^{[i]})$$

(Hint: Use Σ_1^B-$\mathbf{IND}$*. For the bound on* $|W|$*, show that* $|W^{[i]}| \leq |X| + i$*.)*

(c) *Define $X \times Y$ in terms of $X \otimes Y$. Conclude that the string multiplication function is provably total in $\mathbf{V}^1$.*

(d) *Recall string functions $\varnothing$, S and $X+Y$ from Example 5.42. Show that the following are theorems of $\mathbf{V}^1(\varnothing, S, +, \times)$:*
  (i) $X \times \varnothing = \varnothing$.
  (ii) $X \times S(Y) = (X \times Y) + X$.

Now we argue that the subtheory **IOPEN** of Peano Arithmetic (Definition 3.7) can be interpreted in $\mathbf{V}^1$ by interpreting each number x by the unique string X such that $bin(X) = x$. Then $+$ and $\times$ in **IOPEN** are interpreted by Example 5.42 and Exercise 6.11 respectively, 0 is interpreted by $\varnothing$, and 1 by the string constant $1 = \{0\}$. It is easy to give a Σ_0^B formula defining the relation $X \leq Y$ to interpret $\leq$. Then one can check that $\mathbf{V}^0 \vdash S(X) = X+1$, and with the help of Exercises 5.44 and 6.11 it is not hard to show that $\mathbf{V}^1(\varnothing, 1, +, \times)$ proves the string interpretations of axioms **B**1, ..., **B**8 for **PA**.

It remains to show that $\mathbf{V}^1(\varnothing, 1, +, \times)$ proves the string interpretations of the induction axiom scheme (10) (on page 39) for open formulas $\varphi(x)$. In fact this will follow from our discussion in Section 8C (see Corollary 8.54 and Theorem 8.27).

Consequently $\mathbf{V}^1$ proves the string interpretations of the formulas given in Example 3.8 (commutativity and associativity of $+, \times$ etc). In fact it is not hard to show that $\mathbf{V}^1$ also proves the formulas (involving $+, \times, \leq$) given in Example 3.9, even though some of the proofs given there involve induction on Σ_1^b-formulas rather than on just open formulas.

Exercise 6.12 (String Division and Remainder in $\mathbf{V}^1$). *Consider the string division function $X \div Y = \lfloor X/Y \rfloor$ and the string remainder function $Rem(X,Y) = X - Y \times (X \div Y)$. These functions can be Σ_1^1-defined in $\mathbf{V}^1$ by the following steps. Suppose that $Y \leq X$, and let z be such that $z + |Y| = |X|$.*

(a) *Give a Σ_0^B-bit-definition for the table U, where the row $U^{[i]}$ of U is Y shifted left by i bits, for $0 \leq i \leq z$.*

(b) *Prove in $\mathbf{V}^1$ the existence and uniqueness of a table W such that*

$$W^{[z]} = X \wedge \forall i < z\big((W^{[i+1]} < U^{[i+1]} \supset W^{[i]} = W^{[i+1]}) \wedge \\ (U^{[i+1]} \leq W^{[i+1]} \supset W^{[i]} + U^{[i+1]} = W^{[i+1]})\big)$$

(c) *Define $X \div Y$ and $Rem(X,Y)$ using W.*

(d) *Show in $\mathbf{V}^1(+, \times, \div, Rem)$ that*

$$X = (Y \times (X \div Y)) + Rem(X,Y)$$

6B.1. The "if" direction of Theorem 6.6. We will give two proofs of the fact that every polynomial time function is Σ_1^1-definable in $\mathbf{V}^1$. The first is based directly on Turing machine computations, and the second is based on Cobham's characterization of **FP**. We give the second proof in more detail, since it provides the basis for the universal theory **VPV** described in Chapter 8.

The key idea for the first proof is that the computation of a polytime Turing machine M on a given input $\vec{x}, \vec{X}$ can be encoded as a string of configurations (see Definition 5.51 for notation)

$$Z = \langle Z^{[0]}, Z^{[1]}, \dots Z^{[m]} \rangle$$

whose length is bounded by some polynomial in $\vec{x}, |\vec{X}|$, and whose existence we need to prove in $\mathbf{V}^1$. The output of M can then be extracted from Z easily. The defining axiom for the polytime function computed by M is the formula that states the existence of such Z.

Exercise 6.13. *Describe a method of coding Turing machine configurations by strings, and show that for each Turing machine* M *working on input* $\vec{x}, \vec{X}$ *there are* Σ_0^B*-definable string functions in* $\mathbf{V}^0$*:* $Init_{\mathsf{M}}(\vec{x}, \vec{X})$*,* $Next_{\mathsf{M}}(Z)$ *and* $Out_{\mathsf{M}}(Z)$ *such that*

- $Init_{\mathsf{M}}(\vec{x}, \vec{X})$ *is the initial configuration of* M *on input* $(\vec{x}, \vec{X})$*;*
- $Z' = Next_{\mathsf{M}}(Z)$ *if* Z *and* Z' *code two consecutive configurations of* M*, or* $Z' = Z$ *if* Z *codes a final configuration of* M*, or* $Z' = \varnothing$ *if* Z *does not code a configuration of* M*.*
- $Out_{\mathsf{M}}(Z)$ *is the tape contents of a configuration* Z *of* M*, or* $\varnothing$ *if* Z *does not code a configuration of* M*.*

Below we will use all three functions in the above exercise, as well as the string function $Row(z, Y)$ (Definition 5.51). Because these functions are Σ_0^B-definable in $\mathbf{V}^0$, it follows from the $\mathbf{FAC}^0$ Elimination Lemma 5.74 that any $\Sigma_0^B(\mathcal{L}_A^2 \cup \{Init, Next, Out, Row\})$ formula can be transformed into a provably equivalent $\Sigma_0^B(\mathcal{L}_A^2)$ formula. Formally we will work in the conservative extension of $\mathbf{V}^1$ consisting of $\mathbf{V}^1$ together with the defining axioms for these functions, although we will continue to refer to this theory as simply $\mathbf{V}^1$. Thus each Σ_0^B (resp. Σ_1^B) formula below with the new functions is provably equivalent to a Σ_0^B (resp. Σ_1^B) formula in the language of $\mathbf{V}^1$.

First proof of the $\Longleftarrow$ direction of Theorem 6.6. Consider the case of string functions. (The case of number functions is similar.) Suppose that $F(\vec{x}, \vec{X})$ is a polytime function. Let M be a Turing machine which computes $F(\vec{x}, \vec{X})$ in time polynomial of $\vec{x}, |\vec{X}|$, and let $t(\vec{x}, |\vec{X}|)$ be a bound on the running time of M on input $\vec{x}, \vec{X}$. We may assume that M halts with $F(\vec{x}, \vec{X})$ equal to the contents of its tape, so that $Out_{\mathsf{M}}(Z) = F(\vec{x}, \vec{X})$ if Z codes the final configuration. Then

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \exists Z \le \langle t, t \rangle (\varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z) \wedge Y = Out_{\mathsf{M}}(Z^{[t]})) \tag{94}$$

where $\varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z)$ is the formula

$$Z^{[0]} = Init_{\mathsf{M}}(\vec{x}, \vec{X}) \wedge \forall z < t \; (Z^{[z+1]} = Next_{\mathsf{M}}(Z^{[z]}))$$

We will show that the RHS of (94) is a defining axiom for F in $\mathbf{V}^1$, i.e.,

$$\mathbf{V}^1 \vdash \forall \vec{x} \forall \vec{X} \exists ! Y \exists Z \leq \langle t, t\rangle (\varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z) \wedge Y = Out_{\mathsf{M}}(Z^{[t]}))$$

For the uniqueness of Y, it suffices to verify that if Z_1 and Z_2 are two strings satisfying

$$|Z_k| \leq \langle t, t\rangle \wedge \varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z_k)$$

(for $k = 1, 2$), then for all z,

$$z \leq t \supset Z_1^{[z]} = Z_2^{[z]} \tag{95}$$

This follows in $\mathbf{V}^1$ using Σ_0^B-**IND** on the formula (95) with induction on z.

For the existence of Y, we need to show that $\mathbf{V}^1$ proves

$$\forall \vec{x} \forall \vec{X} \exists Z \leq \langle t, t\rangle\ \varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z)$$

This formula can be proved in $\mathbf{V}^1$ by using number induction axiom (Corollary 6.1) on b for the Σ_1^B formula

$$\exists W \leq \langle b, t\rangle (W^{[0]} = Init_{\mathsf{M}}(\vec{x}, \vec{X}) \wedge \forall z < b W^{[z+1]} = Next_{\mathsf{M}}(W^{[z]})) \qquad \dashv$$

Exercise 6.14. *Carry out details of the induction step in the proof of the above formula.*

An alternative proof for the above direction of Theorem 6.6 can be obtained by using Cobham's characterization of **FP**. To explain this, we need the notion of *limited recursion.* First we introduce the $\mathbf{AC}^0$ string function $Cut(x, X)$, which is the initial segment of X and contains all elements of X that are $< x$. It has the Σ_0^B-bit-defining axiom

$$Cut(x, X)(z) \leftrightarrow z < x \wedge X(z) \tag{96}$$

Notation. We will sometimes write $X^{<x}$ for $Cut(x, X)$.

Definition 6.15 (Limited Recursion). *A string function $F(y, \vec{x}, \vec{X})$ is defined by* limited recursion *from $G(\vec{x}, \vec{X})$ and $H(y, \vec{x}, \vec{X}, Z)$ iff*

$$F(0, \vec{x}, \vec{X}) = G(\vec{x}, \vec{X}) \tag{97}$$

$$F(y+1, \vec{x}, \vec{X}) = (H(y, \vec{x}, \vec{X}, F(y, \vec{x}, \vec{X})))^{<t(y, \vec{x}, \vec{X})} \tag{98}$$

for some $\mathcal{L}_A^2$-term t representing a polynomial in $y, \vec{x}, |\vec{X}|$.

For two-sorted function classes, we can also define the notion of limited recursion for a number function. However here we can just appeal to Proposition 6.5 (a) when we have to deal with number functions. A version of Cobham's characterization of **FP** is as follows.

Theorem 6.16 (Cobham's Characterization of **FP**). *A string function is in **FP** iff it can be obtained from $\mathbf{AC}^0$ functions by finitely many applications of composition and limited recursion.*

PROOF SKETCH. The $\Longleftarrow$ direction follows from the fact that $\mathbf{AC}^0$ functions are in $\mathbf{FP}$, and that applying the operations composition and limited recursion to functions in $\mathbf{FP}$ results in functions in $\mathbf{FP}$.

For the $\Longrightarrow$ direction, the function F computed by a polytime Turing machine M can be defined from the $\mathbf{AC}^0$ functions $\mathit{Init}_{\mathsf{M}}$, $\mathit{Next}_{\mathsf{M}}$ and $\mathit{Out}_{\mathsf{M}}$ by limited recursion and composition. In more detail, we can define a string function $\mathit{Conf}_{\mathsf{M}}(y, \vec{x}, \vec{X})$ to be the string coding the configuration of M on input $(\vec{x}, \vec{X})$ at time y. Then $\mathit{Conf}_{\mathsf{M}}$ satisfies the recursion

$$\mathit{Conf}_{\mathsf{M}}(0, \vec{x}, \vec{X}) = \mathit{Init}_M(\vec{x}, \vec{X})$$
$$\mathit{Conf}_{\mathsf{M}}(y+1, \vec{x}, \vec{X}) = \mathit{Next}_{\mathsf{M}}(\mathit{Conf}_{\mathsf{M}}(y, \vec{x}, \vec{X}))$$

To turn this recursion into one fitting Definition 6.15 we apply

$$\mathit{Cut}(t(y, \vec{x}, \vec{X}), \dots)$$

to the RHS of the second equation, for a suitable $\mathcal{L}_A^2$-term t bounding the run time of M. Then

(99) $$F(\vec{x}, \vec{X}) = \mathit{Out}_{\mathsf{M}}(\mathit{Conf}_{\mathsf{M}}(t(\vec{x}, \vec{X}), \vec{x}, \vec{X})) \qquad \dashv$$

6B.2. Application of Cobham's Theorem.

SECOND PROOF OF THE $\Longleftarrow$ DIRECTION OF THEOREM 6.6. We use Cobham's characterization of $\mathbf{FP}$ to show that the polytime string functions are Σ_1^1-definable in $\mathbf{V}^1$. It follows from Proposition 6.5 that the polytime number functions are also Σ_1^1-definable in $\mathbf{V}^1$.

We proceed by induction on the number of applications of composition and limited recursion needed to obtain F from $\mathbf{AC}^0$ functions. For the base case, the $\mathbf{AC}^0$ functions are Σ_1^1-definable in $\mathbf{V}^0$ (Corollary 5.62), hence also in $\mathbf{V}^1$. For the induction step, we need to show that the Σ_1^1-definable functions of $\mathbf{V}^1$ are closed under composition and limited recursion. The case of composition is easily seen to hold for any theory $\mathcal{T}$ (see exercise 5.30). Hence it suffices to prove the case of limited recursion.

Suppose that $G(\vec{x}, \vec{X})$ and $H(y, \vec{x}, \vec{X}, Z)$ are Σ_1^1-definable functions in $\mathbf{V}^1$, and $F(y, \vec{x}, \vec{X})$ is defined by limited recursion from G and H as in (97) and (98) for some polynomial p. Then we can Σ_1^1-define F by coding the sequence of values $F(0), F(1), \dots, F(y)$ as the rows $W^{[0]}, W^{[1]}, \dots, W^{[y]}$ of a single array W. Thus (omitting $\vec{x}, \vec{X}$):

$$Y = F(y) \leftrightarrow \big(\exists W\, W^{[0]} = G() \wedge$$
$$(\forall z < y\ W^{[z+1]} = (H(z, W^{[z]}))^{<t(z)}) \wedge Y = W^{[y]}\big)$$

The RHS is not immediately equivalent to a Σ_1^1 formula when the equations involving G and H are replaced by Σ_1^1 formulas using the defining axioms for G and H. This is because of the number quantifier $\forall z < y$ of the middle conjunct, which is mixed in between the existential string quantifiers. We obtain a Σ_1^1-defining axiom for F from the RHS as follows:

By assumption, G and H have Σ_1^1-defining axioms. Therefore there are Σ_0^B formulas φ_G and φ_H so that

$$W = G() \leftrightarrow \exists\vec{U}\varphi_G(\vec{U}, W), \qquad W = H(y, Z) \leftrightarrow \exists\vec{V}\varphi_H(y, Z, \vec{V}, W)$$

and

$$\mathbf{V}^1 \vdash \exists! W \exists\vec{U}\varphi_G(\vec{U}, W) \tag{100}$$

$$\mathbf{V}^1 \vdash \forall y \forall Z \exists! W \exists\vec{V}\varphi_H(y, Z, \vec{V}, W) \tag{101}$$

The Σ_1^1-defining axiom for F is obtained by using arrays $\vec{V}$ for which $\vec{V}^{[z]}$ (row z in the arrays $\vec{V}$) codes the values of $\vec{V}$ needed to satisfy (101) when evaluating $H(z, W^{[z]})$.

$$\begin{aligned} Y = F(y) \leftrightarrow \big(\exists W \exists\vec{U}\exists\vec{V}\ \varphi_G(\vec{U}, W^{[0]}) \wedge \\ (\forall z < y \varphi_H(z, W^{[z]}, \vec{V}^{[z]}, (W^{[z+1]})^{<t(z)})) \wedge Y = W^{[y]}\big) \end{aligned} \tag{102}$$

Since the terms such as $(W^{[z+1]})^{<t(z)}$ are easily seen to be Σ_0^B-bit-definable, it follows from Lemma 5.74 that this defining axiom can be replaced by an equivalent Σ_1^1-formula (see the discussion following Exercise 6.13).

It is easy to see that $\mathbf{V}^1$ proves the uniqueness of Y by proving that if W_1 and W_2 satisfy (102), then for $z \le y$ we have $W_1^{[z]} = W_2^{[z]}$. This is by number induction on $z \le y$, and follows from the uniqueness of W in (100) and (101).

Now we show that $\mathbf{V}^1$proves the existence of Y satisfying the RHS of (102). We start by noting that all of the initial string quantifiers can be bounded. This follows from Parikh's Theorem, using (100) and (101). Let $\psi(y)$ be the Σ_1^B-formula obtained from this bounded form of the RHS of (102), with the final conjunct $Y = W^{[y]}$ deleted. Thus $\psi(y)$ asserts the existence of an array

$$W = (W^{[0]}, W^{[1]}, \ldots, W^{[y]})$$

whose rows are the successive values

$$F(0), F(1), \ldots, F(y)$$

We show that $\mathbf{V}^1$ proves $\psi(y)$ by induction on y. The base case follows from (100): If W' satisfies the existential quantifier $\exists W$ in (100), then W satisfying $\psi(y)$ can be defined using multiple comprehension (Lemma 6.2):

$$W(0, i) \leftrightarrow W'(i)$$

For the induction step, the new values of W and $\vec{V}$ for $y+1$ are obtained by pasting together the previous values for y, together with values from (101) with (y, Z) in φ_H replaced by $(y, W^{[y]})$. The pasting is again defined using multiple comprehension.

Hence $\mathbf{V}^1 \vdash \psi(y)$. From this it follows that $\mathbf{V}^1$ proves the existence of Y satisfying the RHS of (102): just set $Y = W^{[y]}$. Hence $F(y)$ is Σ_1^1-definable in $\mathbf{V}^1$. $\dashv$

6C. The Replacement Axiom Scheme

Recall that the classes Σ_i^B and Π_i^B consist of formulas in prenex form, whose string quantifiers precede the number quantifiers. Below we define more general classes.

Definition 6.17 ($\mathbf{g}\Sigma_i^B(\mathcal{L})$ and $\mathbf{g}\Pi_i^B(\mathcal{L})$). *For a vocabulary $\mathcal{L}$ extending $\mathcal{L}_A^2$, define*

$$\mathbf{g}\Sigma_0^B(\mathcal{L}) = \mathbf{g}\Pi_0^B(\mathcal{L}) = \Sigma_0^B(\mathcal{L})$$

For $i \geq 0$, $\mathbf{g}\Sigma_{i+1}^B$ is the closure of $\mathbf{g}\Pi_i^B$ under $\wedge$, $\vee$, $\forall x \leq t$, $\exists x \leq t$ and $\exists X \leq t$. Similarly, $\mathbf{g}\Pi_{i+1}^B$ is the closure of $\mathbf{g}\Sigma_i^B$ under $\wedge$, $\vee$, $\forall x \leq t$, $\exists x \leq t$ and $\forall X \leq t$.

As usual, we will drop mention of $\mathcal{L}$ when it is clear from context. Notice that for $i \geq 0$, $\Sigma_i^B \subset \Sigma_0^B(\Sigma_i^B) \subset \mathbf{g}\Sigma_i^B$, and $\Pi_i^B \subset \Sigma_0^B(\Pi_i^B) \subset \mathbf{g}\Pi_i^B$. Also

$$\Sigma_0^B \subset \mathbf{g}\Sigma_1^B \subset \mathbf{g}\Sigma_2^B \subset \dots \qquad \text{and} \qquad \Sigma_0^B \subset \mathbf{g}\Pi_1^B \subset \mathbf{g}\Pi_2^B \subset \dots$$

For any formula φ^+ in $\mathbf{g}\Sigma_i^B$, there is a formula φ in Σ_i^B so that in $\underline{\mathbb{N}}_2$ we have $\varphi^+ \leftrightarrow \varphi$. In particular, when φ^+ is a $\mathbf{g}\Sigma_1^B$ formula of the form

$$\forall x \leq t \exists X \leq t \psi(x, X)$$

where ψ is a Σ_0^B formula, then we can collect the values of X for $x = 0, 1, \dots, t$ into a single array Y whose rows $Y^{[0]}, Y^{[1]}, \dots Y^{[t]}$ are these successive values of X. Thus we can take φ to be

$$\exists Y \leq \langle t, t \rangle \forall x \leq t (|Y^{[x]}| \leq t \wedge \psi(x, Y^{[x]}))$$

In this case φ^+ is a logical consequence of φ, and $\varphi^+ \supset \varphi$ is true in $\underline{\mathbb{N}}_2$. In this section we are concerned with the provability of formulas of the type $\varphi^+ \supset \varphi$ in our theories. Consider the following axiom scheme.

Definition 6.18 (Replacement Axiom). *For a set Φ of formulas over the vocabulary $\mathcal{L}$, the* replacement axiom scheme *for Φ, denoted by Φ-***REPL***, is the set of all formulas (over $\mathcal{L} \cup \{Row\}$):*

$$(103) \quad (\forall x \leq b \exists X \leq c \varphi(x, X)) \supset \exists Z \leq \langle b, c \rangle \forall x \leq b (|Z^{[x]}| \leq c \wedge \varphi(x, Z^{[x]}))$$

where φ is in Φ.

Note that in (103) the LHS is a logical consequence of the RHS. Also (103) is true in the expansion of the standard model $\underline{\mathbb{N}}_2$, for any formula φ.

The function *Row* occurs on the RHS of (103), but by the *Row* Elimination Lemma 5.52 (or more generally the $\mathbf{FAC}^0$ Elimination Lemma 5.74), any $\Sigma_0^B(Row)$ formula is equivalent to a $\Sigma_0^B(\mathcal{L}_A^2)$ formula. So in the context of the theories with underlying vocabulary $\mathcal{L}_A^2$ (such as $\mathbf{V}^i$, or $\tilde{\mathbf{V}}^1$ below), we define (103) to be the equivalent $\mathcal{L}_A^2$ formula which is obtained by transforming every atomic sub-formula containing *Row* into a $\Sigma_0^B(\mathcal{L}_A^2)$ formula.

Notation. When we say that a theory $\mathcal{T}$ with vocabulary $\mathcal{L}$ proves a **REPL** axiom scheme (e.g., $\Sigma_0^B(\mathcal{L})$-**REPL**), then either $\mathcal{L}_A^2 \cup \{Row\} \subseteq \mathcal{L}$, or $\mathcal{L} = \mathcal{L}_A^2$ and (103) is as above.

Recall that a *single*-Σ_1^B formula has the form $\exists X \le t\psi(X)$, where ψ is a Σ_0^B formula.

Lemma 6.19. *Suppose that $\mathcal{T}$ is a polynomial–bounded theory which proves the $\Sigma_0^B(\mathcal{L})$-**REPL** axiom scheme, where $\mathcal{L}$ is the vocabulary of $\mathcal{T}$ (so either $\mathcal{L} = \mathcal{L}_A^2$, or $\mathcal{L}_A^2 \cup \{Row\} \subseteq \mathcal{L}$). Then for each $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formula φ there is a single-$\Sigma_1^B(\mathcal{L})$ formula φ' so that $\mathcal{T} \vdash \varphi \leftrightarrow \varphi'$.*

Proof. We prove by structural induction on the formula φ. For the base case, if φ is a $\Sigma_0^B(\mathcal{L})$ formula, then we can simply take $\varphi' \equiv \varphi$.

For the induction step, consider the interesting case where φ has the form $\forall x \le s\theta(x)$, where θ is a $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formula but not a $\Sigma_0^B(\mathcal{L})$ formula. By the induction hypothesis, $\theta(x)$ is equivalent in $\mathcal{T}$ to a *single*-$\Sigma_1^B(\mathcal{L})$ formula $\exists X \le t\psi(x, X)$, where ψ is a $\Sigma_0^B(\mathcal{L})$ formula. In other words,

$$\mathcal{T} \vdash \varphi \leftrightarrow \forall x \le s \exists X \le t\psi(x, X)$$

Now $\mathcal{T}$ proves φ is equivalent to a *single*-$\Sigma_1^B(\mathcal{L})$ formula by $\Sigma_0^B(\mathcal{L})$-**REPL**.

The other cases for the induction step follow easily with the help of exercise 5.55, which shows that a prefix of several bounded string quantifiers can be collapsed into a single one. ⊣

In the next lemma we generalize the previous lemma. Part (b) follows easily from (a), and (a) can be proved by induction on i. The base case is proved in Lemma 6.19. The induction step is similar to the base case.

Lemma 6.20. *Let $\mathcal{T}$ be a polynomial–bounded theory with vocabulary $\mathcal{L}$ which proves the $\Pi_i^B(\mathcal{L})$-**REPL** axiom scheme, for some $i \ge 0$ (so either $\mathcal{L} = \mathcal{L}_A^2$, or $\mathcal{L}_A^2 \cup \{Row\} \subseteq \mathcal{L}$). Then*

(a) *For each $\mathbf{g}\Sigma_{i+1}^B(\mathcal{L})$ formula φ there is a $\Sigma_{i+1}^B(\mathcal{L})$ formula φ' so that $\mathcal{T} \vdash \varphi \leftrightarrow \varphi'$.*

(b) *For each $\mathbf{g}\Pi_{i+1}^B(\mathcal{L})$ formula φ there is a $\Pi_{i+1}^B(\mathcal{L})$ formula φ' so that $\mathcal{T} \vdash \varphi \leftrightarrow \varphi'$.*

Exercise 6.21. *Prove the above lemma.*

Exercise 6.22. *Let $\mathcal{T}$, $\mathcal{L}$ and i be as in Lemma 6.20 above. Show that $\mathcal{T}$ proves the $\Sigma_{i+1}^B(\mathcal{L})$-**REPL** axiom scheme.*

The next lemma shows that $\mathbf{V}^1$ proves the Σ_1^B-**REPL** axiom scheme. It is important to note that the analogous statement does not hold for $\mathbf{V}^0$: we will prove later (see Section 8F) that $\mathbf{V}^0$ *does not* prove the Σ_0^B-**REPL** axiom scheme. (It follows from Exercise 6.22 that over $\mathbf{V}^0$ Σ_1^B-**REPL** follows from Σ_0^B-**REPL**, i.e., the two axioms schemes are equivalent over $\mathbf{V}^0$.) Also, we will introduce the universal theory **VPV** which characterizes **P** in the same way that $\mathbf{V}^1$ characterizes **P**, and we will show that it is unlikely that **VPV** proves Σ_1^B-**REPL**.

LEMMA 6.23. *Let $\mathcal{T}$ be an extension of $\mathbf{V}^0$, where the vocabulary $\mathcal{L}$ of $\mathcal{T}$ is either $\mathcal{L}_A^2$ or $\mathcal{L}_A^2 \cup \{Row\} \subseteq \mathcal{L}$). Suppose that $\mathcal{T}$ proves the $\Sigma_{i+1}^B(\mathcal{L})$-**IND** axiom scheme, for some $i \geq 0$. Then $\mathcal{T}$ also proves the $\Pi_i^B(\mathcal{L})$-**REPL** axiom scheme.*

PROOF. Let φ be a $\Pi_i^B(\mathcal{L})$ formula. We will show that $\mathcal{T}$ proves (103). Intuitively, the RHS of (103) is the formula which states the existence of an array Z having b rows, whose x-th row $Z^{[x]}$ satisfies $\varphi(x, Z^{[x]})$. We will prove by number induction the existence of the initial segments of Z, and hence derive the existence of Z.

Formally we need to make sure that the RHS of (103) is equivalent to a $\Sigma_{i+1}^B(\mathcal{L})$ formula. First consider the case where $i = 0$, so φ is a $\Sigma_0^B(\mathcal{L})$ formula. Let

$$\psi(z) \equiv \exists Z \leq \langle z, c\rangle \forall x \leq z\ (|Z^{[x]}| \leq c \wedge \varphi(x, Z^{[x]}))$$

Then $\psi(z)$ is a $\Sigma_1^B(\mathcal{L})$ formula and the RHS of (103) is just $\psi(b)$. Our task is to show in $\mathcal{T}$ that $\psi(z)$ holds for $z \leq b$, assuming the LHS of (103). This is proved in $\mathcal{T}$ by induction on $z \leq b$. For the base case, $\psi(0)$ follows from the LHS of (103) by putting $x = 0$. The induction step follows from the induction hypothesis and the LHS of (103), using Σ_0^B-**COMP**.

For the case where $i \geq 1$, note that when φ is a $\Pi_i^B(\mathcal{L})$ formula, the RHS of (103) is not really a $\Sigma_{i+1}^B(\mathcal{L})$ formula. But it is equivalent (in $\mathcal{T}$) to:

$$\exists Z \leq \langle b, c\rangle \forall Y \leq b\ (|Z^{[|Y|]}| \leq c \wedge \varphi(x, Z^{[|Y|]}))$$

which is equivalent to a $\Sigma_{i+1}^B(\mathcal{L})$ formula. Let ψ be the equivalent $\Sigma_{i+1}^B(\mathcal{L})$ formula, then we can use the same arguments as for the case $i = 0$. ⊣

From Exercise 6.22, Lemma 6.23, Corollary 6.1, Corollary 6.4, and Lemma 6.19 we have:

COROLLARY 6.24. *For $i \geq 1$, the theory $\mathbf{V}^i$ proves the $\mathbf{g}\Sigma_i^B$-**REPL** axiom scheme. For each $\mathbf{g}\Sigma_i^B$ (resp. $\mathbf{g}\Pi_i^B$) formula φ, there is a single-Σ_i^B (resp. single-Π_i^B) formula φ' such that $\mathbf{V}^i \vdash \varphi \leftrightarrow \varphi'$. Also $\mathbf{V}^i$ proves $\Sigma_0^B(\mathbf{g}\Sigma_i^B \cup \mathbf{g}\Pi_i^B)$-**IND**.*

6C.1. Extending $\mathbf{V}^1$ by Polytime Functions. By the Extension by Definition Theorem 3.30, if we extend $\mathbf{V}^1$ by a collection $\mathcal{L}$ of its Σ^1_1-definable functions (i.e., polytime functions), Δ^1_1-definable predicates (i.e., polytime predicates), and their defining axioms, then we obtain a conservative extension $\mathbf{V}^1(\mathcal{L})$ of $\mathbf{V}^1$. Here we want to show further that $\mathbf{V}^1(\mathcal{L})$ proves the $\Sigma^B_1(\mathcal{L})$-**COMP** axiom scheme. This is similar to the situation for $\mathbf{V}^0$, where it follows from Corollary 5.39 and Lemma 5.40 that $\mathbf{V}^0(\mathcal{L})$ is conservative over $\mathbf{V}^0$, and it proves the $\Sigma^B_0(\mathcal{L})$-**COMP** axiom scheme for a collection $\mathcal{L}$ of $\mathbf{AC}^0$ functions. Note that for the case of $\mathbf{V}^0$, the $\mathbf{AC}^0$ string functions are Σ^B_0-bit-definable in $\mathbf{V}^0$.

Here it suffices to show that any $\Sigma^B_1(\mathcal{L})$ formula is provably equivalent in $\mathbf{V}^1(\mathcal{L})$ to a $\Sigma^B_1(\mathcal{L}^2_A)$ formula. We will prove this by structural induction on the $\Sigma^B_1(\mathcal{L})$ formula. For the induction step, we use Corollary 6.24 above. More generally, we prove:

Lemma 6.25 (Σ^B_1-Transformation Lemma). *Let $\mathcal{T}$ be a polynomial-bounded theory over the vocabulary $\mathcal{L} \supseteq \mathcal{L}^2_A \cup \{Row\}$. Suppose that $\mathcal{T}$ proves $\Sigma^B_0(\mathcal{L})$-**REPL**. Let $\mathcal{T}'$ be the extension of $\mathcal{T}$ which is obtained by adding to $\mathcal{T}$ a $\Sigma^1_1(\mathcal{L})$-definable function or a $\Delta^1_1(\mathcal{L})$-definable predicate, and its defining axiom, and $\mathcal{L}'$ be the vocabulary of $\mathcal{T}'$. Then*

(a) *$\mathcal{T}'$ is conservative over $\mathcal{T}$, and $\mathcal{T}'$ is polynomial-bounded;*
(b) *For any $\Sigma^B_1(\mathcal{L}')$ formula φ^+, there is a $\Sigma^B_1(\mathcal{L})$ formula φ so that $\mathcal{T}' \vdash \varphi^+ \leftrightarrow \varphi$;*
(c) *For any $\Sigma^B_0(\mathcal{L}')$ formula φ^+, there are a $\Sigma^B_1(\mathcal{L})$ formula φ_1 and a $\Pi^B_1(\mathcal{L})$ formula φ_2 so that $\mathcal{T}' \vdash \varphi^+ \leftrightarrow \varphi_1$, and $\mathcal{T} \vdash \varphi_1 \leftrightarrow \varphi_2$;*
(d) *$\mathcal{T}'$ proves the $\Sigma^B_1(\mathcal{L}')$-**REPL** axiom scheme.*

Indeed, by Exercise 5.55, the formulas φ and φ_1 can be taken to be *single*-$\Sigma^B_1(\mathcal{L})$ formulas, and φ_2 can be taken to be a *single*-$\Pi^B_1(\mathcal{L})$ formula.

Proof. For (a) the conservativity of $\mathcal{T}'$ over $\mathcal{T}$ follows from the Extension by Definition Theorem 3.30. Also, $\mathcal{T}'$ is polynomial-bounded because $\mathcal{T}$ is, and the Σ^1_1-definable functions of $\mathcal{T}$ are polynomially bounded (Corollary 5.29).

Part (b) follows from (c), and (d) follows from (c) and Exercise 6.22 (for the case $i = 0$). We prove (c) for the case of extending $\mathcal{T}$ by a Σ^1_1-definable string function. The case of adding a Σ^1_1-definable number function or a Δ^1_1-definable predicate is similar, and is left as an exercise.

Let F be the $\Sigma^1_1(\mathcal{L})$-definable function in $\mathcal{T}$. Since $\mathcal{T}$ is a polynomial-bounded theory, F is polynomially bounded in $\mathcal{T}$, and is $\Sigma^B_1(\mathcal{L})$-definable in $\mathcal{T}$ (Corollary 5.29). So there is a $\Sigma^B_1(\mathcal{L})$ formula $\varphi_F(\vec{x}, \vec{X}, Y)$ such that

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \varphi_F(\vec{x}, \vec{X}, Y) \tag{104}$$

and

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists ! Y \le t \varphi_F(\vec{x}, \vec{X}, Y) \tag{105}$$

By Lemma 6.19, it suffices to prove a simpler statement, i.e., that there exist a $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formula φ_1 and a $\mathbf{g}\Pi_1^B(\mathcal{L})$ formula φ_2 such that $\mathcal{T}' \vdash \varphi^+ \leftrightarrow \varphi_1$ and $\mathcal{T} \vdash \varphi_1 \leftrightarrow \varphi_2$. We prove this by induction on the nesting depth of F in φ^+. For the base case, F does not occur in φ^+, and there is nothing to prove. For the induction step, first we prove:

Claim. Suppose that for each atomic sub-formula ψ of φ^+, there are a $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formula ψ_1 and a $\mathbf{g}\Pi_1^B(\mathcal{L})$ formula ψ_2 so that $\mathcal{T}' \vdash \psi^+ \leftrightarrow \psi_1$ and $\mathcal{T} \vdash \psi_1 \leftrightarrow \psi_2$. Then there are a $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formula φ_1 and a $\mathbf{g}\Pi_1^B(\mathcal{L})$ formula φ_2 so that $\mathcal{T}' \vdash \varphi^+ \leftrightarrow \varphi_1$ and $\mathcal{T} \vdash \varphi_1 \leftrightarrow \varphi_2$.

We prove the claim by structural induction on φ^+. The base case holds trivially. The induction step is immediate from definition of $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formulas and the DeMorgan's laws.

Now we return to the proof of the induction step for (c). By the claim, it suffices to consider the atomic formulas over $\mathcal{L}'$. We can reduce the nesting depth of F as follows. The maximum nesting depth of F is the depth of F in (different) terms of the form $F(\vec{s}, \vec{T})$, where $\vec{s}$, $\vec{T}$ are terms with less nesting depth of F. We will show how to eliminate one such term from φ^+. In the general case all such terms can be eliminated using the same method. Write φ^+ as $\varphi^+(F(\vec{s}, \vec{T}))$. Then using (104) and (105) it is easy to see that (writing t for $t(\vec{s}, \vec{T})$):

$$\mathcal{T}' \vdash \varphi^+(F(\vec{s}, \vec{T})) \leftrightarrow \exists Y \le t(\varphi_F(\vec{s}, \vec{T}, Y) \wedge \varphi^+(Y))$$

and

$$\mathcal{T}' \vdash \exists Y \le t(\varphi_F(\vec{s}, \vec{T}, Y) \wedge \varphi^+(Y)) \leftrightarrow \forall Y \le t(\varphi_F(\vec{s}, \vec{T}, Y) \supset \varphi^+(Y))$$

The last line has the form $\mathcal{T}' \vdash \varphi_1' \leftrightarrow \varphi_2'$, where φ_1' is equivalent to a $\Sigma_1^B(\mathcal{L}')$ formula and φ_2' is equivalent to a $\Pi_1^B(\mathcal{L}')$ formula. Further φ_1' and φ_2' have less nesting depth of F than $\varphi^+(F(\vec{s}, \vec{T}))$. By applying the induction hypothesis to the atomic sub-formulas, we obtain a $\mathbf{g}\Sigma_1^B(\mathcal{L})$ formula φ_1 and a $\mathbf{g}\Pi_1^B(\mathcal{L})$ formula φ_2 that satisfy the induction step. $\dashv$

Exercise 6.26. *Prove Lemma 6.25 (c) for the cases of extending $\mathcal{T}$ by a Σ_1^1-definable number function and a Δ_1^1-definable predicate.*

Corollary 6.27. *Suppose that $\mathcal{T}_0$ is a polynomial-bounded theory with vocabulary $\mathcal{L}_0 \supseteq \mathcal{L}_A^2 \cup \{Row\}$, and that $\mathcal{T}_0$ proves the $\Sigma_0^B(\mathcal{L}_0)$-**REPL** axiom scheme. Let $\mathcal{T}_0 \subset \mathcal{T}_1 \subset \mathcal{T}_2 \subset \dots$ be a sequence of extensions of $\mathcal{T}_0$ where each $\mathcal{T}_i$ has vocabulary $\mathcal{L}_i$ and each $\mathcal{T}_{i+1}$ is obtained from $\mathcal{T}_i$ by adding the defining axiom for a $\Sigma_1^1(\mathcal{L}_i)$-definable function or a $\Delta_1^1(\mathcal{L}_i)$-definable predicate. Let*

$$\mathcal{T} = \bigcup_{i \ge 0} \mathcal{T}_i$$

*Then $\mathcal{T}$ is a polynomial-bounded theory which is conservative over $\mathcal{T}_0$ and proves the $\Sigma_1^B(\mathcal{L})$-**REPL** axiom scheme, where $\mathcal{L}$ is the vocabulary of $\mathcal{T}$. Furthermore, each function in $\mathcal{L}$ is $\Sigma_1^1(\mathcal{L}_0)$-definable in $\mathcal{T}_0$, and* *each predicate in $\mathcal{L}$ is $\Delta_1^1(\mathcal{L}_0)$-definable in $\mathcal{T}_0$. Finally each $\Sigma_1^B(\mathcal{L})$ formula is provably equivalent in $\mathcal{T}$ to a $\Sigma_1^B(\mathcal{L}_0)$ formula.*

The corollary is proved using Lemma 6.25 by proving by induction on i that the analogous statement holds for each theory $\mathcal{T}_i$. The conservativity of $\mathcal{T}$ follows from the conservativity of each $\mathcal{T}_i$ by compactness.

The corollary can be applied to the case in which $\mathcal{T}_0 = \mathbf{V}^1$, since by Corollary 6.24, $\mathbf{V}^1$ proves Σ_1^B-**REPL**, and we may assume that $\mathcal{T}_1$ is $\mathbf{V}^1(Row)$. We will use Corollary 6.27 for $\mathcal{T}_0 = \mathbf{V}^1(Row)$ in Subsection 6D.2 when we prove the Witnessing Theorem for $\mathbf{V}^1$.

6D. The Witnessing Theorem for $\mathbf{V}^1$

To prove the $\Longrightarrow$ direction of Theorem 6.6, i.e., every Σ_1^1-definable function in $\mathbf{V}^1$ is a polytime function, we prove the Witnessing Theorem for $\mathbf{V}^1$ below. Recall that by the $\Longleftarrow$ direction, each polytime function has a Σ_1^1-defining axiom in $\mathbf{V}^1$.

Theorem 6.28 (Witnessing Theorem for $\mathbf{V}^1$). *Suppose that $\varphi(\vec{x},\vec{y},\vec{X},\vec{Y})$ is a Σ_0^B formula, and that*

$$\mathbf{V}^1 \vdash \forall\vec{x}\forall\vec{X}\exists\vec{y}\exists\vec{Y}\varphi(\vec{x},\vec{y},\vec{X},\vec{Y})$$

Then there are polytime functions $f_1,\ldots,f_k$, $F_1,\ldots,F_m$ so that

$$\mathbf{V}^1(f_1,\ldots,f_k,F_1,\ldots,F_m) \vdash \forall\vec{x}\forall\vec{X}\varphi(\vec{x},\vec{f}(\vec{x},\vec{X}),\vec{X},\vec{F}(\vec{x},\vec{X}))$$

A more general witnessing statement follows from this theorem and Corollary 6.27 and Lemma 6.19.

Corollary 6.29. *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L}$ which results from $\mathbf{V}^1$ by a sequence of extensions by Σ_1^1-definable functions and Δ_1^1-definable predicates. If*

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists Y\varphi(\vec{x},\vec{X},Y)$$

where φ is in $\mathbf{g}\Sigma_1^B(\mathcal{L})$ then there is a polytime function F such that

$$\mathcal{T}(F) \vdash \forall\vec{x}\forall\vec{X}\varphi(\vec{x},\vec{X},F(\vec{x},\vec{X}))$$

Example 6.30 (Prime Recognition). *Any polynomial time prime recognition algorithm (such as the one by Agrawal et al [1]) gives a predicate $Prime(X)$ which according to Corollary 6.8 is Δ_1^B definable in $\mathbf{V}^1$. It follows by the Witnessing Theorem that if $\mathbf{V}^1$ proves the correctness of the algorithm, then binary integers can be factored in polynomial time. Here correctness means*

$$Prime(X) \leftrightarrow \big(2 \le |X| \wedge \forall Y\forall Z(Y \times Z = X \supset (X = Y \vee X = Z))\big)$$

(Recall that $Y \times Z$ is Σ_1^1 definable in $\mathbf{V}^1$, by Exercise 6.11). In fact, the right-to-left direction of this correctness statement implies

$$\forall X\exists Y\exists Z\big((Y \times Z = X \wedge X \ne Y \wedge X \ne Z) \vee Prime(X) \vee |X| < 2\big)$$

Thus if $\mathbf{V}^1(Prime, \times)$ *proves correctness then polynomial time witnessing functions for* Y *and* Z *would provide proper factors for each nonprime* X *with* $|X| \geq 2$.

EXERCISE 6.31 (Prime Factorization). *Show that* $\mathbf{V}^1$ *proves that every binary integer* X *greater than 1 can be represented as a product of primes. Use the fact that* $\mathbf{V}^1$ *proves the* Σ_1^B-**MAX** *axioms (Corollary 5.8), where we are trying to maximize* k *such that for some string* $Y = \langle Z_1, \ldots, Z_k \rangle$ *with each* Z_i *a binary number* ≥ 2, $\prod Z_i = X$. *Explain why it does not follow from the Witnessing Theorem for* $\mathbf{V}^1$ *that binary integers can be factored into primes in polynomial time.*

As in the proof of the Witnessing Theorem for $\mathbf{V}^0$ (Subsection 5E.2), the Witnessing Theorem for $\mathbf{V}^1$ follows from the following special case.

LEMMA 6.32. *Suppose that* $\varphi(\vec{x}, \vec{X}, Y)$ *is a* Σ_0^B *formula such that*

$$\mathbf{V}^1 \vdash \forall\vec{x}\forall\vec{X}\exists Y \varphi(\vec{x}, \vec{X}, Y)$$

Then there is a polytime function F *so that*

$$\mathbf{V}^1(F) \vdash \forall\vec{x}\forall\vec{X}\varphi(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

Our first attempt to prove the lemma would be to consider an anchored $\mathbf{LK}^2$-$\mathbf{V}^1$ proof π of $\exists Y \leq t\ \varphi(\vec{x}, \vec{X}, Y)$, and proceed as in the proof of Lemma 5.65. In this case, however, a Σ_1^B-**COMP** axiom

$$\exists X \leq y \forall z < y(X(z) \leftrightarrow \varphi(z)) \tag{106}$$

is not in general provably equivalent to a Σ_1^B formula, because of the clause $\varphi(z) \supset X(z)$. So the $\mathbf{LK}^2$-$\mathbf{V}^1$ proof π could contain formulas which are not Σ_1^1. To get around this difficulty, we begin by showing that $\mathbf{V}^1$ can be axiomatized by Σ_1^B-**IND** and Σ_0^B-**COMP** instead of Σ_1^B-**COMP**. Consider the theory $\tilde{\mathbf{V}}^1$:

DEFINITION 6.33. *The theory* $\tilde{\mathbf{V}}^1$ *has vocabulary* $\mathcal{L}_A^2$ *and has the axioms of* $\mathbf{V}^0$ *and the* Σ_1^B-**IND** *axiom scheme.*

By Exercise 5.55, $\tilde{\mathbf{V}}^1$ can be axiomatized by $\mathbf{V}^0$ and the *single*-Σ_1^B-**IND** axiom scheme.

LEMMA 6.34. $\tilde{\mathbf{V}}^1$ *proves the* Σ_1^B-**REPL** *axioms.*

PROOF. Corollary 6.24 states this for $\mathbf{V}^1$, and the only properties of $\mathbf{V}^1$ used in the proof are that $\mathbf{V}^1$ extends $\mathbf{V}^0$ and proves the Σ_1^B-**IND** axioms. Hence the same proof works for $\tilde{\mathbf{V}}^1$. ⊣

THEOREM 6.35. *The theories* $\mathbf{V}^1$ *and* $\tilde{\mathbf{V}}^1$ *are the same.*

PROOF. By Corollary 6.1, $\mathbf{V}^1$ proves the Σ_1^B-**IND** axiom scheme. Therefore $\tilde{\mathbf{V}}^1 \subseteq \mathbf{V}^1$. It remains to prove the other direction.

As noted earlier, (106) is not in general equivalent to a Σ_1^B formula, so we cannot use Σ_1^B-**IND** directly on (106) to prove the existence of X.

We introduce the number function *numones*(y, X), which is the number of elements of X that are $< y$. Recall that $seq(u, Z) = (Z)^u$ is the $\mathbf{AC}^0$ function used for coding a finite sequence of numbers (Definition 5.56). The function *numones* has the defining axiom:

$$\text{(107)} \quad numones(y, X) = z \leftrightarrow$$
$$z \le y \wedge \exists Z \le 1 + \langle y, y\rangle \big((Z)^0 = 0 \wedge (Z)^y = z \wedge$$
$$\forall u < y((X(u) \supset (Z)^{u+1} = (Z)^u + 1) \wedge (\neg X(u) \supset (Z)^{u+1} = (Z)^u))\big)$$

Here Z codes a sequence of $(y+1)$ numbers so that $(Z)^u = numones(u, X)$, for $u \le y$.

Exercise 6.36. (a) *Show that* (107) *is a* Σ_1^B *definition of numones in* $\tilde{\mathbf{V}}^1$*, i.e., show that* $\tilde{\mathbf{V}}^1 \vdash \forall y \forall X \exists ! z \varphi_{numones}(y, z, X)$*, where* $\varphi_{numones}(y, z, X)$ *is the RHS of* (107).
(b) *Show that the following is a theorem of* $\tilde{\mathbf{V}}^1(numones)$.

$$\exists x < y\big(X(x) \wedge \neg Y(x) \wedge \forall u < y\ (u \ne x \supset (X(u) \leftrightarrow Y(u)))\big)$$
$$\supset numones(y, X) = numones(y, Y) + 1$$

Although (106) may not be Σ_1^B, the result of replacing $\leftrightarrow$ by $\supset$ is Σ_1^B. Motivated by this, we define

$$\eta(y, Y) \equiv \forall z < y(Y(z) \supset \varphi(z))$$

Let X be the set satisfying the existential quantifier in (106). Then $\eta(y, Y)$ asserts $Y \subseteq X$.

Now consider the formula

$$\psi(w, y) \equiv \exists Y \le y\big(\eta(y, Y) \wedge w = numones(y, Y)\big)$$

For any w and Y that satisfy $\psi(w, y)$, we have $w \le numones(y, X)$, and $Y = X$ iff Y satisfies $\psi(w_0, y)$, where w_0 is the maximal value for w. To formalize this argument, we need the Σ_1^B-**MAX** axioms, which by Definition 5.5 have the form

$$\varphi(0) \supset \exists x \le y\big(\varphi(x) \wedge \neg \exists z \le y(x < z \wedge \varphi(z))\big)$$

where $\varphi(x)$ is Σ_1^B. These are provable in $\mathbf{V}^1$ by Corollary 6.1.

Exercise 6.37. *Show that* $\tilde{\mathbf{V}}^1$ *proves the* Σ_1^B-**MAX** *axioms. Hint: Apply* Σ_1^B-**IND** *to the formula* $\varphi'(x)$ *given by*

$$\exists z \le y\big(x \le z \wedge \varphi(z)\big)$$

Since *numones* is Σ_1^1-definable in $\tilde{\mathbf{V}}^1$, it follows from Lemmas 6.24 and 6.25 that $\tilde{\mathbf{V}}^1(numones)$ is a conservative over $\tilde{\mathbf{V}}^1$ and proves that every $\Sigma_1^B(numones)$-formula is equivalent to some Σ_1^B-formula. Hence by Exercise 6.37, $\tilde{\mathbf{V}}^1(numones)$ proves the Σ_1^B-**MAX**$(numones)$ axioms.

Now apply Σ_1^B-**MAX** for the case $\varphi(w)$ is $\psi(w, y)$. Arguing in $\tilde{\mathbf{V}}^1$, we have $\psi(0, y)$ (take Y to be the empty set), and hence there is a maximum $w_0 \leq y$ satisfying $\psi(w_0, y)$. We argued above that the set Y corresponding to w_0 is the set X satisfying (106), and this argument can be formalized in $\tilde{\mathbf{V}}^1$ using Exercise 6.36. $\dashv$

6D.1. The Sequent System $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$. We now convert $\tilde{\mathbf{V}}^1$ into an equivalent sequent system $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$, which is defined essentially as in Definition 4.27 (for $\Phi = \tilde{\mathbf{V}}^1$), but now we replace the Σ_1^B-**IND** axiom scheme by the Σ_1^B-**IND** inference rule. Recall that for $\mathbf{LK}^2$, terms do not contain any bound variables $x, y, z, \ldots, X, Y, Z, \ldots$, and formulas do not contain free occurrence of any bound variable, or bound occurrence of any free variable.

Definition 6.38 (The **IND** Rule). *For a set Φ of formulas, the Φ-**IND** rule consists of the inferences of the form*

$$\frac{\Gamma, A(b) \longrightarrow A(b+1), \Delta}{\Gamma, A(0) \longrightarrow A(t), \Delta} \tag{108}$$

where A is a formula in Φ.
Restriction. *The variable b is called an* eigenvariable *and does not occur in the bottom sequent.*

Notation. In general, we refer to an $\mathbf{LK}^2$ proof where the **IND** rule is allowed as an $\mathbf{LK}^2$+**IND** proof.

In this chapter we are mainly interested in this rule for the case where Φ is Σ_1^B.

Definition 6.39 ($\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$). *The rules of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ consist of the rules of $\mathbf{LK}^2$ (Section 4D), together with the single-Σ_1^B-**IND** rule* (108). *The non-logical axioms of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ are sequents of the form $\longrightarrow A$, where A is any term substitution instance of a Σ_0^B-**COMP** axiom or a 2-**BASIC** axiom (Figure 2) or an $\mathbf{LK}^2$ equality axiom (Definition 4.26).*

Thus the axioms of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ are the same as those of $\mathbf{LK}^2$-$\mathbf{V}^0$.

The notion of an *anchored* $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proof generalizes the notion of an anchored $\mathbf{LK}^2$ proof (Definition 4.29) to include the rule Σ_1^B-**IND** above. Note that the axioms of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ are closed under substitution of terms for free variables. More generally, we have:

Definition 6.40 (Anchored $\mathbf{LK}^2$ Proof with the **IND** Rule). *An $\mathbf{LK}^2$ proof π where the rule Φ-**IND** is allowed, for some set Φ of formulas, is said to be* anchored *provided that every cut formula in π occurs also either as a formula in the non-logical axioms of π, or as one of the formulas $A(0), A(t)$ in an instance of the rule Φ-**IND*** (108).

The following exercise is to show the soundness of $\mathbf{LK}^2$+**IND** in general. It follows that $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ is sound, in the sense that the sequents provable in $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ are also provable in $\tilde{\mathbf{V}}^1$.

Exercise 6.41 (Soundness of $\mathbf{LK}^2$+$\mathbf{IND}$). *Let Ψ and Φ be sets of formulas. Show that if A has an $\mathbf{LK}^2$-Ψ proof, where the Φ-$\mathbf{IND}$ rule is allowed, then A is a theorem of the theory axiomatized by $\Psi \cup \Phi$-$\mathbf{IND}$.*

To prove the Witnessing Theorem for $\mathbf{V}^1$, we first prove that every theorem of $\tilde{\mathbf{V}}^1$ has an anchored $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proof. This is stated more generally as follows.

Theorem 6.42 (Anchored Completeness for $\mathbf{LK}^2$+$\mathbf{IND}$). *Let Ψ and Φ be two sets of formulas over a vocabulary $\mathcal{L}$, and suppose that Ψ includes formulas which are the semantic equivalents of the equality axioms (Definition 4.26). Suppose that $\mathcal{T}$ is the theory which is axiomatized by the set of axioms $\Psi \cup \Phi$-$\mathbf{IND}$. Let Ψ' and Φ' be the closures of Ψ and Φ respectively under substitution of terms for free variables. Then for any theorem A of $\mathcal{T}$ there is an anchored $\mathbf{LK}^2$-Ψ' proof of $\longrightarrow A$ where instances of the Φ'-$\mathbf{IND}$ rule are allowed.*

To apply this to $\tilde{\mathbf{V}}^1$ (and hence to $\mathbf{V}^1$, by Theorem 6.35) take $\mathcal{T} = \tilde{\mathbf{V}}^1$, $\Phi = \Sigma_1^B$ and $\Psi = 2\text{-}\mathbf{BASIC} \cup \Sigma_0^B\text{-}\mathbf{COMP}$.

Corollary 6.43. *Every theorem of $\mathbf{V}^1$ has an anchored $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proof.*

Proof of Theorem 6.42. We refer to an anchored $\mathbf{LK}^2$+$\mathbf{IND}$ proof of the type stated above simply as an anchored $\mathbf{LK}^2$-Ψ' proof, with the understanding that the Φ'-$\mathbf{IND}$ rule is allowed. We will show that if a sequent $\Gamma \longrightarrow \Delta$ is a theorem of $\mathcal{T}$ (in the sense that its semantic formula given in Definition 2.35 is a theorem of $\mathcal{T}$), then there is an anchored $\mathbf{LK}^2$-Ψ' proof of $\Gamma \longrightarrow \Delta$.

Recall the proof of the Completeness Lemma 2.43 and the Anchored $\mathbf{LK}$ Completeness Theorem 2.47 (outlined in Exercise 2.48). Our proof here is by the same method, i.e., for a sequent $\Gamma \longrightarrow \Delta$ purportedly provable in $\mathcal{T}$, we try to find an anchored $\mathbf{LK}^2$-Ψ' proof of $\Gamma \longrightarrow \Delta$. Our procedure guarantees that in the case where no such proof is found, then we will be able to define a structure that satisfies $\mathcal{T}$ but does not satisfy $\Gamma \longrightarrow \Delta$. Thus we can conclude that $\Gamma \longrightarrow \Delta$ is not provable in $\mathcal{T}$.

We begin by listing all formulas, variables, and terms. In two-sorted logic, there are two sorts of terms: number terms and string terms. So we enumerate all quadruples $\langle A_i, c_j, t_k, T_\ell \rangle$, where A_i is an $\mathcal{L}$-formula, c_j is a free variable, t_k is an $\mathcal{L}$-number term, and T_ℓ is an $\mathcal{L}$-string term. (The term t_k contains only free variables $a, b, \ldots, \alpha, \beta, \ldots$.) The enumeration is such that each quadruple $\langle A_i, c_j, t_k, T_\ell \rangle$ occurs infinitely many times.

The proof π is constructed in stages. Initially π consists of just the sequent $\Gamma \longrightarrow \Delta$. At each stage we expand π by applying the $\mathbf{IND}$ rule and the rules of $\mathbf{LK}^2$ in reverse. We follow the 3 steps listed in the proof of the Completeness Lemma, with necessary modifications.

The idea is that if this proof-building procedure does not terminate, then the term model $\mathcal{M}$ derived from it satisfies $\mathcal{T}$ but not $\Gamma \longrightarrow \Delta$. In particular, in this case the procedure produces an infinite sequence of sequents $\Gamma_n \longrightarrow \Delta_n$ (starting with $\Gamma \longrightarrow \Delta$), and $\mathcal{M}$ is defined in such a way that it satisfies every formula in the antecedents Γ_n, and falsifies every formula in the succedents Δ_n.

We modify the notion of an *active sequent* as follows.

Notation. In the process of constructing π, a sequent is said to be *active* if it is active as defined on page 25, and it cannot be derived from $\longrightarrow B$ for some B in Ψ' using only the exchange and weakening rules.

We use one quadruple $\langle A_i, c_j, t_k, T_\ell \rangle$ of our enumeration in each stage. Here are the details for the next stage in general.

Let $\langle A_i, c_j, t_k, T_\ell \rangle$ be the next quadruple in our enumeration. Call A_i the *active formula* for this stage.

Step 1: If A_i is in Ψ', then expand π at every active sequent $\Gamma' \longrightarrow \Delta'$ as follows:

$$\dfrac{A_i, \Gamma' \longrightarrow \Delta' \qquad \dfrac{\longrightarrow A_i}{\Gamma' \longrightarrow \Delta', A_i}\ \textbf{weakening}}{\Gamma' \longrightarrow \Delta'}\ \textbf{cut}$$

Step 2a: If $A_i \in \Phi$ and c_j has one or more free occurrences in A_i, then we incorporate an application of the **IND** rule for A_i. Let b be a new free variable that does not occur in the proof so far, and let $A(b)$ be the result of substituting b for c_j in A_i. For each active sequent $\Gamma' \longrightarrow \Delta'$ we expand π as follows:

$$\dfrac{\Gamma' \longrightarrow \Delta', A(0) \qquad \dfrac{\dfrac{A(t_j), \Gamma' \longrightarrow \Delta'}{A(t_j), \Gamma', A(0) \longrightarrow \Delta'} \qquad \dfrac{\Gamma', A(b) \longrightarrow A(b+1), \Delta'}{\Gamma', A(0) \longrightarrow A(t_j), \Delta'}}{\Gamma', A(0) \longrightarrow \Delta'}}{\Gamma' \longrightarrow \Delta'}$$

Here the top-right inference is by the Φ-**IND** rule, and the three thick lines are for the weakening, cut and exchange rules (with cut formulas $A(0)$, $A(t_j)$).

Step 2b: Proceed as in the **Step** 2 in the proof of the Anchored **LK** Completeness Lemma 2.43. Here we use the string term T_k in our enumeration for the string quantifiers, in addition to the number term t_j which is for the number quantifiers, just as in the mentioned proof.

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

It is easy to verify that if the above procedure terminates, then the resulting proof π is an anchored $\mathbf{LK}^2$-Ψ' proof of $\Gamma \longrightarrow \Delta$. It remains to show that if the procedure does not halt, then the sequent $\Gamma \longrightarrow \Delta$ is not a logical consequence of $\mathcal{T}$. This is similar as for the Completeness Lemma 2.43, and is left as an exercise. ⊣

Exercise 6.44. *Complete the proof of the Anchored Completeness Lemma for* $\mathbf{LK}^2$+**IND** *above by constructing, in the case where the procedure does not terminate, a term model* $\mathcal{M}$ *(see Definition 2.45) that satisfies* $\mathcal{T}$ *but not the sequent* $\Gamma \longrightarrow \Delta$*. The two equality relations* $=_1$ *and* $=_2$ *are not necessarily interpreted as true equality in the term model, but by our assumption on* Ψ *the equality axioms of Definition 4.26 are satisfied, so the equivalence classes of terms form a true model. Also note that the occurrences of* $A(0)$ *in the antecedent of the construction for* **Step** *2a disappear from the sequents above them, so the term model must be defined in such a way that* $A(0)$ *is not necessarily satisfied. Show nevertheless that the* Φ-**IND** *axioms are satisfied.*

Effectively we have shown that any $\mathbf{LK}^2$ proof with axioms from $\mathcal{T}$ can be transformed into an anchored $\mathbf{LK}^2$+**IND** proof with axioms only from Ψ'. The advantage of the latter type of **LK** proofs is that the cut formulas are now essentially from $\Phi \cup \Psi'$, instead of the instances of Φ-**IND** $\cup\, \Psi$. In the case of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proofs, the cut formulas are restricted to Σ_1^B formulas (indeed, *single*-Σ_1^B formulas), while normally, an $\mathbf{LK}^2$ proof with axiom from $\tilde{\mathbf{V}}^1$ (Definition 2.40) contains cut formulas which are in general not Σ_1^B. This property of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proofs is important for our proof of the Witnessing Theorem for $\mathbf{V}^1$ that we present in the next section.

Proposition 6.45 (Subformula Property of $\mathbf{LK}^2$+**IND**). *Suppose that* Ψ *and* Φ *are sets of formulas, both of which are closed under substitution of terms for free variables. Suppose that* π *is an anchored* $\mathbf{LK}^2$-Ψ *proof of* $\mathcal{S}$*, where the* Φ-**IND** *rule is allowed. Then every formula in every sequent of* π *is a sub-formula of a formula in* $\mathcal{S}$ *or in* $\Psi \cup \Phi$.

6D.2. Proof of the Witnessing Theorem for $\mathbf{V}^1$. Now we prove the Witnessing Theorem for $\mathbf{V}^1$, using the same method as for the proof of the Witnessing Theorem for $\mathbf{V}^0$ (Subsection 5E.2). Here it suffices to prove Lemma 6.32.

Suppose that $\exists Z\varphi(\vec{a}, \vec{\alpha}, Z)$ is a Σ_1^1 theorem of $\mathbf{V}^1$, where φ is a Σ_0^B formula. Then by the Anchored $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ Completeness Theorem 6.42, there is an anchored $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proof π of $\exists Z\varphi(\vec{a}, \vec{\alpha}, Z)$. We may assume that π is in free variable normal form, where now Definition 2.39 is modified to allow applications of the Σ_1^B-**IND** rule to eliminate a variable from a sequent (in addition to $\forall$-right and $\exists$-left). By the Subformula Property of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ (Proposition 6.45), the formulas in π are Σ_1^1 formulas, and in fact they are Σ_0^B formulas or *single*-Σ_1^1 formulas. As a result, every sequent in π has the form (81):

(109)
$$\exists X_1\theta_1(X_1), \ldots, \exists X_m\theta_m(X_m), \Gamma \longrightarrow \Delta, \exists Y_1\psi_1(Y_1), \ldots, \exists Y_n\psi_n(Y_n)$$

for $m, n \geq 0$, where θ_i and ψ_j and all formulas in Γ and Δ are Σ_0^B.

We will prove by induction on the depth in π of a sequent $\mathcal{S}$ of the form (109) that there is a finite collection of polytime functions

$$\mathcal{L} = \{F_1, \dots, F_n, \dots\}$$

so that $\mathbf{V}^1(\mathcal{L})$ proves the (semantic equivalent of the) sequent

$$(110) \qquad \mathcal{S}' =_{\text{def}} \theta_1(\beta_1), \dots, \theta_m(\beta_m), \Gamma \longrightarrow \Delta, \psi_1(F_1), \dots, \psi_n(F_n)$$

i.e., there is an $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L})$ proof of $\mathcal{S}'$. Here F_i stands for $F_i(\vec{a}, \vec{\alpha}, \vec{\beta})$, and $\vec{a}, \vec{\alpha}$ is a list of exactly those variables with free occurrences in $\mathcal{S}$. (This list may be different for different sequents.) Also $\beta_1, ..., \beta_m$ are distinct new free variables corresponding to the bound variables $X_1, ..., X_m$, although the latter variables may not be distinct.

We proceed as in the proof of the Witnessing Theorem for $\mathbf{V}^0$ in Section 5E.2 by considering the cases where $\mathcal{S}$ is an axiom of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ (i.e., an axiom of $\mathbf{V}^0$), or $\mathcal{S}$ is generated using inference rules of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$. The case of the non-logical axioms or the introduction rules for $\neg, \wedge, \vee$ and bounded number quantifiers are dealt with just as in **Cases I** – **VIII** in the proof for $\mathbf{V}^0$. Here we will consider the only new case, i.e., the case of the Σ_1^B-**IND** rule. This is the one that causes the introduction of non-$\mathbf{AC}^0$ witnessing functions.

Case IX: $\mathcal{S}$ is obtained by an application of the Σ_1^B-**IND** rule. Then $\mathcal{S}$ is the bottom sequent of

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda, \exists X \leq r(b)\psi(b, X) \longrightarrow \exists X \leq r(b+1)\psi(b+1, X), \Pi}{\Lambda, \exists X \leq r(0)\psi(0, X) \longrightarrow \exists X \leq r(t)\psi(t, X), \Pi}$$

where b does not occur in $\mathcal{S}$, and ψ is Σ_0^B.

By the induction hypothesis for the top sequent $\mathcal{S}_1$, there is a finite collection $\mathcal{L}$ of polytime functions, and a polytime function $G(b, \beta) \in \mathcal{L}$ (suppressing arguments for the other variables present) such that $\mathbf{V}^1(\mathcal{L})$ proves the sequent S_1', which is

(111)

$$\Lambda', |\beta| \leq r(b) \wedge \psi(b, \beta) \longrightarrow |G(b, \beta)| \leq r(b+1) \wedge \psi(b+1, G(b, \beta)), \Pi'$$

Note that by the variable restriction, b and β do not occur in Λ', and can only occur in Π' as arguments to witnessing functions $F_i(b, \beta)$.

We define the witness function $\hat{G}(t, \beta)$ for the formula $\exists X \leq r(t)\psi(t, X)$ in the succedent of $\mathcal{S}$ by limited recursion (Definition 6.15) as follows:

$$(112) \qquad \hat{G}(0, \beta) = \beta$$

$$(113) \qquad \hat{G}(z+1, \beta) = (G(z, \hat{G}(z, \beta)))^{<r(z+1)}$$

Since G is a polytime function, by Cobham's Theorem 6.16, $\hat{G}$ is also a polytime function.

Let $F_1^1(b,\beta),\ldots,F_m^1(b,\beta)\in\mathcal{L}$ be the witnessing functions in Π'. Consider the sequent

$$(114)\quad \Lambda', |\hat{G}(b,\beta)|\le r(b)\wedge\psi(b,\hat{G}(b,\beta))\longrightarrow |\hat{G}(b+1,\beta)|\le r(b+1)\wedge\psi(b+1,\hat{G}(b+1,\beta)),\Pi''$$

which is obtained from (111) by substituting $\hat{G}(b,\beta)$ for β, and writing $\hat{G}(b+1,\beta)$ for $G(b,\hat{G}(b,\beta))$ (using (113)). In particular, Π'' is obtained from Π' by replacing each witnessing function $F_i^1(b,\beta)$ for $\mathcal{S}_1$ by $F_i^2(b,\beta)$, where

$$F_i^2(b,\beta)=F_i^1(b,\hat{G}(b,\beta))\qquad(1\le i\le m)$$

Let $\mathcal{L}'=\mathcal{L}\cup\{\hat{G},F_1^2,\ldots,F_m^2\}$. Then since (111) is a theorem of $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L})$, (114) is a theorem of $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}')$. Note that (114) is of the form

$$(115)\qquad \Lambda',\rho(b,\beta)\longrightarrow\rho(b+1,\beta),\Pi''$$

where

$$\rho(b,\beta)\equiv|\hat{G}(b,\beta)|\le r(b)\wedge\psi(b,\hat{G}(b,\beta))$$

Here ρ is a $\Sigma_0^B(\mathcal{L}')$ formula.

Notice that in Π'', b occurs (only) as an argument to F_i^2. So we cannot apply the **IND** rule to (115). Moreover, b should not occur in our desired sequent $\mathcal{S}'$. We remove b from Π'' by introducing the number function h:

$$h(\beta)=\min y<t\ \neg\rho(y+1,\beta)$$

i.e., h has the $\Sigma_0^B(\mathcal{L}')$-defining axiom

$$(116)\quad h(\beta)=y\leftrightarrow y\le t\wedge(y=t\vee\neg\rho(y+1,\beta))\wedge\forall z<y\rho(z+1,\beta)$$

Then h is a polytime function, and can be defined from $\rho(b,\beta)$ using limited recursion. Define for each i, $1\le i\le m$,

$$F_i(\beta)=F_i^2(h(\beta),\beta)$$

Then F_i is a polytime function. Let Π''' be Π'' with each witnessing function $F_i^2(b,\beta)$ replaced by $F_i(\beta)$. Also define (by composition):

$$G^*(\beta)=\hat{G}(t,\beta)$$

Now define $\mathcal{S}'$ to be the sequent:

(117)

$$\mathcal{S}'=\ \Lambda',|\beta|\le r(0)\wedge\psi(0,\beta)\longrightarrow|G^*(\beta)|\le r(t)\wedge\psi(t,G^*(\beta)),\Pi'''$$

Then $\mathcal{S}'$ is of the form (110). It remains to show that $\mathcal{S}'$ is provable in $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$, where $\mathcal{L}''$ is $\mathcal{L}'$ together with the new functions in $\mathcal{S}'$, i.e., $\mathcal{L}''=\mathcal{L}'\cup\{h,F_1,\ldots,F_m,G^*\}$.

First, by (112) the sequent (117) is equivalent to

$$(118)\qquad \Lambda',\rho(0,\beta)\longrightarrow\rho(t,\beta),\Pi'''$$

Then by replacing b in (115) with $h(\beta)$, $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$ proves

$$\Lambda', \rho(h(\beta), \beta) \longrightarrow \rho(h(\beta)+1, \beta), \Pi''' \tag{119}$$

Next, by the definition of h (116), $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$ proves the sequents

$$\rho(0, \beta) \longrightarrow \rho(h(\beta), \beta) \qquad \text{and} \qquad \rho(h(\beta)+1, \beta) \longrightarrow \rho(t, \beta)$$

From this and (119), it follows that $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$ proves (118), and hence (117). □

6E. Notes

Our theory $\mathbf{V}^1$ is essentially Zambella's Theory Σ_1^p-*comp* in [85], and is a variation of the theory V_1^1 in [55], which in turn is defined in the style of Buss's second-order theories [12]. It is a two-sorted version of Buss's $\mathbf{S}_2^1$. Our Σ_1^B formulas correspond to *strict* Σ_1^b formulas, but this does not really matter, as shown in Section 6C.

The Σ_1^1 Definability Theorem for $\mathbf{V}^1$ is essentially due to Buss [12] who proved it for his first-order theory $\mathbf{S}_2^1$. Exercise 6.31 ($\mathbf{V}^1$ proves the prime factorization theorem) is due to Jeřábek [46]. The interesting part of Theorem 6.35, that $\tilde{\mathbf{V}}^1$ proves the Σ_1^B-$\mathbf{COMP}$ axioms, is essentially Theorem 1 in [15].

Chapter 7

PROPOSITIONAL TRANSLATIONS

In Section 2A we presented Gentzen's Propositional Calculus **PK** and showed that **PK** is sound and complete; i.e. a propositional formula is valid iff it is provable in **PK**. In this chapter we introduce the general notion of *propositional proof system* (or simply *proof system*) and study its complexity. We are particularly interested in which families of tautologies have polynomial length proofs. In the (apparently unlikely) event that there is a polynomial $p(n)$ such that for every n, every tautology of length n has a proof in the system of length at most $p(n)$, then we say that the system is *polynomially bounded*. The question of existence (or nonexistence) of a polynomially bounded proof system is equivalent to the important complexity theory question of whether $\mathbf{NP} = co\text{-}\mathbf{NP}$.

Here our main interest is the relationship between bounded arithmetic and propositional proof systems. There is an extensive literature on the complexity of proof systems (see for example [55] and [9]) which we will barely touch.

One of our goals is to associate a proof system with each of our theories, such as $\mathbf{V}^0, \mathbf{V}^1, \ldots$. In this chapter we associate the proof system constant-depth **Frege** ($\mathbf{AC}^0$**-Frege**) with $\mathbf{V}^0$ and the system extended Frege (**eFrege**) with $\mathbf{V}^1$. Each Σ_0^B theorem in the theory can be translated into a family of tautologies which have polynomial size proofs in the corresponding proof system (the propositional translation), showing that the proof system is sufficiently powerful. On the other hand, in Chapter 10 we show that the soundness of a proof system is provable in the associated theory (the Reflection Principle), showing that the proof system is not too powerful.

In order to associate proof systems with other theories, and in order to translate $\Sigma_1^B, \Sigma_2^B, \ldots$ theorems of our theories (and not just Σ_0^B theorems), we need to generalize the propositional calculus to the quantified propositional calculus (QPC). This we do in Section 7C, and introduce the QPC proof system $\mathbf{G}$ and its subsystems $\mathbf{G}_0^\star, \mathbf{G}_1^\star, \ldots$ and $\mathbf{G}_0, \mathbf{G}_1, \ldots$. We show that for $i \geq 1$ each bounded theorem of $\mathbf{V}^i$ can be translated into a family of valid QPC formulas with polynomial size $\mathbf{G}_i^\star$ proofs. In Chapter 8 we introduce the hierarchy of theories $\mathbf{TV}^i$

and in Chapter 10 we show a similar relation between $\mathbf{TV}^i$ and $\mathbf{G}_i$. This and other results justify saying that $\mathbf{G}_i^\star$ is a kind of nonuniform version of $\mathbf{V}^i$ when considering Σ_i^B-theorems, but not for theorems in general. Similarly for $\mathbf{G}_i$ and $\mathbf{TV}^i$.

7A. Propositional Proof Systems

Recall (Section 2A) that a propositional formula is built from the logical constants $\bot, \top$ (for False, True), the propositional variables (or atoms) $p_1, p_2, \ldots$, connectives $\neg, \vee, \wedge$ and parentheses (,). Also, a tautology is a valid propositional formula (Definition 2.1). We assume that tautologies are coded as binary strings (or more properly finite subsets of $\mathbb{N}$) using some efficient encoding.

DEFINITION 7.1. *$TAUT$ is the set of (strings coding) propositional tautologies.*

A propositional proof system is a formal system for proving tautologies. An example is the system **PK** introduced in Section 2A, where a formal proof of a formula A is a tree of sequents, where the root is $\longrightarrow A$, the leaves are axioms, and the sequent at each internal node follows from its parent sequent(s) by a rule of inference. The soundness and completeness theorems state that $TAUT$ is exactly the set of formulas with formal **PK** proofs. Below we give a very general definition of proof system, and then explain how to make **PK** fit this definition.

DEFINITION 7.2 (Propositional Proof System). *A* propositional proof system *(or simply a* proof system*) is a polytime, surjective (onto) function*

$$F : \{0,1\}^* \longrightarrow TAUT$$

If $F(X) = A$, then we say that X is a proof of A in the system F.

The length of A is denoted $|A|$, and the length (or size) of the proof X is denoted $|X|$. A proof system F is said to be polynomially bounded *if there is a polynomial $p(n)$ such that for all tautologies A, there is a proof X of A in F such that $|X| \le p(|A|)$.*

Informally, a proof system F is polynomially bounded if every tautology has a short proof in F.

EXAMPLE 7.3. **PK** *can be treated as a proof system in the sense of Definition 7.2, because the function*

$$\mathbf{PK}(X) = \begin{cases} A & \text{if } X \text{ codes a } \mathbf{PK} \text{ proof of} \longrightarrow A \\ \top \ (\textit{True}) & \text{otherwise} \end{cases}$$

is a polytime function.

It is not known whether **PK** is polynomially bounded. In fact, the existence of a polynomially bounded proof system is equivalent to the assertion that $\mathbf{NP} = co\text{-}\mathbf{NP}$.

Theorem 7.4. *There exists a polynomially bounded proof system iff* $\mathbf{NP} = co\text{-}\mathbf{NP}$.

Proof. Since $TAUT$ is co-**NP**-complete, we have $\mathbf{NP} = co\text{-}\mathbf{NP}$ iff $TAUT \in \mathbf{NP}$.

($\Longrightarrow$) Suppose that F is a polynomially bounded proof system. Then by definition, there is a polynomial $p(n)$ such that

$$A \in TAUT \Leftrightarrow \exists X \leq p(|A|) F(X) = A$$

This shows that $TAUT \in \mathbf{NP}$: The witness for the membership of A in $TAUT$ is the proof X.

($\Longleftarrow$) If $TAUT \in \mathbf{NP}$, then there is a polytime relation $R(Y, A)$, and a polynomial $p(n)$ such that

$$A \in TAUT \Leftrightarrow \exists Y \leq p(|A|) R(Y, A)$$

Define the proof system F by

$$F(X) = \begin{cases} A & \text{if } X \text{ codes a pair } \langle Y, A\rangle \text{, and } R(Y, A) \\ \top & \text{otherwise} \end{cases}$$

Clearly F is a polynomially bounded proof system. ⊣

The general feeling among complexity theorists is that $\mathbf{NP} \neq co\text{-}\mathbf{NP}$, so the above theorem suggests that no proof system is polynomially bounded. In fact some weak proof systems, including resolution and bounded depth Frege systems (which is introduced below) have been proved to be not polynomially bounded. However it seems to be very difficult to prove this for the system **PK**. The system **PK** is p-equivalent (defined below) to a large class of proof systems, called **Frege** systems, which includes many standard proof systems described in logic text books. This adds interest to the problem of showing that **PK** is not polynomially bounded.

Also because **PK** is p-equivalent to the **Frege** proof systems, we will continue to work with **PK**, and will not define the **Frege** proof systems. Below we introduce **bPK** (*bounded depth* **PK**) and **ePK** (*extended* **PK**). They belong respectively to the families call *bounded depth* **Frege** and *extended* **Frege**.

Definition 7.5. *A proof system F_1 is said to* p-simulate *a proof system F_2 if there is a polytime function G such that $F_2(X) = F_1(G(X))$, for all X. Two proof systems F_1 and F_2 are said to be* p-equivalent *if F_1 p-simulates F_2, and vice versa.*

Thus F_1 p-simulates F_2 if any given F_2-proof X of a tautology A can be transformed (by a polytime function G) into an F_1-proof $G(X)$ of A.

Exercise 7.6. (a) *Show that the relation on proof systems "F_1 p-simulates F_2" is transitive and reflexive.*

(b) *Show that if F_1 p-simulates F_2, and F_2 is polynomially bounded, then F_1 is also polynomially bounded.*

7A.1. Treelike vs Daglike Proof Systems. Proofs in the system **PK** are trees. This tree structure is potentially inefficient, since each sequent in the proof can be used only once as a hypothesis for a rule, and if it needs to be used again in another part of the proof, then it must be rederived. This motivates allowing the proof structure to be a dag (directed acyclic graph), since this allows each sequent to be used repeatedly to derive others.

Definition 7.7 (Treelike vs Daglike). *A proof system is* treelike *if the structure of each proof is required to be a tree. The system is* daglike *if a proof is allowed to have the more general structure of a dag.*

In general a proof, whether treelike or daglike, can be represented as a sequence of "lines", where each line is the contents of some node in the proof. Each line is either an axiom or it follows from an earlier line or earlier lines in the proof (its parent or parents), and the line might be annotated to indicate this information. The proof is a tree if each sequent is a parent of at most one line.

The notions treelike and daglike can be used as adjectives to indicate different version of a proof system. For example, *treelike* **PK** is the same as **PK**, but *daglike* **PK** has the same axioms and rules as **PK**, but allows a proof to take the form of a dag.

The next result shows that for **PK** the distinction is not important. (But it is important for the system $\mathbf{G}_1^\star$ defined later in this chapter.)

Theorem 7.8 (Krajíček[52]). *Treelike* **PK** *p-simulates daglike* **PK**.

Proof. Recall that to each sequent $\mathcal{S} = A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell$ we associate the formula $A_\mathcal{S}$ which gives the meaning of $\mathcal{S}$:

$$A_\mathcal{S} \equiv \neg A_1 \vee \cdots \vee \neg A_k \vee B_1 \vee \cdots \vee B_\ell \tag{120}$$

Here it is not important how we parenthesize $A_\mathcal{S}$ (see Lemma 7.15). Also, there is a treelike **PK** derivation, whose size is bounded by a polynomial in the size of $\mathcal{S}$, of $\mathcal{S}$ from the sequent $\longrightarrow A_\mathcal{S}$.

Suppose that $\pi = \mathcal{S}_1, \ldots, \mathcal{S}_n$ is a daglike **PK** proof. We show:

Claim. The sequence

$$\longrightarrow A_{\mathcal{S}_1};\quad \longrightarrow (A_{\mathcal{S}_1} \wedge A_{\mathcal{S}_2});\quad \ldots;\quad \longrightarrow (A_{\mathcal{S}_1} \wedge \cdots \wedge A_{\mathcal{S}_n});\quad \longrightarrow A_{\mathcal{S}_n}$$

can be augmented to a treelike **PK** proof whose size is bounded by a polynomial in the length of π.

Again it is not important how the conjunctions $A_{\mathcal{S}_1} \wedge \cdots \wedge A_{\mathcal{S}_k}$ are parenthesized. The claim follows easily from the exercise below. $\dashv$

Exercise 7.9. (a) *Show that the following sequents have polynomial size cut-free treelike* **PK** *proofs:*

(i) $\longrightarrow A_{\mathcal{S}}$*, where* $\mathcal{S}$ *is any axiom of* **PK**.

(ii) $A \wedge B \longrightarrow B$*, for any* **PK** *formulas* A, B.

(iii) $A \wedge B \longrightarrow A \wedge B \wedge B$*, for any* **PK** *formulas* A, B.

(b) *Suppose that* $\mathcal{S}$ *is derived from* $\mathcal{S}_1$ *(and* $\mathcal{S}_2$*) by an inference rule of* **PK**. *Show that the following sequents have polynomial size cut-free treelike* **PK** *proofs, for any formula* A*:*

(i) $A \wedge A_{\mathcal{S}_1} \longrightarrow A \wedge A_{\mathcal{S}}$.

(ii) $A \wedge A_{\mathcal{S}_1} \wedge A_{\mathcal{S}_2} \longrightarrow A \wedge A_{\mathcal{S}}$.

The next result will be useful later in the chapter.

Lemma 7.10 (**PK***-Replacement Lemma). *Let* $A(p)$ *and* B *be propositional formulas, and let* $A(B)$ *be the result of substituting* B *for* p *in* $A(p)$. *Then for all propositional formulas* B_1, B_2*, the sequent*

$$(B_1 \leftrightarrow B_2) \longrightarrow (A(B_1) \leftrightarrow A(B_2))$$

has a cut-free treelike **PK** *proof of size bounded by a polynomial in its endsequent.*

Exercise 7.11. *Prove the lemma by giving (using structural induction on* $A(p)$*) cut-free treelike* **PK** *proofs of size polynomial in the size of the endsequents for the following sequents:*

$$A(B_1), B_1 \leftrightarrow B_2 \longrightarrow A(B_2), \qquad A(B_2), B_1 \leftrightarrow B_2 \longrightarrow A(B_1)$$

7A.2. The Pigeonhole Principle and Bounded Depth PK. To show that a proof system F is not polynomially bounded, it suffices to exhibit a family of tautologies that requires F-proofs of super-polynomial size. Similarly, to show that a proof system F_2 does not p-simulate a proof system F_1, it suffices to show the existence of a family of tautologies that has polynomial size F_1-proofs, but requires super-polynomial size F_2-proofs.

There is an important family of tautologies that formalizes the Pigeonhole Principle, which states that if $n+1$ pigeons are placed in n holes, then two pigeons will wind up in the same hole. The principle is formulated using the atoms

$$p_{i,j} \qquad (\text{for } 0 \le i \le n, 0 \le j < n)$$

where $p_{i,j}$ is intended to mean that pigeon i gets placed in hole j. First, the negation of the principle is expressed as an unsatisfiable propositional formula $\neg\mathbf{PHP}_n^{n+1}$, which is the conjunction of the following clauses:

$$(p_{i,0} \vee ... \vee p_{i,n-1}), \qquad 0 \le i \le n \tag{121}$$

$$(\neg p_{i,j} \vee \neg p_{k,j}), \qquad 0 \le i < k \le n,\ 0 \le j < n \tag{122}$$

Here, (121) says that the pigeon i is placed in some hole, and (122) says that two pigeons i and k are not placed in the same hole.

The Pigeonhole Principle itself is equivalent to the negation of $\neg\mathbf{PHP}_n^{n+1}$, which by applying DeMorgan's laws, can be expressed as follows.

DEFINITION 7.12 ($\mathbf{PHP}_n^{n+1}$). *The propositional formula* $\mathbf{PHP}_n^{n+1}$ *is defined to be*

$$(\bigwedge_{0\le i\le n} \bigvee_{0\le j<n} p_{i,j}) \supset \bigvee_{0\le i<k\le n, 0\le j<n} (p_{i,j} \wedge p_{k,j}) \tag{123}$$

Define $\mathbf{PHP} = \{\mathbf{PHP}_n^{n+1} : n \ge 1\}$.

Thus for each $n \ge 1$, $\mathbf{PHP}_n^{n+1}$ is a tautology.

In 1985 Armen Haken proved an exponential lower bound on the length of any Resolution refutation of $\neg\mathbf{PHP}_n^{n+1}$, one of the early important results in propositional proof complexity. On the other hand, in 1987 Buss presented polynomial size Frege proofs of $\mathbf{PHP}_n^{n+1}$. (Buss's proofs are based on the fact that there are propositional formulas $A_k(p_1, ..., p_n)$ of size polynomial in n which express the condition that at least k of $p_1, \dots, p_n$ are true.) It follows that Resolution does not p-simulate **Frege**. (While it is easy to show that **Frege** p-simulates Resolution.)

In fact the family **PHP** does not have polynomial size proofs in a stronger proof system called *bounded depth* **Frege** (also known as $\mathbf{AC}^0$-**Frege**). We will define **bPK**, a representative from these systems. First, we formally define the depth of a formula. Here we think of the connectives $\wedge, \vee$ as having arbitrary fan-in.

DEFINITION 7.13 (Depth of a Formula). *The* depth *of a formula A is the maximal number of times the connective changes in any path in the tree form of A.*

So in particular, the formula $(p_1 \vee \cdots \vee p_n)$ has depth 1, for any n, no matter how the parentheses are inserted. The depth of each clause (121) is 2, and the depth of the conjunction $\neg\mathbf{PHP}_n^{n+1}$ is 3.

DEFINITION 7.14 (Bounded Depth **PK**). *For each constant $d \in \mathbb{N}$ we define a d-**PK** proof to be a **PK** proof in which the* cut *formulas have depth at most d. We define a* bounded depth **PK** system *(or just* **bPK***) to be any system d-**PK** for $d \in \mathbb{N}$.*

Sometimes the definition for a d-**PK** proof is taken to be that *all* formulas in the proof have depth $\le d$. Our definition given above is more general: For proving a formula of depth $\le d$, the two definitions are the same, but here we allow d-**PK** proofs of any formula (not just formulas of depth $\leq d$). Indeed, since any tautology has a **PK** proof without using the cut rule (the **PK** Completeness Theorem 2.8), it follows that d-**PK** is complete, for any $d \geq 0$.

In general, we are not interested in the exact length of bounded depth **PK** proofs, but only interested in the length up to the application of a polynomial. Because of this and the next lemma, we will ignore how parentheses are placed in a disjunction $(A_1 \vee ... \vee A_n)$.

Lemma 7.15. *If A is some parenthesization of $(B_1 \vee ... \vee B_n)$, and A' is another such parenthesization, then there is a cut-free treelike* **PK** *proof of the sequent $A \longrightarrow A'$ consisting of $O(n^2)$ sequents, where each sequent has length at most that of the sequent $A \longrightarrow A'$.*

For example, we may have

$$A \equiv (B_1 \vee (B_2 \vee B_3)) \vee B_4), \qquad A' \equiv (B_1 \vee (B_2 \vee (B_3 \vee B_4)))$$

Proof. By repeated use of the rule $\vee$-left, it is easy to see that there is such a d-**PK** proof of the sequent

$$A \longrightarrow B_1, ..., B_n$$

Now repeated use of $\vee$-right (with exchanges) gives the desired d-**PK** proof. ⊣

In 1988 Ajtai proved that $\mathbf{PHP}_n^{n+1}$ does not have polynomial size bounded depth **Frege** proofs. (In fact he proved the result for a weaker version of PHP which asserts that there is no *bijection* mapping $(n+1)$ pigeons to n holes, see Section 9D.2.) This was strengthened by two groups a few years later to prove the following exponential lower bound, which remains one of the strongest lower bound results in propositional proof complexity.

Theorem 7.16 (Bounded Depth Lower Bound Theorem [8]). *For every $d \in \mathbb{N}$, every d-**PK** proof of $\mathbf{PHP}_n^{n+1}$ must have size at least*

$$2^{n^{\epsilon^d}}$$

where $\epsilon = 1/6$.

In view of Buss's upper bound for $\mathbf{PHP}_n^{n+1}$, we have

Corollary 7.17. *No bounded depth* **Frege** *system p-simulates any* **Frege** *system.*

The lower bound results in propositional proof complexity can be used to obtain independence results in the theories of bounded arithmetic. We will explain this in the next sections.

7B. Translating $\mathbf{V}^0$ to bPK

In this section we give evidence that the propositional proof system **bPK** is a kind of nonuniform version of the Σ_0^B-fragment of $\mathbf{V}^0$ (in Chapter 10 we give more evidence). Intuitively a $\mathbf{V}^0$ proof of a Σ_0^B formula is able to use concepts from the complexity class $\mathbf{AC}^0$. Recall from Subsection 4A that a language in nonuniform $\mathbf{AC}^0$ is specified by polynomial size family of bounded depth formulas. Thus the lines in a polynomial size family of **bPK** proofs express nonuniform $\mathbf{AC}^0$ concepts.

7B.1. Translating Σ_0^B Formulas. We begin by showing how to translate each Σ_0^B formula $\varphi(\vec{x}, \vec{X})$ into a polynomial size bounded depth family

$$\|\varphi(\vec{x}, \vec{X})\| = \{\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}] : \ \vec{m}, \vec{n} \in \mathbb{N}\}$$

of propositional calculus formulas, and then we show how to translate a $\mathbf{V}^0$ proof of a Σ_0^B formula into a polynomial size family of **bPK** proofs. Later we will show how to translate in general a bounded two-sorted formula into a polynomial size family of *quantified propositional calculus*. Here, the depth of each formula in the family $\|\varphi(\vec{x}, \vec{X})\|$ is bounded by a constant which depends only on φ.

We first explain the translation for a Σ_0^B formula $\varphi(X)$ which has a single free (string) variable X. We introduce propositional variables $p_0^X, p_1^X, \dots$, where p_i^X is intended to mean $X(i)$. The translation has the property that for each $n \in \mathbb{N}$, $\varphi(X)[n]$ is valid iff the formula $\forall X(|X| = \underline{n} \supset \varphi(X))$ is true in the standard model, where $\underline{n}$ is the n-th numeral. More generally, there is a one-one correspondence between truth assignments satisfying $\varphi(X)[n]$ and strings X that satisfies $\varphi(X)$ and $|X| = n$.

Notation. We use $val(t)$ for the numerical value of a term t, where t may have numerical constants substituted for variables.

We define $\varphi(X)[n]$ inductively as follows. For the base case, $\varphi(X)$ is an atomic formula. Consider the following possibilities.

- If $\varphi(X)$ is $X = X$, then $\varphi(X)[n] =_{\text{def}} \top$.
- If $\varphi(X)$ is $\top$ or $\bot$, then $\varphi(X)[n] =_{\text{def}} \varphi(X)$.
- If $\varphi(X)$ is $t(|X|) = u(|X|)$, then

$$\varphi(X)[n] =_{\text{def}} \begin{cases} \top & \text{if } val(t(\underline{n})) = val(u(\underline{n})) \\ \bot & \text{otherwise} \end{cases}$$

- Similarly if $\varphi(X)$ is $t(|X|) \le (|X|)$.
- If $\varphi(X)$ is $X(t(|X|))$, then we set $j = val(t(\underline{n}))$. Let

$$\varphi(X)[0] =_{\text{def}} \bot$$

and for $n \geq 1$:

$$\varphi(X)[n] =_{\text{def}} \begin{cases} p_j^X & \text{if } j < n-1 \\ \top & \text{if } j = n-1 \\ \bot & \text{if } j > n-1 \end{cases}$$

For the induction step, $\varphi(X)$ is built from smaller formulas using a propositional connective $\wedge, \vee, \neg$, or a bounded number quantifier. For $\wedge, \vee, \neg$ we make the obvious definitions: If both $\psi(X)[n]$ and $\eta(X)[n]$ are not the logical constants $\bot$ or $\top$, then

$$\begin{aligned} \big(\psi(X) \wedge \eta(X)\big)[n] &=_{\text{def}} (\psi(X)[n] \wedge \eta(X)[n]) \\ \big(\psi(X) \vee \eta(X)\big)[n] &=_{\text{def}} (\psi(X)[n] \vee \eta(X)[n]) \\ \big(\neg\psi(X)\big)[n] &=_{\text{def}} \neg\psi(X)[n] \end{aligned}$$

Otherwise, if either $\psi(X)[n]$ or $\eta(X)[n]$ is a logical constant $\bot$ or $\top$, then we simplify the above definitions in the obvious way. For example,

$$\big(\psi(X) \wedge \eta(X)\big)[n] =_{\text{def}} \begin{cases} \eta(X)[n] & \text{if } \psi(X)[n] \text{ is } \top \\ \psi(X)[n] & \text{if } \eta(X)[n] \text{ is } \top \\ \bot & \text{if either } \psi(X)[n] \text{ or } \eta(X)[n] \text{ is } \bot \end{cases}$$

For the case of bounded number quantifiers, $\varphi(X)$ is $\exists y \leq t(|X|)\ \psi(y, X)$ or $\forall y \leq t(|X|)\ \psi(y, X)$. We define

$$\begin{aligned} \big(\exists y \leq t(|X|)\ \psi(y, X)\big)[n] &=_{\text{def}} \bigvee_{i=0}^{m} \psi(\underline{i}, X)[n] \\ \big(\forall y \leq t(|X|)\ \psi(y, X)\big)[n] &=_{\text{def}} \bigwedge_{i=0}^{m} \psi(\underline{i}, X)[n] \end{aligned}$$

where $m = \mathit{val}(t(\underline{n}))$, and recall that $\underline{i}$ is the i-th numeral. Also, if any of the $\psi(\underline{i}, X)[n]$ is translated into $\top$ or $\bot$, we simplify $\varphi(X)[n]$ just as above.

Recall that $s < t$ stands for $s \leq t \wedge s \neq t$. For $\mathit{val}(t(\underline{n})) \geq 1$ we have

$$\begin{aligned} \big(\exists y < t(|X|)\ \psi(y, X)\big)[n] &\leftrightarrow \bigvee_{i=0}^{m-1} \psi(\underline{i}, X)[n] \\ \big(\forall y < t(|X|)\ \psi(y, X)\big)[n] &\leftrightarrow \bigwedge_{i=0}^{m-1} \psi(\underline{i}, X)[n] \end{aligned}$$

In addition,

$$(\exists y < 0\ \psi(y, X))[n] \leftrightarrow \bot, \qquad (\forall y < 0\ \psi(y, X))[n] \leftrightarrow \top$$

Recall that $\langle x, y\rangle$ is the pairing function, and we write $X(x, y)$ for $X(\langle x, y\rangle)$. We formulate the Pigeonhole Principle using a $\Sigma_0^B(\mathcal{L}_A^2)$ formula $\mathbf{PHP}(y, X)$ below. Here y stands for the number of holes, and X

is intended to be a 2-dimensional Boolean array, with $X(i,j)$ holds iff pigeon i gets placed in hole j (for $0 \leq i \leq y$, $0 \leq j < y$).

EXAMPLE 7.18 (Formulation of **PHP** in Two-Sorted Logic).

$$\mathbf{PHP}(y,X) \equiv \forall i \leq y \exists j < y X(i,j) \supset \exists i \leq y \exists k \leq y \exists j < y(i < k \wedge X(i,j) \wedge X(k,j)) \tag{124}$$

Then for all $1 \leq n \in \mathbb{N}$, $\mathbf{PHP}(\underline{n}, X)[1 + \langle n, n-1 \rangle]$ is just $\mathbf{PHP}_n^{n+1}$ (Definition 7.12).

In general, we can define the translation of a $\Sigma_0^B(\mathcal{L}_A^2)$ formula $\varphi(\vec{x}, \vec{X})$ (i.e., with multiple free variables of both sorts). Then for each string variable X_k we associate a list of propositional variables $p_0^{X_k}, p_1^{X_k}, \ldots$, and we give each free number variable a numerical value. Thus the family $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ is defined so that it is valid iff the formula

$$\forall \vec{x} \forall \vec{X},\ (\bigwedge |X_k| = \underline{n_k}) \supset \varphi(\underline{\vec{m}}, \vec{X})$$

is true in the standard model $\underline{\mathbb{N}}_2$. Here for the base case we have to handle an additional case, i.e., where $\varphi(\vec{x}, \vec{X}) \equiv X_i = X_k$, where $i \neq k$. We reduce this case to other cases by considering φ to be its equivalence given by the LHS of the axiom **SE** (Figure 2):

$$|X_i| = |X_k| \wedge \forall x < |X_i|(X_i(x) \leftrightarrow X_k(x))$$

LEMMA 7.19. *For every $\Sigma_0^B(\mathcal{L}_A^2)$ formula $\varphi(\vec{x}, \vec{X})$, there is a constant $d \in \mathbb{N}$ and a polynomial $\mathbf{p}(\vec{m}, \vec{n})$ such that for all $\vec{m}, \vec{n} \in \mathbb{N}$, the propositional formula $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ has depth at most d and size at most $\mathbf{p}(\vec{m}, \vec{n})$.*

PROOF. The proof is by structural induction on φ, and is straightforward. ⊣

Now we come to the main result of this section:

THEOREM 7.20 ($\mathbf{V}^0$ Translation Theorem). *Suppose that $\varphi(\vec{x}, \vec{X})$ is a Σ_0^B formula such that $\mathbf{V}^0 \vdash \forall \vec{x} \forall \vec{X} \varphi(\vec{x}, \vec{X})$. Then the propositional family $\|\varphi(\vec{x}, \vec{X})\|$ has polynomial size bounded depth **PK** proofs. That is, there are a constant d and a polynomial $\mathbf{p}(\vec{m}, \vec{n})$ such that for all $1 \leq \vec{m}, \vec{n} \in \mathbb{N}$, $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ has a d-**PK** proof of size at most $\mathbf{p}(\vec{m}, \vec{n})$. Further there is an algorithm which finds a d-**PK** proof of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ in time bounded by a polynomial in $(\vec{m}, \vec{n})$.*

(See Theorem 7.61 for a generalization of this result which applies to all bounded theorems of $\mathbf{V}^0$.)

In view of the Bounded Depth Lower Bound Theorem 7.16 above, we have:

COROLLARY 7.21 (Independence of PHP from $\mathbf{V}^0$). *The true $\forall \Sigma_0^B$ sentence*

$$\forall y \forall X\ \mathbf{PHP}(y, X)$$

(see Example 7.18) is not a theorem of $\mathbf{V}^0$.

To prove the $\mathbf{V}^0$ Translation Theorem, the idea is to translate each sequent in an $\mathbf{LK}^2$ proof of $\varphi(\vec{a}, \vec{\alpha})$ into a **bPK** sequent which has a short proof. The issue here is that an $\mathbf{LK}^2$-$\mathbf{V}^0$ proof may contain Σ_1^B formulas (i.e., the Σ_0^B-**COMP** axioms), whose translation we have not discussed. We introduce the theory $\widetilde{\mathbf{V}}^0$ which plays the same role for $\mathbf{V}^0$ as $\tilde{\mathbf{V}}^1$ does for $\mathbf{V}^1$. In the next subsection we define $\widetilde{\mathbf{V}}^0$ and the associated sequent system $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ (an analogue of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$), and use these to prove the $\mathbf{V}^0$ Translation Theorem.

7B.2. $\widetilde{\mathbf{V}}^0$ and $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$.

DEFINITION 7.22. *The theory* $\widetilde{\mathbf{V}}^0$ *has vocabulary* $\mathcal{L}_A^2$ *and is axiomatized by* 2-**BASIC** *and the* Σ_0^B-**IND** *axiom scheme.*

Thus $\widetilde{\mathbf{V}}^0$ is the same as $\mathbf{V}^0$, except the Σ_0^B-**COMP** axioms are replaced by the Σ_0^B-**IND** axioms. By Corollary 5.8, $\mathbf{V}^0$ proves the Σ_0^B-**IND** axiom scheme, hence $\widetilde{\mathbf{V}}^0 \subseteq \mathbf{V}^0$.

Unlike the $\tilde{\mathbf{V}}^1$, $\mathbf{V}^1$ case, unfortunately $\mathbf{V}^0$ is not the same as $\widetilde{\mathbf{V}}^0$, because $\widetilde{\mathbf{V}}^0$ does not prove the Σ_0^B-**COMP** axioms. To see this, expand the standard (single-sorted) model $\underline{\mathbb{N}}$ to a $\mathcal{L}_A^2$ structure $\mathcal{M}$ by letting the string universe be $\{\varnothing\}$, where $|\varnothing| = 0$. Then it is easy to see that $\mathcal{M}$ is a model of $\widetilde{\mathbf{V}}^0$, but not of $\mathbf{V}^0$. Nevertheless, we can prove a weaker statement.

DEFINITION 7.23 (Φ-Conservative Extension). *Let* Φ *be a set of formulas in the vocabulary* $\mathcal{L}$. *Suppose that* $\mathcal{T}$ *is a theory over* $\mathcal{L}$, *and* $\mathcal{T}'$ *is an extension of* $\mathcal{T}$ *(the vocabulary of* $\mathcal{T}'$ *may contain function or predicate symbols not in* $\mathcal{L}$*). Then we say that* $\mathcal{T}'$ *is a* Φ-conservative extension *of* $\mathcal{T}$ *if for every formula* $\varphi \in \Phi$, *if* $\mathcal{T}' \vdash \varphi$ *then* $\mathcal{T} \vdash \varphi$.

So if Φ is the set of all $\mathcal{L}$ formulas, then $\mathcal{T}'$ is Φ-conservative over $\mathcal{T}$ precisely when it is conservative over $\mathcal{T}$. For the case of $\widetilde{\mathbf{V}}^0$ and $\mathbf{V}^0$, we can take Φ to be Σ_0^B.

LEMMA 7.24. $\mathbf{V}^0$ *is* Σ_0^B*-conservative over of* $\widetilde{\mathbf{V}}^0$.

By our definition of semantics (Sections 4B.2 and 2B.2), this is the same as saying that $\mathbf{V}^0$ is $\forall\Sigma_0^B$-conservative over $\widetilde{\mathbf{V}}^0$, where $\forall\Sigma_0^B$ is the universal closure of Σ_0^B (Definition 2.41).

PROOF. We noted earlier that $\widetilde{\mathbf{V}}^0 \subseteq \mathbf{V}^0$ (by Corollary 5.8). The proof that every Σ_0^B theorem of $\mathbf{V}^0$ is also provable in $\widetilde{\mathbf{V}}^0$ is like the proof that $\mathbf{V}^0$ is conservative over $\mathbf{I\Delta}_0$ (Theorem 5.9). We use the following lemma, which is proved in the same way as Lemma 5.10 (any model of $\mathbf{I\Delta}_0$ can be expanded to a model of $\mathbf{V}^0$). In the present case, U_2' is defined as before in (53), except that now the formula φ is allowed parameters from U_2.

LEMMA 7.25. *Every model $\mathcal{M} = \langle U_1, U_2 \rangle$ for $\widetilde{\mathbf{V}}^0$ can be extended to a model $\mathcal{M}' = \langle U_1', U_2' \rangle$ of $\mathbf{V}^0$, where $U_1 = U_1'$ and $U_2 \subseteq U_2'$.*

It follows that if $\varphi(\vec{x}, \vec{X})$ is a Σ_0^B formula with all free variables indicated, and $\vec{a}$ are any elements in U_1 and $\vec{\alpha}$ are any elements in U_2, then

$$\mathcal{M} \models \varphi(\vec{a}, \vec{\alpha}) \qquad \text{iff} \qquad \mathcal{M}' \models \varphi(\vec{a}, \vec{\alpha})$$

(The proof actually shows that $\mathbf{V}^0$ is Φ-conservative over $\widetilde{\mathbf{V}}^0$ for a set Φ larger than Σ_0^B, i.e., Φ contains formulas with unbounded number quantifiers and without string quantifiers. But we do not need this fact here.) ⊣

The sequent system $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ is analogous to $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$:

DEFINITION 7.26 ($\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$). *The rules of $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ consist of the rules of $\mathbf{LK}^2$ (Section 4D), together with the Σ_0^B-**IND** rule (Definition 6.38). The non-logical axioms of $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ are sequents of the form $\longrightarrow A$, where A is any term substitution instance of a 2-**BASIC** axiom (Figure 2) or an $\mathbf{LK}^2$ equality axiom (Definition 4.26).*

Recall the notion of an anchored $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ proof from Definition 6.40, and the Anchored Completeness Lemma for $\mathbf{LK}^2$+**IND** 6.42. We are now ready to prove the $\mathbf{V}^0$ Translation Theorem.

7B.3. Proof of the Translation Theorem for $\mathbf{V}^0$. By assumption, $\varphi(\vec{a}, \vec{\alpha})$ is a Σ_0^B theorem of $\mathbf{V}^0$. By the Anchored Completeness Lemma for $\mathbf{LK}^2$+**IND** 6.42, there is an anchored $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ proof π of $\varphi(\vec{a}, \vec{\alpha})$. We may assume that π is in free variable normal form, where (as in Subsection 6D.2) we modify Definition 2.39 to allow the rule Σ_0^B-**IND** to eliminate a variable. By the Subformula Property of $\mathbf{LK}^2$+**IND** (Proposition 6.45), every formula in every sequent of π is Σ_0^B. So every sequent $\mathcal{S}$ in π has the form

$$\psi_1(\vec{b}, \vec{\beta}), \ldots, \psi_k(\vec{b}, \vec{\beta}) \longrightarrow \eta_1(\vec{b}, \vec{\beta}), \ldots, \eta_\ell(\vec{b}, \vec{\beta})$$

where ψ_i, η_j are Σ_0^B formulas, and $(\vec{b}, \vec{\beta})$ are all the free variables in $\mathcal{S}$ (which may be different for different sequents). The translation $\mathcal{S}[\vec{m}; \vec{n}]$ is obtained from the translations $\psi_i(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}]$ and $\eta_j(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}]$ as follows. First, if any $\psi_i(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}]$ is $\bot$, or any $\eta_j(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}]$ is $\top$, then $\mathcal{S}[\vec{m}; \vec{n}]$ is the axiom

$$\longrightarrow \top \tag{125}$$

Otherwise, $\mathcal{S}[\vec{m}; \vec{n}]$ has the form

$$\mathcal{S}[\vec{m}; \vec{n}] =_{\text{def}} \qquad \ldots, \psi_i(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}], \ldots \longrightarrow \ldots, \eta_j(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}], \ldots$$

where the antecedent consists of all $\psi_i(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}]$ that are not $\top$, and the succedent consists of all $\eta_j(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}]$ that are not $\bot$.

We will prove by induction on the number of lines above this sequent in π that there are a constant d and a polynomial $\mathbf{p}$ depending on π, such that the propositional sequent $\mathcal{S}[\vec{m};\vec{n}]$ has a d-**PK** proof of size at most $\mathbf{p}(\vec{m},\vec{n})$, for all $\vec{m},\vec{n} \in \mathbb{N}$. It is straightforward to verify that the proof can be obtained in time polynomial in $\vec{m},\vec{n}$.

For the base case, $\mathcal{S}$ is a non-logical axiom of $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$. Thus $\mathcal{S}$ is of the form $\longrightarrow \eta$, where η is a term substitution instance of the 2-**BASIC** axioms, or $\mathcal{S}$ is an instance of the Equality axioms (Definition 4.26). First, any string variable X can occur in an instance of **B**1–**B**12 only in the context of a number term $|X|$. Since these axioms are true in the standard model $\underline{\mathbb{N}}_2$, they translate into the propositional constant $\top$. Therefore if η is an instance of **B**1–**B**12, then $\longrightarrow \eta$ translates into the axiom (125) of **PK**.

Instances of **L**1 and **L**2 translate into (125). Consider, for example, an instance of **L**1:

$$\eta(\vec{b},\gamma,\vec{\beta}) \equiv \gamma(t) \supset t < |\gamma|$$

where $\vec{b},\vec{\beta}$ denote all (free) variables occurring in the $\mathcal{L}^2_A$-number term $t = t(\vec{b},|\gamma|,|\vec{\beta}|)$. By definition, in order to get $\eta(\vec{b},\gamma,\vec{\beta})[\vec{m};n,\vec{n}]$, first we obtain the formulas

$$\begin{cases} p^\gamma_i \supset \top & \text{if } i < n-1 \\ \top \supset \top & \text{if } i = n-1 \\ \bot \supset \bot & \text{if } i > n-1 \end{cases}$$

where $i = \mathit{val}(t(\vec{m},n,\vec{n}))$. Simplifying these formulas results in

$$\eta(\vec{b},\gamma,\vec{\beta})[\vec{m};n,\vec{n}] =_{\text{def}} \top$$

By definition, any instance of the axiom **SE** translates into a formula of the form $A \supset A$, where A is the translation of the LHS of **SE**. This tautology has a short cut-free derivation **PK**.

Similar (and simple) arguments show that if $\mathcal{S}$ is an instance of any of the Equality Axioms, then its $\mathcal{S}[\vec{m};n,\vec{n}]$ has a short d-**PK** proof, for some small constant d. (This constant accounts for the fact that we translate $X = Y$ using the LHS of **SE**, which translates into a propositional formula of depth 3.)

For the induction step, we consider the rules of $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$. Since all formulas in π are Σ^B_0, the string quantifier rules are never applied. If $\mathcal{S}$ is obtained from $\mathcal{S}_1$ (and $\mathcal{S}_2$) by one of the introduction rules for the connectives $\wedge$, $\vee$ and $\neg$ and the translation(s) of the auxiliary formula(s) are not simplified to Boolean constants then we can apply the same rules to get the **PK** proof of $\mathcal{S}[\vec{m};\vec{n}]$ from the **PK** proof(s) of $\mathcal{S}_1[\vec{m};\vec{n}]$ (and $\mathcal{S}_2[\vec{m};\vec{n}]$). Otherwise, if an auxiliary formula is translated into $\top$ or $\bot$ then it can be seen that $\mathcal{S}[\vec{m};\vec{n}]$ is the same as $\mathcal{S}_1[\vec{m};\vec{n}]$ (or $\mathcal{S}_2[\vec{m};\vec{n}]$). No new cut is needed for this step.

For the case of the cut rule, the cut formula $\psi(\vec{b},\vec{\beta})$ is Σ^B_0, and since π is in free variable normal form, no variable is eliminated by the rule. Consider the interesting case where the translation of $\psi(\vec{b},\vec{\beta})$ is not a constant $\top$ or $\bot$. The corresponding **PK** proof also uses the cut rule, where the cut formula is a propositional translation $\psi(\vec{b},\vec{\beta})[\vec{m};\vec{n}]$ of this formula, which according Lemma 7.19 has bounded depth d independent of $\vec{m},\vec{n}$.

Consider the case of the number $\forall$-right. Suppose that the inference is

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda \longrightarrow \Pi, c \le t(\vec{b},|\vec{\beta}|) \supset \eta(\vec{b},c,\vec{\beta})}{\Lambda \longrightarrow \Pi, \forall x \le t(\vec{b},|\vec{\beta}|)\ \eta(\vec{b},x,\vec{\beta})}$$

where c does not occur in $\mathcal{S}$. By the induction hypothesis, there are a constant $d \in \mathbb{N}$ and a polynomial $\mathbf{p}(\vec{m},i,\vec{n})$ so that for each $\langle \vec{m},i,\vec{n}\rangle$, there is a d-**PK** proof $\pi[\vec{m},i;\vec{n}]$ of size $\le \mathbf{p}(\vec{m},i,\vec{n})$ of the sequent $\mathcal{S}_1[\vec{m},i;\vec{n}]$. Note that if for some $i \le r$, $\eta(\vec{b},c,\vec{\beta})[\vec{m},i;\vec{n}]$ is $\bot$ then $\forall x \le t(\vec{b},|\vec{\beta}|)\ \eta(\vec{b},x,\vec{\beta})$ translates into $\bot$ and hence $\mathcal{S}[\vec{m};\vec{n}] = \mathcal{S}_1[\vec{m},i;\vec{n}]$ and we are done. Moreover, if all $\eta(\vec{b},c,\vec{\beta})[\vec{m},i;\vec{n}]$ (for $i \le r$) are $\top$ then $\mathcal{S}[\vec{m};\vec{n}]$ is the axiom $\longrightarrow \top$ and we are also done.

Now, if some $\eta(\vec{b},c,\vec{\beta})[\vec{m},i;\vec{n}]$ is $\top$ then it will be deleted from the translation of $\forall x \le t(\vec{b},|\vec{\beta}|)\ \eta(\vec{b},x,\vec{\beta})$, and the sequent $\mathcal{S}_1[\vec{m},i;\vec{n}]$ is the axiom $\longrightarrow \top$ and it will not be used in the following derivation. So suppose that for all $i \le r$, $\eta(\vec{b},c,\vec{\beta})[\vec{m},i;\vec{n}]$ is neither $\top$ nor $\bot$. Then for $i \le r$, $\mathcal{S}_1[\vec{m},i;\vec{n}]$ is

$$\Lambda[\vec{m};\vec{n}] \longrightarrow \Pi[\vec{m};\vec{n}],\ \eta(\vec{b},c,\vec{\beta})[\vec{m},i;\vec{n}]$$

The sequent $\mathcal{S}$ translates into

$$\mathcal{S}[\vec{m};\vec{n}] =_{\mathrm{def}} \Lambda[\vec{m};\vec{n}] \longrightarrow \Pi[\vec{m};\vec{n}],\ \bigwedge_{i=0}^{r} \eta(\vec{b},\underline{i},\vec{\beta})[\vec{m};\vec{n}]$$

Thus $\mathcal{S}[\vec{m};\vec{n}]$ is obtained from $\mathcal{S}_1[\vec{m},i;\vec{n}]$ (for $i = 0,1,\ldots,r$) by the $\wedge$-right rule. No new instance of the cut rule is needed. This proof of $\mathcal{S}[\vec{m};\vec{n}]$ has size slightly more than the sum of the $(m+1)$ proofs $\pi[\vec{m},i;\vec{n}]$, and m is a polynomial in $\vec{m},\vec{n}$. Hence the resulting proof is bounded in size by a polynomial in $\vec{m},\vec{n}$.

The case $\exists$-left is similar, and the cases $\forall$-left, $\exists$-right are straightforward. These are left as an exercise.

EXERCISE 7.27. *Take care of the other number quantifier cases.*

Finally we consider the case that $\mathcal{S}$ is obtained by the Σ_0^B-**IND** rule:

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda, \psi(c) \longrightarrow \psi(c+1), \Pi}{\Lambda, \psi(0) \longrightarrow \psi(t), \Pi}$$

where c does not occur in $\mathcal{S}$, and we have suppressed all free variables except c (here t is of the form $t(\vec{b},|\vec{\beta}|)$). By the induction hypothesis, there are polynomial size d-**PK** proofs $\pi[\vec{m},i;\vec{n}]$ of the propositional sequents

$$\mathcal{S}_1[\vec{m},i;\vec{n}] =_{\mathrm{def}} \Lambda[\vec{m};\vec{n}], \psi(c)[\vec{m},i;\vec{n}] \longrightarrow \psi(c+1)[\vec{m},i;\vec{n}], \Pi[\vec{m};\vec{n}]$$

for some constant $d \in \mathbb{N}$. Let $r = val(t(\vec{m}, \vec{n}))$. The sequent $\mathcal{S}$ translates into

$$\mathcal{S}[\vec{m};\vec{n}] =_{\text{def}} \Lambda[\vec{m};\vec{n}], \psi(0)[\vec{m};\vec{n}] \longrightarrow \psi(r)[\vec{m};\vec{n}], \Pi[\vec{m};\vec{n}]$$

Now if $r = 0$ then $\mathcal{S}[\vec{m};\vec{n}]$ is derived from the following axiom of **PK** simply by weakening:

$$\psi(0)[\vec{m};\vec{n}] \longrightarrow \psi(0)[\vec{m};\vec{n}]$$

For $r > 0$, we combine these proofs $\pi[\vec{m}, i; \vec{n}]$ for $i = 0, 1, \ldots, r-1$ by using repeated cuts, with cut formulas $\psi(i)[\vec{m};\vec{n}]$, $1 \leq i \leq r-1$. By Lemma 7.19, these formulas have depth bounded by a constant depending only on ψ. Also, given that each $\pi[\vec{m}, i; \vec{n}]$ has a polynomial bounded size, the proof $\pi[\vec{m};\vec{n}]$ is easily shown to be bounded in size by some polynomial in $\vec{m}, \vec{n}$. This completes the proof of the Translation Theorem for $\mathbf{V}^0$. ⊣

Note that the Σ_0^B-**IND** axioms are Σ_0^B. So in fact we could have defined $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ to include the Σ_0^B-**IND** *axiom scheme* instead of the Σ_0^B-**IND** *rule*. Here we can use the following version of the Σ_0^B-**IND** axiom:

$$(126) \qquad (\varphi(0) \wedge \forall x < t(\varphi(x) \supset \varphi(x+1))) \supset \forall z \leq t\varphi(z)$$

where t is any term not involving x or z, and φ is a Σ_0^B formula which may contain other free variables.

In this way, the case of the Σ_0^B-**IND** rule in the induction step of the proof above is replaced by two cases: One for the base case where the axiom is an Σ_0^B-**IND** axiom, and one for the induction step, in the case of the cut rule where the cut formula is an instance of the Σ_0^B-**IND** axioms. The latter is dealt with just as any other instance of the cut rule. Handling the former is left as an exercise.

EXERCISE 7.28. *Show directly (without using Theorem 7.20) that the translation of* (126) *above has polynomial size d-**PK** proofs, where d depends only on φ.*

7C. Quantified Propositional Calculus

Quantified Propositional Calculus (QPC) is an extension of the Propositional Calculus (Section 2A) which allows quantifiers over propositional variables. In this section we will discuss the sequent system **G** which extends Gentzen's system **PK** by the introduction rules for the propositional quantifiers. There are subsystems of **G** that relate to the first-order theories in the same way that **bPK** relates to $\mathbf{V}^0$. Here we will show this relationship between $\mathbf{V}^1$ and the subsystem $\mathbf{G}_1^\star$ of **G**.

Formally, QPC formulas (or simply formulas) are built from

- propositional constants $\top, \bot$;
- free variables $p, q, r, \ldots$;

- bound variables $x, y, z, \ldots$;
- connectives $\wedge, \vee, \neg$;
- quantifiers $\exists, \forall$;
- parentheses (,);

according to the following rules:

(a) $\top$, $\bot$, and p are *atomic* formulas, for any free variable p;
(b) if φ and ψ are formulas, so are $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $\neg\varphi$;
(c) if $\varphi(p)$ is a formula, then $\forall x\varphi(x)$ and $\exists x\varphi(x)$ are formulas, for any free variable p and bound variable x.

A QPC sentence (or just sentence) is a QPC formula with no occurrence of a free variable.

EXAMPLE 7.29. *The following is a QPC formula:*

$$\forall x\exists y\big((\neg y \vee (\neg x \wedge p)) \wedge (y \vee x \vee \neg p)\big) \tag{127}$$

A truth assignment is an assignment of truth values True, False to the free variables. The truth value of a QPC formula is defined inductively, much as in the case of the Propositional Calculus. Here in the induction step, for the case of the quantifiers we use the equivalences

$$\forall x\varphi(x) \leftrightarrow (\varphi(\bot) \wedge \varphi(\top)) \qquad \text{and} \qquad \exists x\varphi(x) \leftrightarrow (\varphi(\bot) \vee \varphi(\top))$$

A QPC formula is *valid* if it is true under all assignments. The notions of *satisfiability* and *logical consequence* (Definition 2.1) generalize to QPC in the obvious way. So, for example, the formula (127) is valid (choose $y \leftrightarrow (\neg x \wedge p)$).

It is a standard result in complexity theory that the problem of determining validity of a formula of QPC is **PSPACE** complete (see Appendix A). Furthermore, it is natural to define a language $L \subseteq \{0,1\}^*$ to be in nonuniform **PSPACE** if there is a polynomial size family $\langle \varphi_n(\vec{p})\rangle$ of QPC formulas such that $\varphi_n(p_1, ..., p_n)$ defines the strings of length n in L. (Actually this defines the class **PSPACE**/*poly*, which is PSPACE with polynomial advice.) For this and other reasons, **G** (defined below) is a natural choice for a QPC proof system corresponding to the complexity class **PSPACE**. However if the number of quantifier alternations in a QPC formula is limited by some constant k, then the validity problem for such formulas is in the polynomial hierarchy.

DEFINITION 7.30 (Σ_i^q and Π_i^q). $\Sigma_0^q = \Pi_0^q$ *is the class of quantifier-free formulas of QPC. For* $i \geq 0$, Σ_{i+1}^q *and* Π_{i+1}^q *are the smallest classes of QPC formulas satisfying*

1) $\Sigma_i^q \cup \Pi_i^q \subseteq \Sigma_{i+1}^q \cap \Pi_{i+1}^q$
2) Σ_{i+1}^q *is closed under* $\vee$ *and* $\wedge$ *and existential quantification*
3) Π_{i+1}^q *is closed under* $\vee$ *and* $\wedge$ *and universal quantification*
4) *if* $A \in \Sigma_{i+1}^q$ *then* $\neg A \in \Pi_{i+1}^q$
5) *if* $A \in \Pi_{i+1}^q$ *then* $\neg A \in \Sigma_{i+1}^q$

Thus

$$\Sigma_0^q = \Pi_0^q \subset \cdots \subset \Sigma_i^q \cap \Pi_i^q \subset \Sigma_i^q \cup \Pi_i^q \subset \Sigma_{i+1}^q \cap \Pi_{i+1}^q \subset \ldots$$

For $i \geq 0$ every formula in Σ_{i+1}^q has a prenex form with at most i alternations of quantifiers, with the outermost quantifier being $\exists$. Similarly for Π_{i+1}^q with the outermost quantifier being $\forall$. Checking the validity of a Σ_i^q (resp. Π_i^q) sentence is Σ_i^p-complete (resp. Π_i^p-complete), for $i \geq 1$. For $i = 0$, this problem is $\mathbf{NC}^1$-complete.

7C.1. QPC Proof Systems. We generalize Definition 7.2 in the obvious way to define the notion of *QPC proof system* where now F maps $\{0,1\}^*$ onto the set of valid QPC formulas. Since the validity problem for QPC formulas is complete for **PSPACE**, the following result is proved in the same way as Theorem 7.4.

Theorem 7.31. *There exists a polynomially bounded QPC proof system iff* $\mathbf{NP} = \mathbf{PSPACE}$.

The assertion $\mathbf{NP} = \mathbf{PSPACE}$ is considerably more implausible than $\mathbf{NP} = co\text{-}\mathbf{NP}$, but still the existence of a polynomially bounded QPC proof system is open.

The notions *p-simulate* and *p-equivalent* from Definition 7.5 apply in the obvious way to QPC proof systems.

7C.2. The System G. The QPC proof system **G** is a sequent system which includes the axioms and rules for **PK**, where now formulas are interpreted to be QPC formulas. It also has the following four quantifier introduction rules:

$\forall$ introduction rules:

$$\forall\text{-left: } \frac{A(B), \Gamma \longrightarrow \Delta}{\forall x A(x), \Gamma \longrightarrow \Delta} \qquad \forall\text{-right: } \frac{\Gamma \longrightarrow \Delta, A(p)}{\Gamma \longrightarrow \Delta, \forall x A(x)}$$

$\exists$ introduction rules:

$$\exists\text{-left: } \frac{A(p), \Gamma \longrightarrow \Delta}{\exists x A(x), \Gamma \longrightarrow \Delta} \qquad \exists\text{-right: } \frac{\Gamma \longrightarrow \Delta, A(B)}{\Gamma \longrightarrow \Delta, \exists x A(x)}$$

Restriction. In the rules $\forall$-right and $\exists$-left, p is a free variable called an *eigenvariable* that must not occur in the bottom sequent. For the rules $\forall$-left and $\exists$-right, the formula B is called the *target* formula and may be any quantifier-free formula (with no bound variables).

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

Proofs in **G** are dags of sequents, which generalizes the treelike structure of **LK** proofs (see Subsection 7A.1). The notion of free variable normal form (Definition 2.39) readily extends to **G** proofs. In fact every treelike **G** proof can be easily transformed to one in free variable normal form by renaming variables and substituting the constant $\bot$ for some variables.

THEOREM 7.32 (Soundness and Completeness of **G**). *A sequent of* **G** *is valid iff it has a* **G** *proof. In fact, valid sequents have cut-free* **G** *proofs.*

PROOF. Soundness is easy: Provable sequents of **G** are valid because the axioms of **G** are valid, and the rules preserve validity.

For completeness, we first point out that a valid quantifier-free sequent of QPC has a cut-free **G** proof, by the **PK** Completeness Theorem 2.8. In general, we prove the result by induction on the maximum quantifier depth of the formulas in the sequent (and then induction on the number of formulas in the sequent of maximum quantifier depth). We have just proved the base case, where the sequent is quantifier-free. For the induction step, the interesting cases are where the sequent is of the form

$$\forall x A(x), \Gamma \longrightarrow \Delta \qquad \text{or} \qquad \Gamma \longrightarrow \Delta, \exists x A(x)$$

These two cases are dual. So consider the sequent

$$\forall x A(x), \Gamma \longrightarrow \Delta \tag{128}$$

We can reduce the quantifier depth in $\forall x A(x)$ by showing that (128) is valid iff the sequent

$$A(\top), A(\bot), \Gamma \longrightarrow \Delta \tag{129}$$

is valid. ⊣

EXERCISE 7.33. *Carry out the details in the induction step in the above proof of the completeness of* **G**.

The proof above shows that actually **G** remains complete when the target formulas B in $\forall$-left and $\exists$-right are restricted to be in the set $\{\top, \bot\}$. In fact, the restricted system is p-equivalent to **G**. This can be shown with the help of the following exercise.

EXERCISE 7.34. *Show that the following sequents has cut-free* **G** *proofs of size* $\mathcal{O}(|A(B)|^2)$, *where* A *and* B *are any QPC formulas.*

(a) $B, A(B) \longrightarrow A(\top)$
(b) $A(B) \longrightarrow A(\bot), B$
(c) $B, A(\top) \longrightarrow A(B)$
(d) $A(\bot) \longrightarrow A(B), B$

(Hint: Prove by structural induction on A *for (a) and (c) simultaneously. Similarly for (b) and (d).)*

EXERCISE 7.35 (Morioka [61]). *Let* **KPG** *be the modification of* **G** *resulting from relaxing the condition that the target formula* B *in the rules* $\forall$*-left and* $\exists$*-right must be quantifier-free (so* B *is allowed to be any QPC formula). Show that* **G** *p-simulates* **KPG**. *Show that the same holds even if* **G** *is restricted so that the target formulas* B *in the rules* $\forall$*-left and* $\exists$*-right are restricted to be in the set* $\{\top, \bot\}$. *Use Exercise 7.34.*

The original system $\mathbf{G}$ defined in [56] is actually $\mathbf{KPG}$ as defined in the above exercise. Thus the original $\mathbf{G}$ and our $\mathbf{G}$ are p-equivalent.

The proof of completeness in Theorem 7.32 could yield proofs of doubly exponential size. For example if the formula $\forall x A(x)$ in (128) begins with k universal quantifiers, then eliminating them all using (129) would yield 2^k copies of A, and the resulting valid sequent could require a proof exponential in its length. We now prove a singly-exponential upper bound for G proofs which allow cuts on atomic formulas.

We say that an occurrence of a symbol in a formula is *positive* (resp. *negative*) if it is in the scope of an even (resp. odd) number of $\neg$'s.

Definition 7.36 (Sequent Length). *An occurrence of a connective c in a sequent $\Gamma \longrightarrow \Delta$ is* general *if c is $\wedge$ or $\forall$ and occurs positively in Δ or negatively in Γ, or if c is $\vee$ or $\exists$ and c occurs negatively in Δ or positively in Γ. A* restricted *occurrence is defined similarly, except Δ and Γ are interchanged. For a sequent S, $|S|_g$ (resp. $|S|_r$) denotes the number of occurrences in S of general connectives (resp. $\neg$'s and restricted connectives). Also $|S|$ denotes the total number of occurrences of symbols in S, counting variables $p, q, r, \ldots, x, y, z, \ldots$ as one symbol each.*

Theorem 7.37. *If S is a valid sequent in the language of $\mathbf{G}$ with n distinct free variables, then S has a treelike $\mathbf{G}$ proof with $O(|S|_r 2^{|S|_g + n})$ sequents (not counting weakenings and exchanges) in which all cut formulas are atomic and each sequent in the proof has length $O(|S|)$. If S is quantifier-free, or if all quantifier occurrences in S are general, then the proof is cut-free and the bound is improved to $O(|S|_r 2^{|S|_g})$.*

Proof. **Notation**. We say that a free variable p is *determined* in a sequent $A_1, \ldots, A_k \longrightarrow B_1, \ldots B_\ell$ if one of the formulas A_i or B_j is the atomic formula p. A sequent is *determined* if all of its free variables are determined.

Note that if all free variables of a sequent are determined, then there is at most one truth assignment to these free variables which fails to satisfy the sequent.

Lemma 7.38. *If S is a valid sequent with all of its free variables determined, then S has a treelike $\mathbf{G}$ proof with $O(|S|_r 2^{|S|_g})$ sequents (not counting weakenings and exchanges) in which all cut formulas are atomic and each sequent in the proof has length $O(|S|)$. If S is quantifier-free or if all quantifier occurrences in S are general, then the same bound applies even if not all free variables in S are determined, and further the proof is treelike and cut-free.*

The second sentence of Theorem 7.37 follows immediately from the lemma. We now prove the first sentence of the theorem from the lemma. Let F be the set of free variables in S. For each of the 2^n subsets K of F let S_K be the sequent resulting from S by appending a list of the variables in K to the antecedent and the variables in $F - K$ to the consequent. For example if $S = \Gamma \longrightarrow \Delta$ and $F = \{p_1, p_2, p_3\}$ and $K = \{p_2\}$, then S_K is

$$p_2, \Gamma \longrightarrow \Delta, p_1, p_3$$

Each S_K is valid and determined, and hence by the lemma has a proof with $O(|S|_r 2^{|S|_g})$ sequents. Then S can be derived by combining these 2^n proofs with 2^{n-1} atomic cuts. ⊣

Proof of Lemma 7.38. We use induction on the total number of connectives $\wedge, \vee, \neg, \forall, \exists$ in S. The base case is immediate, since any valid sequent with no such connectives is a subsequent of an axiom.

For the induction step, we have a case for each of the connectives $\wedge, \vee, \neg, \forall, \exists$. We consider a formula A occurring in the consequent: The argument for the antecedent is dual. If A is of the form $\neg B$ then S has the form $\Gamma \longrightarrow \Delta, \neg B$. Let S' be the sequent $B, \Gamma \longrightarrow \Delta$. Then S' is valid (and determined if S is) and $|S'|_r = |S|_r - 1$, so the induction hypothesis applies and S can be derived from S' by the rule $\neg$-right. The case in which A has the form $B \vee C$ is similar, using the rule $\vee$-right.

If S has the form $\Gamma \longrightarrow \Delta, (B \wedge C)$, then $\Gamma \longrightarrow \Delta, B$ and $\Gamma \longrightarrow \Delta, C$ are each valid (and determined if S is) and have reduced $|S|_g$, and S can be derived by $\wedge$-right from these two sequents.

Suppose that S is $\Gamma \longrightarrow \Delta, \forall x A(x)$. Then $S' = \Gamma \longrightarrow \Delta, A(p)$ is valid, where p is a new free variable. Further $|S'|_g = |S|_g - 1$ and S follows from S' using $\forall$-right. This takes care of the second sentence in the lemma, but for the first sentence there is the problem that S' may not be determined, even if S is. But each of the sequents $p, \Gamma \longrightarrow \Delta, A(p)$ and $\Gamma \longrightarrow \Delta, A(p), p$ is valid and determined if S is, and by the induction hypothesis can be proved with $O(|S|_r 2^{|S|_g - 1})$ sequents. Further S can be derived from these two sequents with a cut on p and $\forall$-right, making a total of $O(|S|_r 2^{|S|_g} + 2) = O(|S|_r 2^{|S|_g})$ sequents.

Finally consider the case in which S is $\Gamma \longrightarrow \Delta, \exists x A(x)$. Since the occurrence of $\exists$ is restricted, the second sentence of the lemma does not apply, so we may assume that S is determined and valid. We claim that one of the two sequents $\Gamma \longrightarrow \Delta, A(\top)$ and $\Gamma \longrightarrow \Delta, A(\bot)$ is valid (they are both determined). To see this, note that since S is determined there is at most one truth assignment τ to the free variables of S that could falsify $\Gamma \longrightarrow \Delta$. If no such τ exists, we are done. Otherwise τ satisfies $\exists x A(x)$, and hence τ satisfies either $A(\top)$ or $A(\bot)$. Hence we may apply the induction hypothesis to one of these sequents, and obtain S using $\exists$-right. ⊣

7D. The Systems $\mathbf{G}_i$ and $\mathbf{G}_i^\star$

Definition 7.39 ($\mathbf{G}_i$ and $\mathbf{G}_i^\star$). *For each $i \geq 0$, $\mathbf{G}_i$ is the subsystem of $\mathbf{G}$ in which cut formulas are restricted to $\Sigma_i^q \cup \Pi_i^q$. The system $\mathbf{G}_i^\star$ is treelike $\mathbf{G}_i$.*

The following result is immediate from Theorem 7.37.

Corollary 7.40. *Every valid QPC sequent has a $\mathbf{G}_0^\star$ proof of size $2^{O(|S|)}$.*

Theorem 7.41. *For $i \geq 0$, $\mathbf{G}_{i+1}^\star$* p*-simulates $\mathbf{G}_i$, when the systems are restricted to proving $\Sigma_i^q \cup \Pi_i^q$ formulas. Treelike $\mathbf{G}$* p*-simulates $\mathbf{G}$.*

Proof. The argument is similar to the proof of Theorem 7.8, except for the quantifier rules $\forall$-right and $\exists$-left we can no longer argue that the conclusion is a logical consequence of the hypotheses. However for each rule deriving a sequent S from a sequent S_1 we know that $\forall A_S$ is a logical consequence of $\forall A_{S_1}$, where $\forall B$ is the universal closure of B. Thus we replace the Claim in the earlier proof by arguing that if $\pi = S_1, \ldots, S_n$ is a daglike $\mathbf{G}$ proof then

(130)
$$\longrightarrow \forall A_{S_1}; \quad \longrightarrow (\forall A_{S_1} \wedge \forall A_{S_2}); \ldots; \quad \longrightarrow (\forall A_{S_1} \wedge \cdots \wedge \forall A_{S_n}); \quad \longrightarrow A_{S_n}$$

can be augmented to a treelike $\mathbf{G}$ proof whose size is bounded by a polynomial in the length of π, and in which cut formulas are restricted to subformulas of formulas in the sequence. The theorem then follows from the fact if the all formulas in the sequent S are in $\Sigma_i^q \cup \Pi_i^q$ then the formula $\forall A_S$ is in Π_{i+1}^q.

Our new claim follows from Exercise 7.9 (b), the fact that for every axiom S of $\mathbf{G}$, $\longrightarrow \forall A_S$ has an easy $\mathbf{G}_0^\star$ proof, and the exercise below. ⊣

Exercise 7.42. (a) *Suppose that if S is derived from S_1 (and S_2) by an inference rule of $\mathbf{G}$. Show that the following sequents have polynomial size cut-free $\mathbf{G}$ proofs for any formula A. (For the* **PK** *rules it is helpful to use Exercise 7.9 (b).)*
(i) $A \wedge \forall A_{S_1} \longrightarrow A \wedge \forall A_S$.
(ii) $A \wedge \forall A_{S_1} \wedge \forall A_{S_2} \longrightarrow A \wedge \forall A_S$.
(b) *Show that for every sequent $S = \Gamma \longrightarrow \Delta$, the sequent*

$$\forall A_S, \Gamma \longrightarrow \Delta$$

has a polynomial size cut-free treelike $\mathbf{G}$ proof.

The next result strengthens Theorem 7.41 for the case $i = 0$.

Theorem 7.43 (Morioka [61]). *$\mathbf{G}_0^\star$* p*-simulates $\mathbf{G}_0$ restricted to proving prenex Σ_1^q formulas.*

PROOF SKETCH. Note that the proof of Theorem 7.8 (treelike **PK** p-simulates daglike **PK**) does not adapt to this case, because that argument requires cuts on conjunctions of earlier lines in the proof, which now would involve quantifiers.

Instead, following [61], we argue that a form of Gentzen's Midsequent Theorem can be made to work in polynomial time. Let π be a $\mathbf{G}_0$ proof of a sequent

$$\longrightarrow \exists x_1 \ldots \exists x_m C(\vec{p}, x_1, \ldots, x_m) \tag{131}$$

where $C(\vec{p}, x_1, \ldots, x_m)$ is quantifier-free. Since all cut formulas in π are quantifier-free, it follows that every quantified formula in π is an ancestor of the conclusion, and must occur on the RHS and must have the form

$$\exists x_k \ldots \exists x_m C(\vec{p}, B_1 \ldots B_{k-1}, x_k, \ldots, x_m) \tag{132}$$

for some quantifier-free formulas $B_1, \ldots, B_{k-1}$ and some k, $1 \leq k \leq m$. Let us call a formula a *π-prototype* if it is quantifier-free and is the auxiliary formula in an $\exists$-right rule (so it is the quantifier-free parent of a formula of the form (132), with $k = m + 1$). Thus a π-prototype has the form $C(\vec{p}, B_1 \ldots B_m)$.

The *Herbrand π disjunction* S_π is the sequent

$$\longrightarrow A_1, \ldots, A_h$$

where $A_1, \ldots, A_h$ is a list of all the π-prototypes. It turns out that S_π is a valid sequent, and in fact π can be transformed into a **PK** proof π' of S_π in polynomial time. To form π' from π, delete each quantified formula (i.e. each formula of the form (132)) from π and add formulas from the list $A_1, \ldots, A_h$ to the RHS of each sequent so that each π-prototype is in the succedent of every sequent. The result can be turned into a **PK** proof of S_π by deleting applications of the rule $\exists$-right, and adding weakenings, exchanges, and contractions.

We may assume that the **PK** proof π' of S_π is treelike, by Theorem 7.8. Now π' is easily augmented to a treelike proof of (131) using the rules $\exists$-right, exchange and contraction. $\dashv$

We now show that for $\mathbf{G}_i^\star$ we may as well assume that all cut formulas are prenex Σ_i^q. We start by proving an easy lemma which applies to both $\mathbf{G}_i$ and $\mathbf{G}_i^\star$.

LEMMA 7.44. *If $\mathbf{G}_i$ (resp. $\mathbf{G}_i^\star$) is modified so that cuts are restricted to Σ_i^q-formulas, then the resulting system* p*-simulates $\mathbf{G}_i$ (resp. $\mathbf{G}_i^\star$).*

PROOF. If A is a Π_i^q formula, then any application of the cut rule to A can be replaced by first moving A to the opposite side of each parent sequent using $\neg$ introduction, and then cutting $\neg A$. $\dashv$

THEOREM 7.45 (Morioka [61]). *Let $\hat{\mathbf{G}}_i^\star$ be $\mathbf{G}_i^\star$ with cut formulas restricted to prenex Σ_i^q formulas. Then $\hat{\mathbf{G}}_i^\star$* p*-simulates $\mathbf{G}_i^\star$.*

Proof. Fix $i \geq 1$. Let π be a $\mathbf{G}_i^\star$ proof. We may assume that π is in free variable normal form.

Consider an application of the cut rule in π, with cut formula A.

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

We may assume that A is Σ_i^q, since if A is Π_i^q we can simply insert $\neg$-introduction steps just before the cut so that the cut formula becomes $\neg A$. Our task is to show that this cut on A can be replaced with a cut on A', where A' is some prenex form of A. To do this we will replace the tree derivation of $\Gamma \longrightarrow \Delta, A$ with a similar derivation of $\Gamma \longrightarrow \Delta, A'$, and similarly replace the derivation of $A, \Gamma \longrightarrow \Delta$ by one of $A', \Gamma \longrightarrow \Delta$.

The proof of the Prenex Form Theorem 2.75 lists ten equivalences as follows:

$$\begin{array}{rr}
(\forall x B \wedge C) \Longleftrightarrow \forall x(B \wedge C) & (\forall x B \vee C) \Longleftrightarrow \forall x(B \vee C) \\
(C \wedge \forall x B) \Longleftrightarrow \forall x(C \wedge B) & (C \vee \forall x B) \Longleftrightarrow \forall x(C \vee B) \\
(\exists x B \wedge C) \Longleftrightarrow \exists x(B \wedge C) & (\exists x B \vee C) \Longleftrightarrow \exists x(B \vee C) \\
(C \wedge \exists x B) \Longleftrightarrow \exists x(C \wedge B) & (C \vee \exists x B) \Longleftrightarrow \exists x(C \vee B) \\
\neg \forall x B \Longleftrightarrow \exists x \neg B & \neg \exists x B \Longleftrightarrow \forall x \neg B
\end{array}$$

(where x does not occur free in C).

To put a formula in prenex form (which is in the same class Σ_j^q or Π_j^q with the original formula), it suffices to successively transform a formula $A(B(\vec{x}))$ to $A(B'(\vec{x}))$, where $B \Longleftrightarrow B'$ is one of the above equivalences and $\vec{x}$ is a list of the variables in B which are bound by quantifiers in A.

Consider a derivation of $\Gamma \longrightarrow \Delta, A(B(\vec{x}))$ or $A(B(\vec{x})), \Gamma \longrightarrow \Delta$ in π. If we trace the ancestors of $A(B(\vec{x}))$ up through this derivation, each path either ends when the ancestor is formed by a weakening, or it includes an occurrence of $B(\vec{D})$, where $\vec{D}$ is the list of target formulas and eigenvariables used by the quantifier introduction rules in forming $A(B(\vec{x}))$ from $B(\vec{D})$.

Thus it suffices to show, for each of the above equivalences $B \Longleftrightarrow B'$, how to convert a derivation of $\Lambda \longrightarrow \Pi, B$ to one of $\Lambda \longrightarrow \Pi, B'$ and a derivation of $B, \Lambda \longrightarrow \Pi$ to one of $B', \Lambda \longrightarrow \Pi$. (In the application to the previous paragraph, B would be $B(\vec{D})$, and B' would be $B'(\vec{D})$.)

Consider, for example, converting a derivation of

$$\Lambda \longrightarrow \Pi, \neg \forall x C(x)$$

to one of

$$\Lambda \longrightarrow \Pi, \exists x \neg C(x)$$

The ancestral paths of $\neg \forall x C(x)$ which do not end in weakening include $\forall x C(x)$ in the antecedent and then $C(D)$ in the antecedent, for some target formula D. Thus we have arrived at a sequent

$$C(D), \Lambda' \longrightarrow \Pi'$$

We modify the derivation after this point by using $\neg$-right and $\exists$-right to obtain

$$\Lambda' \longrightarrow \Pi', \exists x \neg C(x)$$

and continue the derivation as before, omitting the steps which formed $\neg\forall x C(x)$ from $C(D)$.

The argument is similar if $\neg\forall x C(x)$ is in the antecedent.

Now consider converting a derivation of

$$\Lambda \longrightarrow \Pi, \forall x C(x) \wedge D$$

to a derivation of

$$\Lambda \longrightarrow \Pi, \forall x(C(x) \wedge D)$$

The ancestral paths of $\forall x C(x) \wedge D$ which do not end in weakening split after an $\wedge$-right, where the left branch has a $\forall$-right step

$$\frac{\Lambda' \to \Pi', C(p)}{\Lambda' \to \Pi', \forall x C(x)}$$

We modify this by combining it with the right branch just after the split as follows:

$$\frac{\dfrac{\Lambda'' \longrightarrow \Pi'', C(p) \qquad \Lambda'' \longrightarrow \Pi'', D}{\Lambda'' \longrightarrow \Pi'', C(p) \wedge D}}{\Lambda'' \longrightarrow \Pi'', \forall x(C(x) \wedge D)}$$

Here it is important that the original derivation be in free variable normal form, both in order to insure that p does not occur in D, and to guarantee that the variable restrictions continue to hold in the modified derivation of $\Lambda \longrightarrow \Pi, \forall x(C(x) \wedge D)$.

The other cases are handled similarly. $\dashv$

A part of the reverse direction of Theorem 7.41 is shown in the next theorem.

Theorem 7.46 (Perron [70]). *For $i \geq 1$, $\mathbf{G}_i$ p-simulates $\mathbf{G}^\star_{i+1}$(for all formulas).*

From this theorem and Theorem 7.41 we have:

Corollary 7.47. *For $i \geq 1$, $\mathbf{G}_i$ and $\mathbf{G}^\star_{i+1}$ are p-equivalent for proving formulas in $\Sigma^q_i \cup \Pi^q_i$.*

Proof of Theorem 7.46. Let π be a $\mathbf{G}^\star_{i+1}$ proof of a formula A. We show how to get a suitable $\mathbf{G}_i$ proof π' of A from π. The idea is to replace cuts of formulas C not in $\Pi^q_i \cup \Sigma^q_i$ by cuts on simpler ancestors of C. By Theorem 7.45 we can assume that all cut formulas in π are prenex Σ^q_{i+1} formulas. Furthermore, we can assume that π is in free variable normal form.

Assume that in π for all axioms of the form

$$B \longrightarrow B$$

the formula B is quantifier free. This is possible because from these axioms we can easily derive any axiom with quantified formulas. Similarly assume that only quantifier free formulas are used for the weakening rules.

An occurrence of a formula $\exists\vec{x}B(\vec{x})$ in π is said to be *tagged* if it occurs in the antecedent Γ of a sequent

$$\mathcal{S} = \Gamma \longrightarrow \Delta$$

and B is in $(\Pi_i^q - \Sigma_i^q)$ and some descendant in π of $\exists\vec{x}B(\vec{x})$ is cut. Let

$$B(\overrightarrow{q^1}), B(\overrightarrow{q^2}), \ldots, B(\overrightarrow{q^k}) \tag{133}$$

be all $(\Pi_i^q - \Sigma_i^q)$ ancestors of $\exists\vec{x}B(\vec{x})$, where the variables $\overrightarrow{q^i}$ are eigenvariables in π. By our assumptions above, every $(\Sigma_{i+1}^q - \Pi_i^q)$ ancestor of $\exists\vec{x}B(\vec{x})$ lies on a path from some sequent containing some $B(\overrightarrow{q^i})$ to $\mathcal{S}$.

Define

$$\mathcal{S}' = \Gamma' \longrightarrow \Delta$$

where Γ' is obtained from Γ by replacing *every* tagged formula $\exists\vec{x}B(\vec{x})$ in Γ (possibly for more than one formula $B(\vec{x})$) by its corresponding list (133). By free variable normal form, the eigenvariables $\overrightarrow{q^i}$ associated with distinct tagged formulas in Γ are distinct. Notice that $\mathcal{S}'$ has size bounded by the size of π.

We will describe a polynomial time algorithm which successively transforms, for each sequent $\mathcal{S}$ in π, the (treelike) derivation $\pi_{\mathcal{S}}$ of $\mathcal{S}$ to a daglike $\mathbf{G}_i$ derivation $\pi'_{\mathcal{S}}$ of $\mathcal{S}'$. Note that if $\mathcal{S}$ is the final sequent in π then $\mathcal{S}' = \mathcal{S}$, and the theorem is proved.

The algorithm starts with the leaves of the proof tree π and works its way down to the endsequent. The leaf sequents are axioms, which by our assumptions have no tagged formulas, so there is nothing to do. For the general step we need to consider the rule used to derive $\mathcal{S}$. If the principle formula in the rule is not tagged, then $\pi'_{\mathcal{S}}$ is constructed using the same rule applied to the transformed proof(s) of the parent(s). If the principle formula is tagged, then the rule cannot be weakening by our assumptions, so it must be one of $\exists$-left, contraction-left, or cut. For $\exists$-left or contraction-left there is nothing to do: just use the transformed proof of the parent sequent.

Hence the only non-trivial case is where $\mathcal{S}$ is derived by cutting a tagged formula. So suppose that $\mathcal{S}_3$ is a sequent in π and is derived from $\mathcal{S}_1$ and $\mathcal{S}_2$ as below:

$$\frac{\mathcal{S}_1 \qquad \mathcal{S}_2}{\mathcal{S}_3} = \frac{\Gamma \longrightarrow \Delta, \exists\vec{x}B(\vec{x}) \qquad \exists\vec{x}B(\vec{x}), \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta}$$

Here $B(\vec{x})$ is a formula in $(\Pi_i^q - \Sigma_i^q)$. Suppose that

$$\mathcal{S}_3' = \Gamma' \longrightarrow \Delta$$

Then note that

$$\mathcal{S}_1' = \Gamma' \longrightarrow \Delta, \exists\vec{x}B(\vec{x})$$

and $\mathcal{S}_2'$ has the form

$$\mathcal{S}_2' = B(\overrightarrow{q^1}), B(\overrightarrow{q^2}), \ldots, B(\overrightarrow{q^k}), \Gamma' \longrightarrow \Delta$$

where no eigenvariable in any $\overrightarrow{q_i}$ occurs in Γ' or Δ. We have previously found short $\mathbf{G}_i$ derivations $\pi'_{\mathcal{S}_1}$, $\pi'_{\mathcal{S}_2}$ of the sequents $\mathcal{S}_1'$, $\mathcal{S}_2'$. The idea is to convert $\pi'_{\mathcal{S}_1}$ into a $\mathbf{G}_i$ derivation of $\Gamma' \longrightarrow \Delta$ by cutting 'topmost' ancestors of $\exists\vec{x}B(\vec{x})$ using substitution instances of $\mathcal{S}_2'$.

First we add Γ' to the antecedent and Δ to the succedent of every sequent in $\pi'_{\mathcal{S}_1}$ (and add necessary weakenings to have a legitimate proof). Call the result $\pi''_{\mathcal{S}_1}$.

Now consider a sequent

$$\mathcal{S}_{11} \;=\; \Lambda \longrightarrow \Pi, B(\vec{C}) \tag{134}$$

in π where $B(\vec{C})$ is an ancestor of $\exists\vec{x}B(\vec{x})$ in $\mathcal{S}_1$. (Here $\vec{C}$ consists of Σ_0^q formulas.) We say that $B(\vec{C})$ is a *topmost* ancestor if it has no further ancestor $B(\vec{C})$ in π; i.e. $B(\vec{C})$ is the principle formula in the $\forall$-right rule used to derive the sequent (134). In $\pi''_{\mathcal{S}_1}$ $\mathcal{S}_{11}'$ has become

$$\Gamma', \Lambda' \longrightarrow \Delta, \Pi, B(\vec{C})$$

Apply the Substitution Lemma 7.48 below and using contractions left we create for each topmost ancestor $B(\vec{C})$ of $\exists\vec{x}B(\vec{x})$ a $\mathbf{G}_i^\star$ derivation of the form

$$\frac{\mathcal{S}_2'}{B(\vec{C}), \Gamma' \longrightarrow \Delta} \tag{135}$$

(Since there may be more than one topmost ancestor with different formulas $\vec{C}$, the sequent $\mathcal{S}_2'$ may have to be used more than once, which is why our transformed proof may not be treelike). For each topmost ancestor $B(\vec{C})$ in turn, working from the top of π down, insert the following derivation in $\pi''_{\mathcal{S}_1}$:

$$\frac{\Gamma', \Lambda' \longrightarrow \Delta, \Pi, B(\vec{C}) \qquad B(\vec{C}), \Gamma' \longrightarrow \Delta}{\Gamma', \Lambda' \longrightarrow \Pi, \Delta} \tag{136}$$

(where the upper right sequent is derived by (135)) and remove all descendants of $B(\vec{C})$ in the so-far transformed $\pi''_{\mathcal{S}_1}$ as far as possible. If a descendant is the principle formula in a contraction then simply delete that contraction rule. If a descendant is a side formula in a two-parent rule, then progress must wait until the matching side formula in the other parent is removed. When this is done for each topmost ancestor, all descendants of the form $\exists\vec{x}B(\vec{x})$ will be removed, and we obtain a proof of the sequent

$$\Gamma', \Gamma' \longrightarrow \Delta, \Delta$$

With additional applications of the contraction rules we obtain a legitimate derivation $\pi'_{\mathcal{S}_3}$ of $\mathcal{S}'_3$.

Finally we verify that the final $\mathbf{G}_i$ proof π' has size polynomial in the size of π. Notice that all new sequents have size polynomial in the size of π. (The bottom sequent in (135) is the only sequent that might have size larger than π.) So it remains to show that the number of sequents in π' is bounded by a polynomial in the size $|\pi|$ of π.

For a sequent $\mathcal{S}$ in π let $n_{\mathcal{S}'}$ denote the number of sequents used in the derivation of $\mathcal{S}'$ in π'. Consider the interesting case of the cut rule in the algorithm above. It suffices to show that for some polynomial $\mathbf{p}$ we have

$$n_{\mathcal{S}'_3} \le n_{\mathcal{S}'_1} + n_{\mathcal{S}'_2} + \mathbf{p}(|\pi|)$$

This follows from the fact that for each sequent $\mathcal{S}$ (134) in π the total number of sequents in the derivations (135) and (136), as well as the number of applications of weakening and contraction rules described above are bounded above by some polynomial in $|\pi|$ independent of $\mathcal{S}_3$. ⊣

Lemma 7.48 (Substitution Lemma). *There is a polynomial size $\mathbf{G}_i^\star$ derivation*

$$\frac{\Gamma(p), \Gamma' \longrightarrow \Delta(p), \Delta'}{\Gamma(B), \Gamma' \longrightarrow \Delta(B), \Delta'} \tag{137}$$

where B is a quantifier-free formula, all formulas in Γ and Δ are in $\Sigma_i^q \cup \Pi_i^q$, and p does not occur in the bottom sequent.

To prove the above lemma we need:

Lemma 7.49 ($\mathbf{G}_0^\star$-Replacement Lemma). *Let $A(p)$ be a quantified propositional formula, and let $A(B)$ be the result of substituting the formula B for p in $A(p)$. Then for all formulas B_1, B_2, the sequent*

$$B_1 \leftrightarrow B_2 \longrightarrow A(B_1) \leftrightarrow A(B_2)$$

has a $\mathbf{G}_0^\star$ proof of size bounded by a polynomial in the size of its endsequent.

Exercise 7.50. *Prove the Lemma. (See Exercise 7.11.)*

Proof of the Substitution Lemma. From the $\mathbf{G}_0^\star$-Replacement Lemma above, we have a $\mathbf{G}_0^\star$ proof of

$$p \leftrightarrow B, A(p) \longrightarrow A(B)$$

for each formula $A(p)$ in $\Delta(p)$. From these and

$$\Gamma(p), \Gamma' \longrightarrow \Delta(p), \Delta'$$

we obtain (by the cut rule on the formulas $A(p)$ in $\Delta(p)$)

$$p \leftrightarrow B, \Gamma(p), \Gamma' \longrightarrow \Delta(B), \Delta' \tag{138}$$

Again, by the $\mathbf{G}_0^\star$-Replacement Lemma, we have $\mathbf{G}_0^\star$ derivations of

$$p \leftrightarrow B, A(B) \longrightarrow A(p)$$

for all formulas $A(p)$ in $\Gamma(p)$. From these and (138) we obtain

$$p \leftrightarrow B, \Gamma(B), \Gamma' \longrightarrow \Delta(B), \Delta'$$

Now by the $\exists$-left rule we get

$$\exists x(x \leftrightarrow B), \Gamma(B), \Gamma' \longrightarrow \Delta(B), \Delta'$$

Finally, it is easy to see that the sequent

$$\longrightarrow \exists x(x \leftrightarrow B)$$

can be derived in $\mathbf{G}_0^\star$. Consequently, by the cut rule on the Σ_1^q formula $\exists x(x \leftrightarrow B)$ we obtain the bottom sequent of (137). It is clear that all the derivations above have size polynomial in the length of the endsequents. ⊣

Unlike the situation for **PK** and $\mathbf{G}_0$, it seems unlikely that $\mathbf{G}_1^\star$ p-simulates $\mathbf{G}_1$. To explain why, we need the notion of witnessing for QPC proof systems.

7D.1. Extended Frege Systems and Witnessing in $\mathbf{G}_1^\star$. In previous chapters we proved witnessing theorems which concern the complexity of witnessing the leading existential quantifiers in a bounded $\mathcal{L}_A^2$ formula, given values for the free variables. The analogous witnessing problem for a QPC formula is trivial, because there are only finitely many possible values for the free variables. However the problem becomes interesting if we consider a family of formulas, and include a proof of the formula as part of the input.

Theorem 7.51 (The Witnessing Theorem for $\mathbf{G}_1^\star$). *There is a polynomial time function $F(\pi, \tau)$ which, given a $\mathbf{G}_1^\star$ proof π of a formula of the form $\exists \vec{x} A(\vec{x}, \vec{p})$ (where $A(\vec{x}, \vec{p})$ is quantifier-free) and an assignment τ to $\vec{p}$, returns an extension τ' of τ such that τ' satisfies $A(\vec{x}, \vec{p})$.*

We show in Theorem 10.57 that if π is a $\mathbf{G}_1$ proof (as opposed to a $\mathbf{G}_1^\star$ proof), then the witnessing problem becomes complete for the search class **PLS** (Polynomial Local Search). Since it seems unlikely that **PLS** problems can all be solved in polynomial time, it seems unlikely that $\mathbf{G}_1^\star$ p-simulates $\mathbf{G}_1$.

In general the problem of computing such τ' from τ without π is complete for $\mathbf{P}^{\mathbf{NP}}$, if we are required to say “no” if there is no witness. Hence it is clear that the proof π provides helpful information.

We will prove the Witnessing Theorem for $\mathbf{G}_1^\star$ by analyzing a closely-related system **ePK**, a member of the class of *extended* **Frege** proof systems. In general, a line in an extended **Frege** proof has the expressive power of a Boolean circuit, and a problem in nonuniform **P** is presented by a polynomial size family of Boolean circuits. The connection between the extended **Frege** proof systems and **P** is thus analogous to that of the bounded depth **Frege** proof systems (e.g., **bPK**) and $\mathbf{AC}^0$ that we have seen (Section 7B), or that of the **Frege** systems and $\mathbf{NC}^1$, as we discussed in the Preface.

Definition 7.52 (Extension Cedent). *The sequence of formulas*

$$\Lambda = \qquad e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_n \leftrightarrow B_n \tag{139}$$

is an extension cedent *provided that for* $i = 1, ..., n$*, the atom* e_i *does not occur in any of the formulas* $B_1, ..., B_i$*. The atoms* $e_1, ..., e_n$ *are called* extension variables*.*

Intuitively, we think of $e_1, ..., e_n$ as gates in a Boolean circuit, where the value of e_i is determined by B_i together with the values of the earlier gates $e_1, \ldots, e_{i-1}$. In an **ePK** proof of an existential statement, some of these extension variables are used to witness the existential quantifiers.

Definition 7.53 (**ePK** Proof). *Let* $\exists\vec{x}A(\vec{x},\vec{p})$ *be a QPC formula with free variables* $\vec{p}$ *such that* $A(\vec{x},\vec{p})$ *is quantifier-free. An* **ePK** *proof of* $\exists\vec{x}A(\vec{x},\vec{p})$ *is a* **PK** *proof of any sequent of the form*

$$\Lambda \longrightarrow A(\vec{e}_1,\vec{p})$$

where Λ *is an extension cedent* (139) *in which the extension variables* $\vec{e}$ *are disjoint from* $\vec{p}$*,* $\vec{e}_1$ *is a subset of* $\vec{e}$*, and each* B_i *contains only variables among* $\vec{e},\vec{p}$*.*

This definition is interesting even in the case that the final formula is quantifier-free. Then the extension variables are not used to witness quantifiers, but they still may be useful in defining polynomial time concepts needed in the proof. As far as we know, **PK** does not p-simulate **ePK** even when the latter is restricted to proving quantifier-free formulas.

Theorem 7.54 (Krajíček [55]). $\mathbf{G}_1^\star$*, restricted to proving prenex* Σ_1^q *formulas, is* p*-equivalent to* **ePK***.*

Before giving the proof, we show how the Witnessing Theorem for $\mathbf{G}_1^\star$ follows from this.

Proof of Theorem 7.51. Let π be a $\mathbf{G}_1^\star$ proof of $\exists\vec{x}A(\vec{x},\vec{p})$, and let τ be an assignment to $\vec{p}$, as in the statement of the Witnessing Theorem. By the preceding theorem, we can transform π to an **ePK** proof of $\exists\vec{x}A(\vec{x},\vec{p})$; that is, a **PK** proof of a sequent

$$e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_n \leftrightarrow B_n \longrightarrow A(\vec{e}_1,\vec{p}) \tag{140}$$

Now given the the assignment τ to $\vec{p}$, values for $e_1, e_2, ..., e_n$ can be computed successively by evaluating $B_1, ..., B_n$, and these values define the desired extension τ' of τ which satisfies $A(\vec{x}, \vec{p})$. ⊣

Proof of Theorem 7.54. First we show that $\mathbf{G}_1^\star$ p-simulates **ePK**. Let π be a (treelike) **ePK** proof of $\exists\vec{x}A(\vec{x}, \vec{p})$. Then π is a **PK** proof of a sequent of the form (140). We show how to extend this **PK** proof to make a $\mathbf{G}_1^\star$ proof of $\exists\vec{x}A(\vec{x}, \vec{p})$. We start by repeated application of $\exists$-right to obtain a proof of

$$e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_n \leftrightarrow B_n \longrightarrow \exists\vec{x}A(\vec{x}, \vec{p}) \tag{141}$$

Now for each formula B there is a short **PK** proof of $\longrightarrow (B \leftrightarrow B)$, and with one application of $\exists$-right we obtain a short $\mathbf{G}_1^\star$ proof of

$$\longrightarrow \exists x(x \leftrightarrow B) \tag{142}$$

Now apply $\exists$-left to (141) to change the formula $(e_n \leftrightarrow B_n)$ to $\exists x(x \leftrightarrow B_n)$. (Note that e_n does not occur elsewhere in (141), so the variable restriction for this rule is satisfied.) Now apply the cut rule to this and (142) to obtain

$$e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_{n-1} \leftrightarrow B_{n-1} \longrightarrow \exists\vec{x}A(\vec{x}, \vec{p})$$

Applying this process a total of n times we may eliminate each formula $e_i \leftrightarrow B_i$ in (141) to obtain the desired $\mathbf{G}_1^\star$ proof of size polynomial in the size of π.

Now we prove the converse. Let π be a $\mathbf{G}_1^\star$ proof of $\longrightarrow \exists\vec{x}A(\vec{x}, \vec{p})$. We may assume that π is in free variable normal form, and by Theorem 7.45 we may assume that all cut formulas in π are prenex Σ_1^q, so each sequent of π has the form

$$S = \quad \ldots, \exists\vec{x^i}\alpha_i(\vec{x^i}, \vec{r}), \ldots, \Gamma \longrightarrow \Delta, \ldots, \exists\vec{y^j}\beta_j(\vec{y^j}, \vec{r}), \ldots \tag{143}$$

where all α_i and β_j as well as all formulas in Γ and Δ are quantifier-free, and $\vec{r}$ is precisely the list of the free variables occurring in S. Notice that $\vec{r}$ may have variables not in $\vec{p}$, which are used as eigenvariables for $\exists$-left.

We transform the proof π to an **ePK** proof π' by transforming each such sequent S to a corresponding quantifier-free sequent S', and supplying a suitable proof of S'. To describe S', we first replace each vector $\vec{x^i}$ of bound variables by a distinct vector $\vec{q^i} = q_1^i, \ldots, q_{\ell_i}^i$ of new free variables, and similarly we replace $\vec{y^i}$ by a new vector $\vec{e^i}$. None of these new variables should occur in π. Then

$$S' = \quad \Lambda, \ldots, \alpha_i(\vec{q^i}, \vec{r}), ..., \Gamma \longrightarrow \Delta, ..., \beta_j(\vec{e^j}, \vec{r}), ..., \tag{144}$$

where Λ is an extension cedent defining the extension variables $\ldots, \vec{e^j}, \ldots$.

If S is the endsequent $\longrightarrow \exists\vec{x}A(\vec{x}, \vec{p})$, then S' has the form $\Lambda \longrightarrow A(\vec{e}_1, \vec{p})$, so π' is the desired **ePK** proof of $\exists\vec{x}A(\vec{x}, \vec{p})$.

We define Λ and show that S' has an **ePK** proof polynomial in the size of the $\mathbf{G}_1^\star$ proof of S, by induction on the depth of S in π.

For the base case, S is an axiom

$$\exists\vec{x}\alpha(\vec{x},\vec{r}) \longrightarrow \exists\vec{x}\alpha(\vec{x},\vec{r})$$

and S' is easy to obtain.

For the induction step there is one case for each rule of $\mathbf{G}_1^\star$.

Case I: Weakening and exchange are trivial, and contraction follows from cut. The single parent rules $\neg$ and $\wedge$-left and $\vee$-right are easy, since the principle formulas are quantifier-free, and the same rule can be applied to form S'.

Case II: For the two parent rules $\wedge$-right and $\vee$-left, the principle formulas are quantifier-free, but we face the difficulty that the extension cedents Λ for the two parents may give inconsistent definitions of the extension variables. This is similar to the difficulty for **Case VII** in the proof of Lemma 5.65 for the $\mathbf{V}^0$ witnessing theorem. There the witnessing functions for a formula in Π for the two parents might be different. We solve the problem in a similar way, by defining the extension variables to values that make them true when possible.

Specifically, consider the case of $\wedge$-right, where for simplicity we assume there is exactly one formula in the succedent beginning with existential quantifiers (that formula cannot be C or D):

$$\frac{S_1 \qquad S_2}{S} = \frac{\Gamma \longrightarrow \Delta, \exists\vec{y}\beta(\vec{y},\overrightarrow{r^1}), C \qquad \Gamma \longrightarrow \Delta, \exists\vec{y}\beta(\vec{y},\overrightarrow{r^2}), D}{\Gamma \longrightarrow \Delta, \exists\vec{y}\beta(\vec{y},\vec{r}), (C \wedge D)}$$

where $\vec{r}$ is the union of the lists $\overrightarrow{r^1}, \overrightarrow{r^2}$. By the induction hypothesis, we have **ePK** proofs of the two sequents

$$S_1' = \Lambda_1, \Gamma' \longrightarrow \Delta, \beta(\vec{e},\overrightarrow{r^1}), C$$

and

$$S_2' = \Lambda_2, \Gamma' \longrightarrow \Delta, \beta(\vec{s},\overrightarrow{r^2}), D$$

where in the the second case we have changed the extension variables from $\vec{e}$ to $\vec{s}$. Since π is treelike, we can assume that the **ePK** derivations of S_1' and S_2' are disjoint, and hence we can change variable names in one proof without affecting the other proof. Thus we may assume that the extension variables defined in Λ_1 and Λ_2 are disjoint, and in particular $\vec{e}$ and $\vec{s}$ have no variable in common. Thus the extension cedents Λ_1 and Λ_2 are consistent. Further we may assume that the variables $\vec{q^i}$ are the same in S_1' and S_2'.

From S_1' and S_2' with $\wedge$-right we obtain

$$\Lambda_1, \Lambda_2, \Gamma' \to \Delta, \beta(\vec{e},\vec{r}), \beta(\vec{s},\vec{r}), (C \wedge D) \tag{145}$$

Now we introduce new extension variables $\vec{t}$, and introduce the extension formulas

$$E_i =_{\mathrm{def}} \big((\beta(\vec{e},\vec{r}) \wedge e_i) \vee (\neg\beta(\vec{s},\vec{r}) \wedge s_i)\big)$$

and define the extension cedent

$$\Lambda_3 = t_1 \leftrightarrow E_1, t_2 \leftrightarrow E_2, ...$$

Then define

$$S' = \qquad \Lambda_1, \Lambda_2, \Lambda_3, \Gamma' \longrightarrow \Delta, \beta(\vec{t}, \vec{r}), (C \wedge D)$$

One can show with the help of Lemma 7.10 that each of the sequents

$$\Lambda_3, \beta(\vec{e}, \vec{r}) \longrightarrow \beta(\vec{t}, \vec{r}) \tag{146}$$

$$\Lambda_3, \beta(\vec{s}, \vec{r}) \longrightarrow \beta(\vec{t}, \vec{r}) \tag{147}$$

has a short **PK** proof. Using these and (145) and two cuts we obtain a short **PK** derivation of S' from S_1 and S_2.

Case III: $\exists$-left is easy, since it just means changing the role of a free eigenvariable r in S'_1 to the variable q in S' corresponding to $\exists x$.

Case IV: Suppose S comes from S_1 using $\exists$-right.

$$\frac{S_1}{S} = \qquad \frac{\Gamma \to \Delta, \exists\vec{y}\beta(B, \vec{y}, \vec{r})}{\Gamma \to \Delta, \exists z\exists\vec{y}\beta(z, \vec{y}, \vec{r})}$$

Here the target formula B is quantifier-free, by definition of **G**. Since π is in free variable normal form, no free variable can be eliminated by this rule, and so the list $\vec{r}$ of free variables in S is the same as for S_1. By the induction hypothesis, we have an **ePK** derivation of

$$S'_1 = \quad \Lambda, \Gamma' \to \Delta', \beta(B, \vec{e}, \vec{r})$$

Let s be a new extension variable, and let

$$S' = \quad \Lambda,\ s \leftrightarrow B,\ \Gamma' \to \Delta', \beta(s, \vec{e}, \vec{r})$$

It follows from the **PK**-Replacement Lemma 7.10 that S' has a short **PK** derivation from S'_1.
Case V: Suppose S comes from S_1, S_2 by cut:

$$\frac{S_1 \qquad S_2}{S_3} = \frac{\Gamma \longrightarrow \Delta, C \qquad C, \Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta}$$

Since π is in free variable normal form, every free variable in C also occurs in the conclusion S_3. Suppose first that the cut formula C is quantifier-free. Then the only difficulty is that the extension cedents Λ for the two parents may give inconsistent definitions of the extension variables witnessing quantifiers in Δ. We handle this difficulty in the same way as for **Case II** above.

The case in which C has existential quantifiers is more complicated, since the definitions of the new extension variables witnessing quantifiers in Δ now depend on witnesses for the quantifiers in C supplied by S'_1. These new definitions are similar to the new witnessing functions defined for the case of cut (**Case VI**) in the proof of Lemma 5.65 used to prove the $\mathbf{V}^0$ Witnessing Theorem. $\dashv$

Exercise 7.55. *Carry out the details of* **Case V** *in the above proof.*

7E. Propositional translations for $\mathbf{V}^i$

In this section we show that for $i \geq 1$, $\mathbf{G}_i^\star$ is closely related to the theory $\mathbf{V}^i$. In fact Theorem 7.57 together with results in Chapter 10 suggest that $\mathbf{G}_i^\star$ restricted to Σ_i^q formulas is a nonuniform version of the Σ_i^B-fragment of $\mathbf{V}^i$. We have already shown by Theorem 7.51 a connection between $\mathbf{G}_1^\star$ and $\mathbf{V}^1$: Σ_1^q-theorems of $\mathbf{G}_1^\star$ can be uniformly witnessed in polynomial time, just as each Σ_1^B-theorem of $\mathbf{V}^1$ can be witnessed in polynomial time.

It is straightforward to extend the propositional translation of $\Sigma_0^B(\mathcal{L}_A^2)$ formulas (Section 7B) to a translation of any bounded $\mathcal{L}_A^2$ formula. Here every $\mathbf{g}\Sigma_i^B$ (resp. $\mathbf{g}\Pi_i^B$) formula $\varphi(\vec{x}, \vec{X})$, with all free variables indicated, translates into a family of Σ_i^q (resp. Π_i^q) formulas:

$$\|\varphi(\vec{x}, \vec{X})\| = \{\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}] : \ \vec{m}, \vec{n} \in \mathbb{N}\}$$

so that $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ is valid iff

$$\underline{\mathbb{N}}_2 \models \forall \vec{X}\big(\big(\bigwedge |\vec{X}| = \underline{\vec{n}}\big) \supset \varphi(\underline{\vec{m}}, \vec{X})\big)$$

The formula $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ has size bounded by a polynomial $\mathbf{p}(\vec{m}, \vec{n})$ which depends only on φ. The free propositional variables in $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ consist of $p_j^{X_i}$, for $0 \leq j < n_i - 1$ for each $n_i \geq 2$.

We define the translation of a bounded $\mathcal{L}_A^2$ formula φ inductively, starting with the Σ_0^B formulas, which is described in Section 7B. For the induction step, consider the case where

$$\varphi(\vec{x}, \vec{X}, Y) \equiv \exists Y \leq t \psi(\vec{x}, \vec{X}, Y)$$

(here t is a number term of the form $t(\vec{x}, |\vec{X}|)$). By the induction hypothesis, $\psi(\vec{x}, \vec{X}, Y)[\vec{m}; \vec{n}, k]$ contains the free propositional variables $p_0^Y, p_1^Y, \ldots$ for Y, in addition to $p_j^{X_i}$ (when $k < 2$, the list $p_0^Y, \ldots, p_{k-2}^Y$ is empty). Define

$$\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}] =_{\text{def}} \exists p_0^Y \ldots \exists p_{r-2}^Y \bigvee_{k=0}^{r} \psi(\vec{x}, \vec{X}, Y)[\vec{m}; \vec{n}, k] \tag{148}$$

where r is the numerical value of t: $r = val(t(\underline{\vec{m}}, \underline{\vec{n}}))$ (recall that $\underline{i}$ is the i-th numeral). Here the free variables p_j^Y become bound, and if $r \leq 1$ then the list $p_0^Y, \ldots, p_{r-2}^Y$ is empty. Also, if any of the formulas $\psi_k(p_0^Y, \ldots, p_{k-2}^Y)$ is a logical constant $\bot$ or $\top$, then we simplify $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ in the obvious way.

The case where $\varphi(\vec{x},\vec{X}) \equiv \forall Y \le t\psi(\vec{x},\vec{X},Y)$ is similar:

$$\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}] =_{\mathrm{def}} \forall p_0^Y \dots \forall p_{r-2}^Y \bigwedge_{k=0}^{r} \psi(\vec{x},\vec{X},Y)[\vec{m};\vec{n},k] \tag{149}$$

(The conjunction is also simplified if any conjunct is a Boolean constant.)

The cases of the Boolean connectives $\wedge$, $\vee$, $\neg$ or the number quantifiers are the same as for Σ_0^B formulas.

PROPOSITION 7.56. *For each $i \ge 0$, if φ is a $\mathbf{g}\Sigma_i^B$ (resp. $\mathbf{g}\Pi_i^B$) formula, then the formulas in $\|\varphi\|$ are Σ_i^q (resp. Π_i^q). There is a polynomial $\mathbf{p}(\vec{m},\vec{n})$ which depends only on φ so that $\varphi[\vec{m};\vec{n}]$ has size $\le \mathbf{p}(\vec{m},\vec{n})$ for all $\vec{m},\vec{n} \in \mathbb{N}$.*

The connection between the theory $\mathbf{V}^i$ and the proof system $\mathbf{G}_i^\star$ is as follows.

THEOREM 7.57 ($\mathbf{V}^i$ Translation Theorem). *Let $i \ge 1$. For any bounded theorem $\varphi(\vec{x},\vec{X})$ of $\mathbf{V}^i$, there is a polytime function $F(\vec{m},\vec{n})$ such that $F(\vec{m},\vec{n})$ is a $\mathbf{G}_i^\star$ proof of $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$, for all $\vec{m},\vec{n} \in \mathbb{N}$.*

PROOF. The proof is similar to that of the Translation Theorem for $\mathbf{V}^0$ 7.20. We consider the case where $i = 1$; the cases where $i > 1$ are handled in the same way. By Corollary 6.43, for every bounded theorem $\varphi(\vec{a},\vec{\alpha})$ of $\mathbf{V}^1$ there is a (treelike) anchored $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ proof π of $\longrightarrow \varphi(\vec{a},\vec{\alpha})$. If we translate each sequent of π into the corresponding QPC sequent, the result is close to a $\mathbf{G}_1^\star$ proof. In particular, since any cut formula in $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ is Σ_1^B, its translation is a Σ_1^q formula, and can be cut in $\mathbf{G}_1^\star$.

Formally, we will prove by induction on the depth of a sequent $\mathcal{S}(\vec{b},\vec{\beta})$ in π that there is a polytime function $F(\vec{m},\vec{n})$ such that $F(\vec{m},\vec{n})$ is a $\mathbf{G}_1^\star$ proof of $\mathcal{S}[\vec{m};\vec{n}]$. For the base case, $\mathcal{S}$ is an axiom of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$. The simple axioms are sequents of Σ_0^B formulas, and these are treated as in the proof of the Translation Theorem for $\mathbf{V}^0$. The remaining axioms are instances of Σ_0^B-**COMP**, so

$$\mathcal{S} = \quad \longrightarrow \exists X \le t \forall z < t(X(z) \leftrightarrow \eta(z))$$

and η is a Σ_0^B formula. Let $r = val(t)$. When $r \le 1$, it is easy to see that $\mathcal{S}$ translates into a trivially valid sequent with a short $\mathbf{G}_0$ proof. Otherwise, if $r \ge 2$, then $\mathcal{S}[\vec{m};\vec{n}]$ is the sequent

$$\longrightarrow \big(\exists X \le t \forall z < t(X(z) \leftrightarrow \eta(z))\big)[\vec{m};\vec{n}]$$

where (replace $[\dots]$ by $[\vec{m};\vec{n}]$):

$$\big(\exists X \le t \forall z < t(X(z) \leftrightarrow \eta(z))\big)[\dots] \equiv \exists p_0^X \dots \exists p_{r-2}^X$$
$$\bigvee_{k=0}^{r} \Big(\bigwedge_{i=0}^{k-2} (p_i^X \leftrightarrow \eta(\underline{i})[\dots]) \ \wedge\ \eta(\underline{k-1})[\dots] \ \wedge\ \bigwedge_{i=k}^{r-1} \neg\eta(\underline{i})[\dots]\Big)$$

where the conjunct $\eta(\underline{k-1})$ is deleted when $k = 0$ and the conjuncts

$$\bigwedge_{i=0}^{k-2} (p_i^X \leftrightarrow \eta(\underline{i})[\ldots]) \qquad \text{and} \qquad \bigwedge_{i=k}^{r-1} \neg\eta(\underline{i})[\ldots]$$

are deleted when their sets of indices are empty.

Exercise 7.58. *Let $A_0, \ldots, A_\ell$ be any* **PK** *formulas ($\ell \geq 0$). Show that the sequent*

$$\longrightarrow \bigwedge_{i=0}^{\ell} \neg A_i, A_0 \wedge \bigwedge_{i=1}^{\ell} \neg A_i, A_1 \wedge \bigwedge_{i=2}^{\ell} \neg A_i, \ldots, A_{\ell-1} \wedge \neg A_\ell, A_\ell$$

has a polynomial size treelike cut-free **PK** *derivation.*

We get $\mathcal{S}[\vec{m};\vec{n}]$ as follows. First we apply the above exercise for $\ell = r-1$ and

$$A_i \equiv \eta(\underline{i})[\vec{m};\vec{n}]$$

Then note that it is straightforward to obtain polynomial size derivations for the following tautologies:

$$\bigwedge_{i=0}^{k-2} (\eta(\underline{i})[\ldots] \leftrightarrow \eta(\underline{i})[\ldots])$$

Now by using the $\wedge$-right and $\vee$-right rules obtain

$$\longrightarrow \bigvee_{k=0}^{r} \Big(\bigwedge_{i=0}^{k-2} (\eta(\underline{i})[\ldots] \leftrightarrow \eta(\underline{i})[\ldots]) \ \wedge \ \eta(\underline{k-1})[\ldots] \ \wedge \ \bigwedge_{i=k}^{r-1} \neg\eta(\underline{i})[\ldots]\Big)$$

From this sequent, by repeatedly applying the $\exists$-right rule we obtain a polynomial size cut-free **G** proof of $\mathcal{S}[\vec{m};\vec{n}]$.

For the induction step, we consider all rules of $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$. In each case, assume that $\mathcal{S}$ is obtained from $\mathcal{S}_1$ (and $\mathcal{S}_2$). We will show that $\mathcal{S}[\ldots]$ has short $\mathbf{G}_1^\star$ derivation from $\mathcal{S}_1[\ldots]$ (and $\mathcal{S}_2[\ldots]$). It is obvious that the polytime function $F(\ldots)$ giving the $\mathbf{G}_1^\star$ proof of $\mathcal{S}[\ldots]$ can be constructed from the polytime function(s) $F_1(\ldots)$ for $\mathcal{S}_1$ (and $F_2(\ldots)$ for $\mathcal{S}_2$).

All rules (including the **IND** rule) except for the string quantifier rules are treated just as in the proof of the Translation Theorem for $\mathbf{V}^0$ (page 162), although now the translation will require cuts on Σ_i^q formulas in general. We consider the string $\exists$-introduction rules. The string $\forall$-introduction rules are dual, and are left as an exercise.

Case string $\exists$-right: Suppose that $\mathcal{S}$ is obtained from $\mathcal{S}_1$ by the string $\exists$-right rule. Note that in $\tilde{\mathbf{V}}^1$, the only string terms are string variables.

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda(\gamma) \longrightarrow \Pi(\gamma), |\gamma| \leq t \wedge \psi(\gamma)}{\Lambda(\gamma) \longrightarrow \Pi(\gamma), \exists Z \leq t\, \psi(Z)}$$

We suppress all free variables except for the principle variable γ. Note that $|\gamma| \le t[\dots,n]$ is either $\top$ or $\bot$. Let $r = val(t)$, then

$$(150)\quad \mathcal{S}_1[\dots,n] =_{\text{def}} \begin{cases} \Lambda[\dots,n] \longrightarrow \Pi[\dots,n], \psi(\gamma)[\dots,n] & \text{if } n \le r \\ \Lambda[\dots,n] \longrightarrow \Pi[\dots,n], \bot & \text{if } n > r \end{cases}$$

By definition (see (148)),

$$\mathcal{S}[\dots,n] =_{\text{def}} \qquad \Lambda[\dots,n] \longrightarrow \Pi[\dots,n], \exists p_0^Z \dots \exists p_{r-2}^Z \bigvee_{k=0}^{r} \psi(Z)[\dots,k]$$

Consider the interesting case where $n \le r$, First, by repeated applications of the rules weakening and $\vee$-right, we obtain from $\mathcal{S}_1[\dots,n]$

$$\Lambda[\dots,n] \longrightarrow \Pi[\dots,n], \bigvee_{k=0}^{r} \psi(\gamma)[\dots,k]$$

Then we can derive $\mathcal{S}[\dots,n]$ using the rule $\exists$-right.

Case string $\exists$-left: Again, suppressing all other free variables:

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{|\gamma| \le t \wedge \psi(\gamma), \Lambda \longrightarrow \Pi}{\exists Z \le t\, \psi(Z), \Lambda \longrightarrow \Pi}$$

where γ does not occur in $\mathcal{S}$, and ψ is Σ_1^B. Let $r = val(t)$, then for $n \le r$,

$$(151)\qquad \mathcal{S}_1[\dots,n] =_{\text{def}} \qquad \psi(\gamma)[\dots,n], \Lambda[\dots] \longrightarrow \Pi[\dots]$$

Also,

$$\mathcal{S}[\dots] =_{\text{def}} \qquad \exists p_0^Z \dots \exists p_{r-2}^Z \bigvee_{n=0}^{r} \psi(Z)[\dots,n],\ \Lambda[\dots] \longrightarrow \Pi[\dots]$$

Now if $r = 0$, then we are done. Otherwise, combine the sequents $\mathcal{S}_1[\dots,n]$ for $n = 0,\dots,r$ by the rule $\vee$-left we obtain

$$\bigvee_{n=0}^{r} \psi(\gamma)[\dots,n],\ \Lambda[\dots] \longrightarrow \Pi[\dots]$$

Thus we get $\mathcal{S}[\dots]$ by $r-1$ applications of the $\exists$-left rule. $\dashv$

EXERCISE 7.59. *Carry out the cases for the string $\forall$-introduction rules.*

7E.1. Translating $\mathbf{V}^0$ to bounded depth $\mathbf{G}_0^\star$. In Section 7B we show that Σ_0^B theorems of $\mathbf{V}^0$ translate into families of tautologies with polynomial-size bounded depth **PK** proofs. We generalize this and show here that the translation of every *bounded* theorem of $\mathbf{V}^0$ has polynomial-size proofs in a subsystem of $\mathbf{G}_0^\star$ that extends **bPK**. First we define the system.

DEFINITION 7.60 (Bounded Depth $\mathbf{G}_0$). *For each constant $d \in \mathbb{N}$, a d-$\mathbf{G}_0$ proof is a $\mathbf{G}$ proof in which all target formulas have depth at most d and all cut formulas are quantifier-free and also have depth at most d. A* bounded depth $\mathbf{G}_0$ *system (or just $\mathbf{bG}_0$) is any system d-$\mathbf{G}_0$ for $d \in \mathbb{N}$. Treelike d-$\mathbf{G}_0$ (resp. treelike $\mathbf{bG}_0$) is denoted by d-$\mathbf{G}_0^\star$ (resp. $\mathbf{bG}_0^\star$).*

Theorem 7.20 is generalized as follows:

THEOREM 7.61. *For any bounded theorem $\varphi(\vec{x}, \vec{X})$ of $\mathbf{V}^0$ there is a constant d and a polytime function $F(\vec{m}, \vec{n})$ such that $F(\vec{m}, \vec{n})$ is a d-$\mathbf{G}_0^\star$-proof of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$, for all $\vec{m}, \vec{n} \in \mathbb{N}$.*

We prove the theorem by translating $\mathbf{LK}^2$-$\mathbf{V}^0$ proofs (as opposed to the $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ proofs used in the proof of Theorem 7.20). An $\mathbf{LK}^2$-$\mathbf{V}^0$ proof can have cut formulas which are Σ_1^B; these are instances of the Σ_0^B-**COMP** axioms. Because in the translation we are not allowed to cut quantified formulas, these instances of Σ_0^B-**COMP** will require different translations than the translation described before Proposition 7.56.

The main idea is as follows. Consider an instance of Σ_0^B-**COMP**:

$$\exists X \le t \forall i < t\big(X(i) \leftrightarrow \psi(i)\big)$$

Instead of introducing quantified Boolean variables p_i^X for the bits $X(i)$ of X we will translate $X(i)$ using the translation of $\psi(i)$. For the string eigenvariable γ that introduces X (in a string $\exists$-left rule) we also translate $\gamma(i)$ using the translation of $\psi(i)$. Now ψ may contain other eigenvariables, so they must be translated first.

Recall the notions of anchored proofs (Definition 2.12 on page 12) and free variable normal form (Section 2B.4 on page 21 and Section 4D.1 on page 88).

PROOF. Since φ is a theorem of $\mathbf{V}^0$, there is an anchored $\mathbf{LK}^2$-$\mathbf{V}^0$ proof π of φ. We can assume that π is in free variable normal form and is treelike. Note that all cut formulas in π are Σ_1^B, and all non-Σ_0^B cut formulas are instances of Σ_0^B-**COMP** axioms. Here we are only interested in the instances of Σ_0^B-**COMP** that will be cut. Furthermore, we can assume that all sequents that contain an instance of Σ_0^B-**COMP** in the succedent are derived from the Σ_0^B-**COMP** axiom by weakenings:

$$\text{(152)} \qquad \frac{\longrightarrow \exists X \forall x < t(X(x) \leftrightarrow \psi(x))}{\Gamma \longrightarrow \exists X \forall x < t(X(x) \leftrightarrow \psi(x)), \Delta}$$

Consider an application of the string $\exists$-left rule that introduces a Σ_0^B-**COMP** formula:

$$\text{(153)} \qquad \frac{\mathcal{S}_1}{\mathcal{S}_2} = \frac{|\gamma| \le t \wedge \forall x < t(\gamma(x) \leftrightarrow \psi(x)), \Gamma \longrightarrow \Delta}{\exists X \le t \forall x < t(X(x) \leftrightarrow \psi(x)), \Gamma \longrightarrow \Delta}$$

where γ does not occur in $\mathcal{S}_2$. Since π is in free normal variable form, each variable γ is used exactly once.

Notation. We say that γ as above is a *comprehension variable in* π. The associated pair $\langle t, \psi\rangle$ as above is called the *defining pair* of γ.

The idea is to translate the bit $\gamma(i)$ of any comprehension variable γ in π using its defining pair (instead of using new atoms $p_0^\gamma, p_1^\gamma, \dots$ as before). Note that two different comprehension variables may have the same defining pair (for example, comprehension variables that introduce two identical copies of a Σ_0^B-**COMP** cut formula which are merged in contraction right or in the branching rules such as $\vee$ left or $\wedge$ right). In this case they will have the same translation.

The defining pair of γ may contain other comprehension variables. For example, in (153) $\psi(x)$ might contain a comprehension variable γ', where its corresponding ψ' is in Γ. In this case γ' must be translated before γ. This motivates the following notions.

Notation. We say that a comprehension variable γ *depends on* a variable β (or b) if β (resp. b) occurs in the subproof of π that ends in a string $\exists$-left that removes γ as in (153) above.

Since π is treelike and in free variable normal form, this dependence relation forms a partial ordering of the comprehension variables.

Notation. The *dependence degrees* of variables in π are defined as follows. All non-comprehension variables have dependence degree 0. The dependence degree of a comprehension variable γ is one plus the maximum dependence degree of all variables occurring in its defining pair.

Translation of formulas in π: The formulas in π are translated in stages as follows. In stage 0 we translate all formulas that do not involve any comprehension variables. Generally, in stage i we translate all formulas that involve some variables of dependence degree i but no variable of higher dependence degree. In each stage, the translation is by induction on the depth of the formulas. Stage 0 is the same as described at the beginning of Section 7E.

Consider stage $(i+1)$ (where $i \geq 0$). For the base case, all atomic formulas have been translated in the previous stage except for atomic formulas of the form $\gamma(s)$, where γ has dependence degree $(i+1)$. For each such γ, let

$$\gamma(s)[n_\gamma, \vec{n}] =_{\mathrm{def}} \begin{cases} \psi(s)[\vec{n}] & \text{if } i < n_\gamma - 1 \\ \top & \text{if } i = n_\gamma - 1 \\ \bot & \text{if } i \geq n_\gamma \end{cases} \tag{154}$$

where $\langle t, \psi\rangle$ is the defining pair for γ, and $i = val(s(\underline{\vec{n}}))$ (recall val from page 158).

The induction step is handled as discussed at the beginning of Section 7E except for the case of the Σ_0^B-**COMP** cut formulas. Intuitively these formulas are true, so they should translate into tautologies. In this case we will show that the tautologies have polynomial size d'-$\mathbf{PK}^\star$ proofs for some d'. Therefore we will simply delete all sequents on the right branches of the cut Σ_0^B-**COMP** rule (these are ancestors of a sequent that contains the cut Σ_0^B-**COMP** formula in its succedent). Also, we will translate all occurrences of the cut Σ_0^B-**COMP** formulas in the antecedents into the empty formula. This completes the description of our translation.

We leave as an exercise to verify that the translation formulas have polynomial sizes and constant depths as desired.

EXERCISE 7.62. *Show that for each Σ_0^B formula $\psi(\vec{x}, \vec{X})$ in π there is a constant d_1 and a polynomial $\mathbf{p}$ that depend on π such that $\psi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ have depth at most d_1 and size $\mathbf{p}(\vec{m}, \vec{n})$, for all $\vec{m}$, $\vec{n}$. Show that Proposition 7.56 continues to hold for non-cut formulas in π.*

Now we show that for all sequents $\mathcal{S}$ of π that are not on the right branches of a cut Σ_0^B-**COMP** rule, the families $\mathcal{S}[\vec{m}; \vec{n}]$ have polynomial size d-$\mathbf{G}_0^\star$ proofs, for some constant d.

The base case, where $\mathcal{S}$ is a nonlogical axiom in 2-**BASIC** (Figure 2), is handled just as in Section 7B.3, with obvious modifications when a free string variable in the axiom is a comprehension variable. The induction step is the same as in the proof of Theorem 7.57 except for the case where $\mathcal{S} = \mathcal{S}_2$ as in (153), i.e., where it is obtained by the string $\exists$-left that introduces a Σ_0^B-**COMP** cut formula.

So suppose that $\mathcal{S}_2$ is obtained from $\mathcal{S}_1$ as in (153). Note that for $n_\gamma \le v$ (where $v = val(t)$)

$$\mathcal{S}_1[\vec{m}; n_\gamma, \vec{n}] =_{\text{def}} \qquad C[\vec{m}; n_\gamma, \vec{n}], \Gamma[\vec{m}; \vec{n}] \longrightarrow \Delta[\vec{m}; \vec{n}]$$

where $C[\dots]$ translates the first formula of $\mathcal{S}_1$. Let $v = val(t)$. By definition:

$$C[\vec{m}; 0, \vec{n}] = \bigwedge_{i=0}^{v-1} \neg\psi(x)[\vec{m}, i; \vec{n}] \qquad (i \text{ is the value of } x)$$

and for $1 \le n_\gamma \le v$ (here let A_i denotes $\psi(x)[\vec{m}, i; \vec{n}]$):

$$C[\vec{m}; n_\gamma, \vec{n}] = \big(\bigwedge_{i=0}^{n_\gamma - 2} (A_i \leftrightarrow A_i) \big) \wedge A_{n_\gamma - 1} \wedge \bigwedge_{i=n_\gamma}^{v-1} \neg A_i$$

Here the conjuncts

$$\bigwedge_{i=0}^{n_\gamma - 2} (A_i \leftrightarrow A_i) \qquad \text{and} \qquad \bigwedge_{i=n_\gamma}^{v-1} \neg A_i$$

are deleted when their sets of indices are empty. Also, by definition

$$\mathcal{S}_2[\vec{m}; \vec{n}] =_{\text{def}} \qquad \Gamma[\vec{m}; \vec{n}] \longrightarrow \Delta[\vec{m}; \vec{n}]$$

Using Exercise 7.58, we can show (by the same arguments as in the proof of Theorem 7.57 below Exercise 7.58) that there are polynomial size cut-free **PK** proofs of the tautologies:

$$\longrightarrow \bigvee_{n_\gamma=0}^{v} C[\vec{m}; n_\gamma, \vec{n}] \tag{155}$$

Moreover, by Exercise 7.62 the above tautologies have depth at most d_1 for some d_1 depending only on π. Therefore the proof of Theorem 7.8 shows that (155) have polynomial-size d_2-$\mathbf{PK}^\star$ proofs, where $d_2 = d_1 + 3$. Hence, by using the $\vee$-left rule for the sequents $\mathcal{S}_1[\vec{m}; n_\gamma, \vec{n}]$ (for $0 \leq n_\gamma \leq v$) and then applying a cut for the resulting sequent with (155) we obtain $\mathcal{S}_2[\vec{m}; \vec{n}]$.

All cut formulas in our translation are either cut formulas in the d_2-$\mathbf{G}_0^\star$ derivations mentioned above, or the translations of Σ_0^B cut formulas in π. Thus they have depth bounded by a constant depending on π. Furthermore, it can be seen that all target formulas are atomic formulas of the form

$$p_i^\alpha$$

for some noncomprehension string variable α. As a result, the translations of π are d-$\mathbf{G}_0^\star$ proofs for some constant d depending on π. ⊣

EXERCISE 7.63. *Reprove Theorem 7.20 using the translation we describe in the proof of Theorem 7.61.*

7F. Notes

Definitions 7.2, 7.5 and Theorem 7.4 are from [34]. Also, the fact that **Frege** proof systems are p-equivalent is proved in [34].

The first propositional translation of an arithmetic theory is described in [28]. The translation of Σ_0^B formulas given in Subsection 7B.1 is from [30], and both this and the $\mathbf{V}^0$ Translation Theorem 7.20 are based on the treatment of $\mathbf{I}\Delta_0(R)$ by Paris and Wilkie [68].

A proof system for the Quantified Propositional Calculus was introduced by Dowd [37]. The system $\mathbf{G}$ and its subsystems $\mathbf{G}_i$ were introduced by Krajíček and Pudlák [56] (see also Section 4.6 of [55]). The original definition of $\mathbf{G}$ is what we refer to as **KPG** in Exercise 7.35 and the original definition of $\mathbf{G}_i$ is **KPG** restricted so that all formulas must be either Σ_i^q or Π_i^q. Our definitions are due to Morioka [61]. Theorem 7.37 is new. Theorem 7.46 is from [70].

The idea of $\mathbf{G}_i^\star$ (treelike $\mathbf{G}_i$) is from [55], and the $\mathbf{V}^1$ Translation Theorem 7.57 is adapted from a similar theorem for $\mathbf{S}_2^1$ also in [55]. Theorem 7.51 is from [29]. Extended Frege proof systems, which inspired the system **ePK** in Section 7D.1, were introduced in [34].

Theorem 7.61 is new.

Chapter 8

THEORIES FOR POLYNOMIAL TIME AND BEYOND

We present a finitely-axiomatizable "minimal" theory for polynomial time over the basic two-sorted vocabulary $\mathcal{L}_A^2$. We show that it is robust by giving three quite different axiomatizations for it under the names $\mathbf{VP}$, $\mathbf{TV}^0$, and $\mathbf{V}^1$-$\mathbf{HORN}$. We also present a universal conservative extension $\mathbf{VPV}$ for this theory which has function symbols for all polynomial time functions based on Cobham's recursion-theoretic characterization of $\mathbf{FP}$. The theory $\mathbf{V}^1$ from Chapter 6 has the same Σ_1^B theorems as the minimal theory, but apparently has more Σ_2^B theorems. The new theories have the following inclusions:

$$\mathbf{VP} = \mathbf{TV}^0 = \mathbf{V}^1\text{-}\mathbf{HORN} \subset_{cons} \widehat{\mathbf{VP}} \subset_{cons} \mathbf{VPV}$$

where $\mathcal{T}_1 \subset_{cons} \mathcal{T}_2$ means that $\mathcal{T}_2$ is a conservative extension of $\mathcal{T}_1$.

Section 8C introduces the $\mathbf{TV}^i$ hierarchy and concentrates on the bottom level $\mathbf{TV}^0$ mentions above. Section 8E is devoted to $\mathbf{TV}^1$, and characterizes the Σ_1^B-definable search problems in this theory as those reducible to polynomial local search. Section 8F proves a form of the Herbrand Theorem known as the KPT Witnessing Theorem, which can be used to prove (or suggest) independence results for Σ_2^B-formulas. As an application we show that $\mathbf{V}^0$ does not prove the Σ_0^B-Replacement scheme, and (unless integer factoring is easy) neither does $\mathbf{VPV}$.

Section 8G proves a host of results on $\mathbf{V}^\infty$, the interleaved $\mathbf{V}^i$ and $\mathbf{TV}^i$ hierarchies. These include the finite axiomatizability of $\mathbf{V}^i$ and $\mathbf{TV}^i$, Σ_i^B-definability results (see Table 1 page 239 for a summary), and the equivalence of the collapse of these hierarchies and the provable collapse of the polynomial hierarchy. Section 8H proves 'RSUV' isomorphism theorems relating our two-sorted theories $\mathbf{V}^i$ and $\mathbf{TV}^i$ to Buss's single-sorted theories $\mathbf{S}_2^i$ and $\mathbf{T}_2^i$.

8A. The Theory VP and Aggregate Functions

The theory $\mathbf{VP}$ extends $\mathbf{V}^0$ by adding a single axiom asserting that the gates of a given monotone Boolean circuit with specified inputs can be evaluated. We will then use the fact that the Monotone Circuit Value problem is complete for **P** under many-one $\mathbf{AC}^0$ reductions to prove that all polynomial time functions are Σ_1^B definable in **VP**. We will show that $\mathbf{V}^1$ extends **VP**, and show that the Σ_1^B theorems of $\mathbf{V}^1$ and **VP** are the same. Later, in Section 8F, we give evidence that **VP** does not prove either the Σ_0^B-**REPL** scheme or the Σ_1^B-**COMP** scheme (which do not consist of Σ_1^B formulas), and hence apparently $\mathbf{V}^1$ is not conservative over **VP**.

It seems that **VP** is a "minimal" theory for polynomial time reasoning because it extends our base theory $\mathbf{V}^0$ by adding one axiom asserting the existence of a solution to a standard complete problem for **P**. We use this same method in Chapter 9 to introduce minimal theories for other complexity classes.

We specify a monotone Boolean circuit (using our two-sorted language $\mathcal{L}_A^2$) by a triple (a, G, E), where the gates are numbered 0, 1, ..., $(a-1)$, and for $x > 1$, $G(x)$ holds iff gate x is an $\wedge$ gate (otherwise gate x is an $\vee$ gate). Gates numbered 0 and 1 are "input" gates, and always have the values 0 and 1 respectively. The edge relation E specifies the inputs to the other gates as follows:

- For $0 \le y < x, 2 \le x < a$, $E(y, x)$ holds iff the output of gate y is connected to an input of gate x.

The Σ_0^B formula $\delta_{MCV}(a, G, E, Y)$ asserts that $Y(x)$ holds iff the output of gate x is 1 (i.e. $\top$), and is defined as follows:

$$(156)\quad \delta_{MCV}(a,G,E,Y) \equiv \neg Y(0) \wedge Y(1) \wedge \forall x < a, 2 \le x \supset$$
$$Y(x) \leftrightarrow [(G(x) \wedge \forall y < x(E(y,x) \supset Y(y))) \vee (\neg G(x) \wedge \exists y < x(E(y,x) \wedge Y(y)))]$$

DEFINITION 8.1 (**VP**). *The theory* **VP** *has vocabulary* $\mathcal{L}_A^2$ *and is axiomatized by the axioms of* $\mathbf{V}^0$ *and one more axiom called* MCV*, where*

$$MCV \equiv \exists Y \le a\ \delta_{MCV}(a, G, E, Y)$$

The next result is immediate from the above definition and the fact that $\mathbf{V}^0$ is finitely axiomatizable (Theorem 5.76).

COROLLARY 8.2. **VP** *is finitely axiomatizable.*

THEOREM 8.3. $\mathbf{V}^1$ *is an extension of* **VP**.

PROOF. It suffices to show that $\mathbf{V}^1$ proves the axiom MCV. But MCV is a Σ_1^B-formula, and is easily proved by induction on a. ⊣

Note that MCV is a bounded formula, and hence **VP** is a polynomial-bounded theory (Definition 5.23). Thus by Parikh's Theorem 5.24 a function is provably total (i.e. Σ_1^1-definable) in **VP** iff it is Σ_1^B-definable in **VP**.

THEOREM 8.4. *A function is provably total in* **VP** *iff it is in* **FP**.

One direction is proved as Theorem 8.8 below, and the other direction is Corollary 8.14.

We introduce a string function F_{MCV} which witnesses the existential quantifier in the axiom MCV. The defining axiom is

$$Y = F_{MCV}(a, G, E) \leftrightarrow |Y| \leq a \wedge \delta_{MCV}(a, G, E, Y) \tag{157}$$

Lemma 8.5. *F_{MCV} is Σ_0^B definable in* **VP**.

Proof. We need to show that **VP** proves

$$\exists! Y,\ |Y| \leq a \wedge \delta_{MCV}(a, G, E, Y)$$

Existence of Y follows from the axiom MCV. Uniqueness can be proved in $\mathbf{V}^0$ by induction on i using the Σ_0^B formula $\psi(i)$ asserting that the first i bits of Y are uniquely determined. ⊣

We define the two-sorted Monotone Circuit Value problem using the relation $R_{MCV}(a, G, E)$, which holds iff the circuit specified by (a, G, E) has output 1 (gate number $a \dot{-} 1$ is designated the output).

Definition 8.6.

$$R_{MCV}(a, G, E) \leftrightarrow \exists Y \leq a,\ \delta_{MCV}(a, G, E, Y) \wedge Y(a \dot{-} 1) \tag{158}$$

The following proposition shows that R_{MCV} is $\mathbf{AC}^0$-many-one complete for **P**.

Proposition 8.7. *For any relation $R(\vec{x}, \vec{X})$ in* **P** *there are functions a_0, G_0, E_0 in $\mathbf{FAC}^0$ such that*

$$R(\vec{x}, \vec{X}) \leftrightarrow R_{MCV}(a_0(\vec{x}, \vec{X}), G_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X})) \tag{159}$$

Proof sketch. First we point out that Circuit Value Problem CVP for Boolean circuits which have $\neg$ gates in addition to $\wedge$ and $\vee$ gates is easily reduced to the Monotone Circuit Value Problem MCV by using the method of "double-rail logic". Given a circuit C which has gates for $\wedge, \vee, \neg$ we compute (in $\mathbf{FAC}^0$) a monotone circuit C' which has two inputs x and x' for each input x of C, and two gates g' and g'' for every gate g in C. This is done such that, assuming that each input x' is the negation of x, then $g' \leftrightarrow g$ and $g'' \leftrightarrow \neg g$. Given an assignment of inputs to C, suitable inputs to C' satisfying $x' \leftrightarrow \neg x$ can trivially be computed by an $\mathbf{FAC}^0$ function. To design C', note that the gate g has one of the three types $\wedge, \vee, \neg$, and in each case (by De Morgan's laws) there are easy monotone circuits which compute both g' and g'' from the inputs to g and their negations.

Now to prove the Proposition it suffices to show that, given a polytime Turing machine M for computing a relation $R(\vec{x}, \vec{X})$, there is an $\mathbf{AC}^0$ function F_{M} such that $F_{\mathsf{M}}(\vec{x}, \vec{X})$ describes a circuit (allowing $\neg$ gates and with given input values) whose gate values describe the computation of M on input $\vec{x}, \vec{X}$.

One way to see how to do this is to consider equation (94) (page 132), where the variable Z describes the computation of a polytime Turing machine. Here the rows $Z^{[z]}$ of Z are computed successively using the $\mathbf{AC}^0$ functions $Init_{\mathsf{M}}$ and $Next_{\mathsf{M}}$. All $\mathbf{AC}^0$ functions are computed by uniform circuit families, which themselves are describable by $\mathbf{AC}^0$ functions. ⊣

THEOREM 8.8. *Every function in* $\mathbf{FP}$ *is* Σ_1^B*-definable in* $\mathbf{VP}$.

PROOF. It suffices to prove this for string functions, since by Proposition 6.5 every number function $f(\vec{x}, \vec{X})$ in $\mathbf{FP}$ has the form $|F(\vec{x}, \vec{X})|$ for some string function F in $\mathbf{FP}$, and by Exercise 5.30 the Σ_1^B definable functions in $\mathbf{VP}$ are closed under composition.

By Definition 5.15, a string function $F(\vec{x}, \vec{X})$ is in $\mathbf{FP}$ iff it is p-bounded and its bit-graph is in $\mathbf{P}$; i.e. there is an $\mathcal{L}_A^2$ term $t(\vec{x}, \vec{X})$ and a relation $B_F(i, \vec{x}, \vec{X})$ in $\mathbf{P}$ such that

$$F(\vec{x}, \vec{X})(i) \leftrightarrow \big(i < t(\vec{x}, \vec{X}) \wedge B_F(i, \vec{x}, \vec{X})\big) \tag{160}$$

Our task is to find a Σ_1^B formula $\varphi_F(\vec{x}, \vec{X}, Z)$ representing the graph of F by satisfying

$$Z = F(\vec{x}, \vec{X}) \leftrightarrow \varphi_F(\vec{x}, \vec{X}, Z)$$

and such that

$$\mathbf{VP} \vdash \exists! Z \varphi_F(\vec{x}, \vec{X}, Z) \tag{161}$$

Since the bit graph B_F of F is a polytime relation, by (158), (159) there are functions a_0, G_0, E_0 in $\mathcal{L}_{\mathbf{FAC}^0}$ such that

(162)

$$B_F(i, \vec{x}, \vec{X}) \leftrightarrow \exists Y \le a_0(i),\ \delta_{MCV}(a_0(i), G_0(i), E_0(i), Y) \wedge Y(a_0(i) \dot{-} 1)$$

where we have suppressed the arguments $(\vec{x}, \vec{X})$ in a_0, G_0, E_0. We can use the function F_{MCV} defined in (157) to witness Y in the above equation, and hence the graph φ_F of F satisfies

(163)

$$\varphi_F(\vec{x}, \vec{X}, Z) \ \leftrightarrow\ \forall i < t[Z(i) \leftrightarrow F_{MCV}(a_0(i), G_0(i), E_0(i))(a_0(i) \dot{-} 1)]$$

Unfortunately the formula on the right is not Σ_1^B, and although the part in brackets $[\dots]$ can be made Σ_1^B, the existential string quantifier there requires the Replacement Axiom (Definition 6.18) to move it in front of the quantifier $\forall i < t$. In Section 8F we give evidence that $\mathbf{VP}$ does not prove Σ_0^B-$\mathbf{REPL}$.

So we use another approach. From (162) we see that for each fixed $i, 0 \le i < t$, the parameters $(a_0(i), G_0(i), E_0(i))$ describe a circuit $C(i)$ which computes bit i of $F(\vec{x}, \vec{X})$. Our task is to describe one circuit $C = C(\vec{x}, \vec{X})$ which combines the circuits $C(0), \dots, C(t-1)$ to compute all of these bits together.

In order to do this it will be helpful to introduce the important notion of the aggregate function F^*_{MCV} of F_{MCV}, where the aggregate F^* of F is the string function that gathers the values of F for a polynomially long sequence of arguments. We use the notation $Z^{[x]} = Row(x, Z)$ (Definition 5.51) and $(Z)^x = seq(x, Z)$ (Definition 5.56)).

Definition 8.9 (Aggregate Function). *Suppose that*

$$F(x_1, \dots, x_k, X_1, \dots, X_n)$$

is a polynomially bounded string function, i.e., for some $\mathcal{L}^2_A$ term t,

$$|F(\vec{x}, \vec{X})| \leq t(\vec{x}, \vec{X})$$

Then $F^\star(b, Z_1, \dots, Z_k, X_1, \dots, X_n)$ is the polynomially bounded string function that satisfies

$$|F^\star(b, \vec{Z}, \vec{X})| \leq \langle b, t(\vec{Z}, \vec{X})\rangle$$

and

$$F^\star(b, \vec{Z}, \vec{X})(w) \leftrightarrow \exists i < b \exists v < w,\ w = \langle i, v\rangle \wedge F((Z_1)^i, \dots, (Z_k)^i, X_1^{[i]}, \dots, X_n^{[i]})(v) \tag{164}$$

Notice that by (164)

$$\forall i < b,\ F^\star(b, \vec{Z}, \vec{X})^{[i]} = F((Z_1)^i, \dots, (Z_k)^i, X_1^{[i]}, \dots, X_n^{[i]}) \tag{165}$$

The use of *seq* in (164) and (165) can be eliminated using its definition 5.56 to obtain an equivalent $\Sigma^B_0(Row, F)$ definition of the bit graph of $F^\star$, but in general the use of *Row* cannot be eliminated to get a $\Sigma^B_0(F)$ definition.

Lemma 8.10 below shows that the aggregate function F^*_{MCV} of F_{MCV} is Σ^B_1-definable in **VP**. We can interpret F^*_{MCV} as assigning values to the gates of a collection $C(0), \dots C(b-1)$ of circuits. Thus (165) becomes

$$F^*_{MCV}(b, Z, U, V)^{[i]} = F_{MCV}((Z)^i, U^{[i]}, V^{[i]}) \tag{166}$$

and writing

$$((Z)^i, U^{[i]}, V^{[i]}) = (a_i, G_i, E_i) \tag{167}$$

we want the the triple (a_i, G_i, E_i) to describe the circuit $C(i)$ which computes bit i of $F(\vec{x}, \vec{X})$. Thus by (163) we want

$$((Z)^i, U^{[i]}, V^{[i]}) = (a_0(i), G_0(i), E_0(i))$$

For this we define "pseudo-aggregate" functions A_1, G_1, E_1 for the functions a_0, G_0, E_0 which, for fixed $\vec{x}, \vec{X}$, collect values for arguments $i < t(\vec{x}, \vec{X})$. Thus for $i < t$

$$\begin{aligned} (A_1(\vec{x}, \vec{X}))^i &= a_0(i, \vec{x}, \vec{X}) \\ G_1(\vec{x}, \vec{X})^{[i]} &= G_0(i, \vec{x}, \vec{X}) \\ E_1(\vec{x}, \vec{X})^{[i]} &= E_0(i, \vec{x}, \vec{X}) \end{aligned}$$

Since each of the functions a_0, G_0, E_0 is in $\mathbf{FAC}^0$, it follows easily from the $\mathbf{FAC}^0$ Elimination Lemma 5.74 that the functions A_1, G_1, E_1 have Σ_0^B-definable bit graphs and hence are themselves in $\mathbf{FAC}^0$.

If $Y = F^*_{MCV}(t, A_1, G_1, E_1)$ (where we have suppressed the arguments $\vec{x}, \vec{X}$) then $Y^{[i]}$ gives the correct assignment to the gates of $C(i)$. Thus for each $i < t$

$$F(\vec{x}, \vec{X})(i) \leftrightarrow Y^{[i]}(a_0(i, \vec{x}, \vec{X}) \dot{-} 1)$$

So we define the $\mathbf{FAC}^0$ function *Extract* by defining its bit graph as follows:

$$\mathit{Extract}(\vec{x}, \vec{X}, Y)(i) \leftrightarrow \big(i < t(\vec{x}, \vec{X}) \wedge Y^{[i]}(a_0(i, \vec{x}, \vec{X}) \dot{-} 1)\big)$$

Then

$$F(\vec{x}, \vec{X}) = \mathit{Extract}(\vec{x}, \vec{X}, F^*_{MCV}(t, A_1, G_1, E_1)) \tag{168}$$

(again suppressing some occurrences of $\vec{x}, \vec{X}$). This, Lemma 8.10 and the fact that the Σ_1^B-definable functions in a polynomial-bounded theory are closed under composition show that F is Σ_1^B-definable in $\mathbf{VP}$. ⊣

To complete the proof of Theorem 8.8 we need the following result.

LEMMA 8.10. *F^*_{MCV} is Σ_1^B-definable in $\mathbf{VP}$, and $\mathbf{VP}(F_{MCV}, F^*_{MCV})$ proves* (166).

PROOF. For $i < b$ let $C(i)$ be the circuit described by (a_i, G_i, E_i) as in (167). We want to embed the circuits $C(0), C(1), \ldots, C(b-1)$ into a single circuit C. Each $C(i)$ has $a_i < |Z|$ gates, and we will be generous and a lot $|Z|$ gates in the embedded version of each $C(i)$, so that C has a total of $b|Z|$ gates. Thus gate j of $C(i)$ corresponds to gate $i|Z| + j$ of C.

Circuit C has the description $(\hat{a}, \hat{G}, \hat{E})$, where $\hat{a} = b|Z|$ and $\hat{G} = \hat{G}(b, Z, U)$ and $\hat{E} = \hat{E}(b, Z, V)$ are $\mathbf{FAC}^0$ functions. These functions are straightforward to define to satisfy the intended embedding of $C(i)$ into C, except that the gates in C corresponding to gates 0 and 1 of $C(i)$ must have constant values 0 and 1 respectively. To achieve this, these gates have no input edges and we make them OR gates and AND gates respectively. Thus for $i, i' < b$

$$\hat{G}(b, Z, U)(i|Z| + j) \leftrightarrow (U^{[i]}(j) \wedge 2 \le j) \vee j = 1 \qquad \text{if } j < |Z|$$
$$\hat{E}(b, Z, V)(i|Z| + j, i'|Z| + k) \leftrightarrow V^{[i]}(j, k) \wedge i = i' \wedge 2 \le k \qquad \text{if } j, k < |Z|$$

This is easily turned into Σ_0^B-definitions of the bit graphs of $\hat{G}$ and $\hat{E}$.

Referring to (166), it remains to define an $\mathbf{FAC}^0$ function *Compile*(b, Z, Y) whose i-th row assigns correct values to the gates of $C(i)$, assuming that Y assigns correct values to the gates of C. Thus

$$\mathit{Compile}(b, Z, Y)(i, j) \leftrightarrow \big(i < b \wedge j < (Z)^i \wedge Y(i|Z| + j)\big)$$

Finally

(169)

$$F^*_{MCV}(b,Z,U,V) = \mathit{Compile}(F_{MCV}(b|Z|, \hat{G}(b,Z,U), \hat{E}(b,Z,V)))$$

and (166) is provable from this equation and the defining axioms for the functions involved. Also F^*_{MCV} is a composition of Σ^B_1-definable functions in **VP**, and hence is itself Σ^B_1-definable in **VP**. ⊣

To prove the converse to Theorem 8.8 we introduce a universal conservative extension of **VP** in the next subsection.

8A.1. The theory $\widehat{\mathbf{VP}}$. Let $\delta'_{MCV}(a,G,E,Y)$ denote a quantifier-free formula in the vocabulary $\mathcal{L}_{\mathbf{FAC}^0}$ which $\overline{\mathbf{V}}^0$ proves equivalent to $\delta_{MCV}(a,G,E,Y)$ (see Lemma 5.70). The function F'_{MCV} has defining axiom

$$Y = F'_{MCV}(a,G,E) \leftrightarrow |Y| \le a \wedge \delta'_{MCV}(a,G,E,Y) \tag{170}$$

Thus F_{MCV} (defined in (157)) and F'_{MCV} are equal as functions, although they have different defining axioms.

Definition 8.11 ($\widehat{\mathbf{VP}}$). *The universal theory $\widehat{\mathbf{VP}}$ has vocabulary*

$$\mathcal{L}_{\widehat{\mathbf{VP}}} = \mathcal{L}_{\mathbf{FAC}^0} \cup \{F'_{MCV}\}$$

and axioms those of $\overline{\mathbf{V}}^0$ together with the defining axiom (170) *for F'_{MCV}.*

Since F_{MCV} is in **FP** and every function in $\mathcal{L}_{\mathbf{FAC}^0}$ is in **FP**, it is clear that every term in the vocabulary of $\widehat{\mathbf{VP}}$ represents a function in **FP**. The next result states the converse.

Theorem 8.12. *Every function in* **FP** *is represented by a term in the vocabulary of $\widehat{\mathbf{VP}}$.*

Proof. Equation (169) expresses F^*_{MCV} as a term involving F_{MCV} and functions in $\mathcal{L}_{\mathbf{FAC}^0}$, and equation (168) (finishing the proof of Theorem 8.8) expresses an arbitrary string function F in **FP** as a term involving F^*_{MCV} and functions in $\mathcal{L}_{\mathbf{FAC}^0}$. By Proposition 6.5 every number function $f(\vec{x},\vec{X})$ in **FP** has the form $|F(\vec{x},\vec{X})|$ for some string function F in **FP**. ⊣

Theorem 8.13. *The theory $\widehat{\mathbf{VP}}$ is a universal conservative extension of* **VP**.

Proof. The formula $\delta_{MCV}(a,G,E,F'_{MCV}(a,G,E))$ is provable in $\widehat{\mathbf{VP}}$ by (170) and implies the axiom MCV for **VP**, and hence $\widehat{\mathbf{VP}}$ is an extension of **VP**.

$\mathbf{VP} + \overline{\mathbf{V}}^0$ is conservative over **VP** because $\overline{\mathbf{V}}^0$ is conservative over $\mathbf{V}^0$ (Theorem5.72). $\widehat{\mathbf{VP}}$ can be obtained from $\mathbf{VP} + \overline{\mathbf{V}}^0$ by adding the defining axiom for F'_{MCV}, and F'_{MCV} is definable in $\mathbf{VP} + \overline{\mathbf{V}}^0$ by

Lemma 8.5 (note that $\overline{\mathbf{V}}^0$ proves the equivalence of the defining axioms for F_{MCV} and F'_{MCV}). Thus by Theorem 5.27, $\widehat{\mathbf{VP}}$ is conservative over $\mathbf{VP} + \overline{\mathbf{V}}^0$ and hence over $\mathbf{VP}$. ⊣

COROLLARY 8.14. *Every function Σ_1^1-definable in* $\mathbf{VP}$ *or* $\widehat{\mathbf{VP}}$ *is in* $\mathbf{FP}$.

PROOF. As observed above every term of $\widehat{\mathbf{VP}}$ stands for a function in $\mathbf{FP}$. Since $\widehat{\mathbf{VP}}$ is a universal theory, it follows from the Herbrand Theorem that the existential quantifiers in any Σ_1^1 theorem of $\widehat{\mathbf{VP}}$ can be witnessed by a combination of terms and hence by functions in $\mathbf{FP}$ (see Section 5F.1 for this argument applied to $\overline{\mathbf{V}}^0$). Therefore every Σ_1^1-definable function in $\widehat{\mathbf{VP}}$ is in $\mathbf{FP}$. Since $\widehat{\mathbf{VP}}$ is an extension of $\mathbf{VP}$, the same is true of $\mathbf{VP}$. ⊣

We wish to show that $\widehat{\mathbf{VP}}$ proves the $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$-**IND** and $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$-**COMP** schemes. Note that Lemma 8.10 easily follows from $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$-**COMP**, and to prove this scheme we need a general result about aggregate functions which will also play an important role in Chapter 9.

THEOREM 8.15 (Aggregate Function). *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L}$ which extends $\mathbf{V}^0$(Row) and proves $\Sigma_0^B(\mathcal{L})$-**COMP**. Suppose that F and $F^\star$ are definable in $\mathcal{T}$ (Definition 5.26) and $\mathcal{T}(F, F^\star)$ proves* (165). *Then $\mathcal{T}(F)$ proves $\Sigma_0^B(\mathcal{L} \cup \{F\})$-**COMP**.*

PROOF. Since $\mathcal{T}$ proves $\Sigma_0^B(\mathcal{L})$-**COMP**, by Lemma 5.50 it proves the Multiple Comprehension axioms for $\Sigma_0^B(\mathcal{L})$.

Claim. For any $\mathcal{L}$-terms $\vec{s}, \vec{T}$ that contain variables $\vec{z}$, $\mathcal{T}(F)$ proves

$$\exists Y \forall z_1 < b_1 \dots \forall z_m < b_m,\ Y^{[\vec{z}]} = F(\vec{s}, \vec{T}) \tag{171}$$

where $Y^{[\vec{z}]}$ denotes $Y^{[\langle \vec{z} \rangle]}$.

PROOF OF THE CLAIM. Since $\mathcal{T}$ proves the Multiple Comprehension axiom scheme for $\Sigma_0^B(\mathcal{L})$ formulas, it proves the existence of $\vec{X}$ such that $X_j^{[\vec{z}]} = T_j$, for $1 \le j \le n$. It also proves the existence of Z_i such that $(Z_i)^{\langle \vec{z} \rangle} = s_i$, for $1 \le i \le k$. Now the value of Y that satisfies (171) is just $F^\star(\langle \vec{b} \rangle, \vec{Z}, \vec{X})$. ⊣

Let $\mathcal{L}' = \mathcal{L} \cup \{F\}$. We show by induction on the quantifier depth of a $\Sigma_0^B(\mathcal{L}')$ formula ψ that $\mathcal{T}(F)$ proves

$$\exists Z \le \langle b_1, \dots, b_m \rangle \forall z_1 < b_1 \dots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \psi(\vec{z}) \tag{172}$$

where $\vec{z}$ are all free number variables of ψ. It follows that $\mathcal{T}(F) \vdash \Sigma_0^B(\mathcal{L}')$-**COMP**.

For the base case, ψ is quantifier-free. The idea is to replace every occurrence of a term $F(\vec{s}, \vec{T})$ in ψ by a new string variable W which has the intended value of $F(\vec{s}, \vec{T})$. The resulting formula is $\Sigma_0^B(\mathcal{L})$, and we can apply the hypothesis.

Formally, suppose that $F(\vec{s}_1, \vec{T}_1), \dots, F(\vec{s}_k, \vec{T}_k)$ are all occurrences of F in ψ. Note that the terms $\vec{s}_i, \vec{T}_i$ may contain $\vec{z}$ as well as nested occurrences of F. Assume further that these F-terms are ordered by depth so that $\vec{s}_1, \vec{T}_1$ do not contain F, and for $1 < i \le k$, any occurrence of F in $\vec{s}_i, \vec{T}_i$ must be of the form $F(\vec{s}_j, \vec{T}_j)$, for some $j < i$. We proceed to eliminate F from ψ by using its defining axiom.

Let $W_1, ..., W_k$ be new string variables. Let $\overrightarrow{s'_1} = \vec{s}_1$, $\overrightarrow{T'_1} = \vec{T}_1$, and for $2 \le i \le k$, $\overrightarrow{s'_i}$ and $\overrightarrow{T'_i}$ be obtained from $\vec{s}_i$ and $\vec{T}_i$ respectively by replacing every maximal occurrence of any $F(\vec{s}_j, \vec{T}_j)$, for $j < i$, by $W_j^{[\vec{z}]}$. Thus F does not occur in any $\overrightarrow{s'_i}$ and $\overrightarrow{T'_i}$, but for $i \ge 2$, $\overrightarrow{s'_i}$ and $\overrightarrow{T'_i}$ may contain $W_1, \dots, W_{i-1}$.

By the Claim above, for $1 \le i \le k$, $\mathcal{T}(F)$ proves the existence of W_i such that

$$\forall z_1 < b_1 \dots \forall z_m < b_m,\ W_i^{[\vec{z}]} = F(\overrightarrow{s'_i}, \overrightarrow{T'_i}) \tag{173}$$

Let $\psi'(\vec{z}, W_1, \dots, W_k)$ be obtained from $\psi(\vec{z})$ by replacing each maximal occurrence of $F(\vec{s}_i, \vec{T}_i)$ by $W_i^{[\vec{z}]}$, for $1 \le i \le k$. Then

$$\mathcal{T} \vdash \exists Z \le \langle b_1, \dots, b_m \rangle \forall z_1 < b_1 \dots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \psi'(\vec{z}, W_1, \dots, W_k)$$

Such Z satisfies (172) when each W_i is defined by (173).

The induction step is straightforward. Consider for example the case $\psi(\vec{z}) \equiv \forall x < t \lambda(\vec{z}, x)$. By the induction hypothesis,

$$\mathcal{T}(F) \vdash \exists Z' \forall z_1 < b_1 \dots \forall z_m < b_m \forall x < t,\ Z'(\vec{z}, x) \leftrightarrow \lambda(\vec{z}, x).$$

Now

$$\mathbf{V}^0 \vdash \exists Z \forall z_1 < b_1 \dots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \forall x < t Z'(\vec{z}, x)$$

and hence $\mathcal{T}(F) \vdash \exists Z \forall \vec{z} < \vec{b}\, Z(\vec{z}) \leftrightarrow \psi(\vec{z})$ ⊣

Corollary 8.16. *$\widehat{\mathbf{VP}}$ proves the $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$-**IND** and $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$-**COMP** axioms.*

Proof. In Theorem 8.15 take $\mathcal{T} = \mathbf{VP} \cup \overline{\mathbf{V}}^0$ and $F = F'_{MCV}$. Then $\mathcal{L} = \mathcal{L}_{\mathbf{FAC}^0}$ so $\mathcal{T}$ proves $\Sigma_0^B(\mathcal{L})$-**COMP** by Lemma 5.71. Also $\mathcal{T}$ proves the defining equations (157) and (170) for F_{MCV} and F'_{MCV} are equivalent, so by Lemmas 8.5 and 8.10 F and $F^\star$ are Σ_1^B-definable in $\mathcal{T}$. Thus the corollary follows from the theorem, since $\widehat{\mathbf{VP}} = \mathcal{T}(F)$ (and $\mathbf{V}^0$ proves Σ_0^B-**IND**). ⊣

Note that the formulas $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$ represent precisely the polynomial time relations, so Corollary 8.16 together with Theorem 8.4 suggest that $\widehat{\mathbf{VP}}$ (and hence **VP**) "capture" polynomial time reasoning. Also **VP** seems to be a minimal such theory (relative to the base theory $\mathbf{V}^0$), since surely polynomial time reasoning should be able to prove the basic axiom MCV, that the monotone circuit value problem is complete for

P. In the next sections we will also prove that **VP** is a robust theory, by giving several equivalent axiomatizations for it.

8B. The Theory VPV

The universal theory **VPV** is based on the single-sorted theory **PV** [28], which historically was the first theory designed to capture polynomial time reasoning. It is an extension of $\widehat{\mathbf{VP}}$, and (unlike $\widehat{\mathbf{VP}}$) it has a function symbol (and not just a term) for every string function in **FP**. We will show that **VPV** is a conservative extension of $\widehat{\mathbf{VP}}$. The vocabulary of **VPV** extends that of $\overline{\mathbf{V}}^0$, with additional function symbols introduced based on Cobham's characterization of **FP** (Theorem 6.16).

Following Definition 6.15, we can write the defining equations for a string function $F(y, \vec{x}, \vec{X})$ defined by limited recursion from $G(\vec{x}, \vec{X})$ and $H(y, \vec{x}, \vec{X}, Z)$ as

(174) $$F(0, \vec{x}, \vec{X}) = G(\vec{x}, \vec{X})$$

(175) $$F(y+1, \vec{x}, \vec{X}) = (H(y, \vec{x}, \vec{X}, F(y, \vec{x}, \vec{X})))^{<t(y,\vec{x},\vec{X})}$$

where the bounding term $t(y, \vec{x}, \vec{X})$ is in $\mathcal{L}_A^2$ and the notation $Z^{<y}$ refers to $Cut(y, Z)$ (the first y bits of Z).

For convenience we repeat the defining axiom (85) for the functions $F_{\varphi(z),t}$ introduced in Section 5F to define $\overline{\mathbf{V}}^0$.

(176) $$F_{\varphi(z),t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X})$$

Definition 8.17. *The vocabulary $\mathcal{L}_{\mathbf{FP}}$ is the smallest set that satisfies*

(1) $\mathcal{L}_{\mathbf{FAC}^0} \subseteq \mathcal{L}_{\mathbf{FP}}$.
(2) *For each open formula $\varphi(z, \vec{x}, \vec{X})$ over $\mathcal{L}_{\mathbf{FP}}$ and term $t = t(\vec{x}, \vec{X})$ of $\mathcal{L}_A^2$ there is a string function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FP}}$.*
(3) *For each triple G, H, t, where $G(\vec{x}, \vec{X})$ and $H(y, \vec{x}, \vec{X}, Z)$ are functions in $\mathcal{L}_{\mathbf{FP}}$ and $t = t(y, \vec{x}, \vec{X})$ is a term in $\mathcal{L}_A^2$, there is a function $F_{G,H,t}$ in $\mathcal{L}_{\mathbf{FP}}$ (with defining equations* (174), (175)*).*

To simplify this definition we have not introduced new number functions of the from $f_{\varphi(z),t}$ that were used along with $F_{\varphi(z),t}$ in the inductive definition of $\mathcal{L}_{\mathbf{FAC}^0}$ (although by 1) everything in $\mathcal{L}_{\mathbf{FAC}^0}$ remains in $\mathcal{L}_{\mathbf{FP}}$). Nevertheless by Cobham's Theorem it is easy to see that semantically the string functions of $\mathcal{L}_{\mathbf{FP}}$ comprise the polytime string functions in **FP**. In particular every string term T over $\mathcal{L}_{\mathbf{FP}}$ is represented by a function symbol of the form $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FP}}$, where (referring to (176)) $\varphi \equiv T(z)$ and t is a suitable bounding term. Note also that every number function in **FP** has the form $|F|$ for some function F in $\mathcal{L}_{\mathbf{FP}}$.

We now define the theory **VPV** in the style of Definition 5.69 of $\overline{\mathbf{V}}^0$.

DEFINITION 8.18. **VPV** *is the theory over* $\mathcal{L}_{\mathbf{FP}}$ *whose axioms are those of* $\overline{\mathbf{V}}^0$ *together with the defining axioms* (176) *for each function* $F_{\varphi(z),t}$ *in* $\mathcal{L}_{\mathbf{FP}}$ *and defining axioms* (174), (175) *for each function* $F_{G,H,t}$ *in* $\mathcal{L}_{\mathbf{FP}}$.

Thus **VPV** is a universal theory which extends $\overline{\mathbf{V}}^0$. Every function introduced in Definition 8.17 is explicitly bounded by a term in $\mathcal{L}_A^2$, and hence **VPV** is a polynomial-bounded theory.

The following general result can be proved by structural induction on φ in the same way as Lemma 3.44 and Lemma 5.70. Our immediate intended application is to take $\mathcal{T} = \mathbf{VPV}$.

LEMMA 8.19. *Let* $\mathcal{T}$ *be a theory with vocabulary* $\mathcal{L}$ *such that* $\mathcal{T}$ *extends* $\mathbf{V}^0$ *and for every open formula* $\varphi(z,\vec{x},\vec{X})$ *over* $\mathcal{L}$ *and term* $t(\vec{x},\vec{X})$ *over* $\mathcal{L}_A^2$ *there is a function* $F_{\varphi(z),t}$ *in* $\mathcal{L}$ *such that*

$$\mathcal{T} \vdash F_{\varphi(z),t}(\vec{x},\vec{X})(z) \leftrightarrow (z < t \wedge \varphi(z,\vec{x},\vec{X}))$$

Then for every $\Sigma_0^B(\mathcal{L})$ *formula* φ *there is an open* $\mathcal{L}$*-formula* φ^+ *such that* $\mathcal{T} \vdash \varphi \leftrightarrow \varphi^+$.

Next we state a general witnessing theorem for universal theories, which applies to **VPV**.

THEOREM 8.20 (Witnessing). *Let* $\mathcal{T}$ *be a universal polynomial-bounded theory which extends* $\mathbf{V}^0$*, with vocabulary* $\mathcal{L}$*, such that for every open formula* $\varphi(z,\vec{x},\vec{X})$ *over* $\mathcal{L}$ *and term* $t(\vec{x},\vec{X})$ *over* $\mathcal{L}_A^2$ *there is a function* $F_{\varphi(z),t}$ *in* $\mathcal{L}$ *such that*

$$\mathcal{T} \vdash F_{\varphi(z),t}(\vec{x},\vec{X})(z) \leftrightarrow z < t \wedge \varphi(z,\vec{x},\vec{X})$$

Then for every theorem of $\mathcal{T}$ *of the form* $\exists Z \varphi(\vec{x},\vec{X},Z)$*, where* φ *is an open formula, there is a function* F *in* $\mathcal{L}$ *such that*

$$\mathcal{T} \vdash \varphi(\vec{x},\vec{X},F(\vec{x},\vec{X}))$$

PROOF. The proof is based on the Herbrand Theorem, and is very similar to the alternative proof of the witnessing theorem for $\mathbf{V}^0$ given in Section 5F.1. This proof defines the witnessing function F by cases, and in fact F has the form $F_{\varphi(z),t}$ for suitable φ, t. By our assumption that $\mathcal{T}$ is polynomial-bounded, we know that there is a bounding term t for $F_{\varphi(z),t}$ in $\mathcal{L}_A^2$ (as opposed to $\mathcal{L}$). $\dashv$

COROLLARY 8.21 (Witnessing for **VPV**). *Every* $\Sigma_1^1(\mathcal{L}_{\mathbf{FP}})$ *theorem of* **VPV** *is witnessed in* **VPV** *by functions in* $\mathcal{L}_{\mathbf{FP}}$.

PROOF. It is clear that **VPV** satisfies the hypotheses for the theory $\mathcal{T}$ in the theorem. Although the theorem only states that formulas of the form $\exists Z\varphi$ (where φ is quantifier-free) can be witnessed, it is easy to generalize it to witness an arbitrary $\Sigma_1^1(\mathcal{L}_{\mathbf{FP}})$ formula $\exists\vec{z}\exists\vec{Z}\varphi$. (See Lemma 5.65 and how it is used to prove the witnessing theorem for $\mathbf{V}^0$.) ⊣

This witnessing result immediately implies the following.

COROLLARY 8.22. *Every function Σ_1^1-definable in* **VPV** *is in* **FP**.

Of course this holds whether we interpret Σ_1^1-definable to mean $\Sigma_1^1(\mathcal{L}_A^2)$-definable, or more generally $\Sigma_1^1(\mathcal{L}_{\mathbf{FP}})$-definable. The converse of the latter, that every polytime function is $\Sigma_1^1(\mathcal{L}_{\mathbf{FP}})$-definable in **VPV**, is obvious, since $\mathcal{L}_{\mathbf{FP}}$ comprises the polytime functions. However we are interested in the stronger converse, that every $\mathcal{L}_{\mathbf{FP}}$-function has a $\Sigma_1^B(\mathcal{L}_A^2)$ definition, provably in **VPV**. This is not straightforward to prove, mainly because we do not have the Σ_0^B-**REPL** axioms available in **VPV** (Theorem 8.82). Section 6C.1 shows how we could proceed if Σ_0^B-**REPL** were available, and Theorem 9.18 shows how we could proceed using aggregate functions. But here we take a different approach: Since $\mathbf{V}^1$ proves the Σ_0^B-**REPL** axioms it is relatively easy to show that every $\mathcal{L}_{\mathbf{FP}}$ function is $\Sigma_1^B(\mathcal{L}_A^2)$-definable in $\mathbf{V}^1$. From this we use the fact that Σ_1^B theorems of $\mathbf{V}^1$ are witnessed in **VPV** to get our desired result (Theorem 8.31).

The next result is proved in the same way as Lemma 5.71.

LEMMA 8.23. **VPV** *proves the* $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-**COMP**, $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-**IND**, $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-**MIN**, *and* $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-**MAX** *axiom schemes.*

DEFINITION 8.24 (Δ_i^B Formula). *Let $\mathcal{T}$ be a theory over $\mathcal{L} \supseteq \mathcal{L}_A^2$. We say that a formula φ is $\Delta_i^B(\mathcal{L})$ in $\mathcal{T}$ if there is a $\Sigma_i^B(\mathcal{L})$ formula φ_1 and a $\Pi_i^B(\mathcal{L})$ formula φ_2 such that $\mathcal{T} \vdash \varphi \leftrightarrow \varphi_1$ and $\mathcal{T} \vdash \varphi \leftrightarrow \varphi_2$.*

COROLLARY 8.25. *If φ is $\Delta_1^B(\mathcal{L}_{\mathbf{FP}})$ in* **VPV** *then* $\mathbf{VPV} \vdash \varphi \leftrightarrow \varphi_0$ *for some open $\mathcal{L}_{\mathbf{FP}}$-formula φ_0.*

PROOF. Suppose that φ is $\Delta_1^B(\mathcal{L}_{\mathbf{FP}})$ in **VPV**, and let φ_1 and φ_2 be as in the definition. Then using pairing functions we may assume that φ_1 and φ_2 each have single string quantifiers, so for some $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-formulas ψ_1, ψ_2 we have

$$\varphi_1 \equiv \exists Y \le t_1 \psi_1(\vec{x}, \vec{X}, Y)$$

$$\varphi_2 \equiv \forall Z \le t_2 \psi_2(\vec{x}, \vec{X}, Z)$$

Since $\mathbf{VPV} \vdash \varphi_2 \supset \varphi_1$ we have

$$\mathbf{VPV} \vdash \exists Y \exists Z,\ \psi_2(\vec{x}, \vec{X}, Z) \supset \psi_1(\vec{x}, \vec{X}, Y)$$

By Corollary 8.21 there are **FP**-functions F and G such that

$$\mathbf{VPV} \vdash \psi_2(\vec{x}, \vec{X}, F(\vec{x}, \vec{X})) \supset \psi_1(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}))$$

Then $\mathbf{VPV} \vdash \varphi \leftrightarrow \varphi_0$, where $\varphi_0 \equiv \psi_1(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}))$. By Lemma 8.19 we may assume ψ_1 is an open $\mathcal{L}_{\mathbf{FP}}$-formula, as required. ⊣

8B.1. Comparing VPV and $\mathbf{V}^1$. Here we prove that every $\mathcal{L}^2_A$-theorem of **VPV** is provable in $\mathbf{V}^1$. We also prove a partial converse, that every Σ^1_1 theorem of $\mathbf{V}^1$ is provable in **VPV**. In Section 8F we show evidence that not all Σ^B_2 theorems of $\mathbf{V}^1$ are provable in **VPV**.

We establish the first assertion by defining an extension $\mathbf{V}^1(\mathbf{VPV})$ of both $\mathbf{V}^1$ and **VPV**, and showing that it is conservative over $\mathbf{V}^1$. We establish the partial converse by showing that every Σ^1_1 theorem of $\mathbf{V}^1$ can be, provably in **VPV**, witnessed by functions in $\mathcal{L}_{\mathbf{FP}}$.

Definition 8.26. *For $i \geq 1$, the theory $\mathbf{V}^i(\mathbf{VPV})$ has vocabulary $\mathcal{L}_{\mathbf{FP}}$, and axioms the union of the axioms for $\mathbf{V}^i$ and for* **VPV**.

Theorem 8.27. (a) *Every function in $\mathcal{L}_{\mathbf{FP}}$ is Σ^B_1-definable in $\mathbf{V}^1$.*

(b) *For $i \geq 1$, every $\Sigma^B_i(\mathcal{L}_{\mathbf{FP}})$-formula is provably equivalent in $\mathbf{V}^1(\mathbf{VPV})$ to a $\Sigma^B_i(\mathcal{L}^2_A)$-formula.*

(c) *For $i \geq 1$, $\mathbf{V}^i(\mathbf{VPV})$ is conservative over $\mathbf{V}^i$.*

Corollary 8.28. *For $i \geq 1$, $\mathbf{V}^i(\mathbf{VPV})$ proves the $\Sigma^B_i(\mathcal{L}_{\mathbf{FP}})$-**COMP**, $\Sigma^B_i(\mathcal{L}_{\mathbf{FP}})$-**IND**, $\Sigma^B_i(\mathcal{L}_{\mathbf{FP}})$-**MIN**, and $\Sigma^B_i(\mathcal{L}_{\mathbf{FP}})$-**MAX** axiom schemes.*

Proof. The corollary follows immediately from part (b) of the theorem, since by Corollary 5.8 $\mathbf{V}^i$ proves these schemes for $\Sigma^B_i(\mathcal{L}^2_A)$-formulas.

Part (a) of the Theorem is essentially proved in Subsection 6B.2. Part (b) for general i follows immediately from the case $i = 1$. Now parts (b) and (c) follow from Corollary 6.27, where we take $\mathcal{T}_0$ to be $\mathbf{V}^1(Row)$, or $\mathbf{V}^i(\mathrm{Row})$ for part (c) (we can get rid of the function *Row* by Lemma 5.52), and the extensions $\mathcal{T}_1, \mathcal{T}_2, \ldots$ are introduced by successively adding the functions in $\mathcal{L}_{\mathbf{FP}}$ and their defining axioms. The fact that the new function introduced in $\mathcal{T}_{i+1}$ is Σ^1_1-definable in $\mathcal{T}_i$ (and even in $\mathcal{T}_0$) is proved in Section 6B.2. ⊣

Theorem 8.29. *Every $\Sigma^1_1(\mathcal{L}_{\mathbf{FP}})$ theorem of $\mathbf{V}^1(\mathbf{VPV})$ is witnessed in* **VPV** *by functions in $\mathcal{L}_{\mathbf{FP}}$.*

Proof. A slight modification of the proof of the Witnessing Theorem for $\mathbf{V}^1$ given in Section 6D.2 proves this theorem. Note that every witnessing function introduced is in **FP**, and, noting that **VPV** proves $\Sigma^B_0(\mathcal{L}_{\mathbf{FP}})$-**IND** (by Lemma 8.23), we see that **VPV** proves the desired sequents. ⊣

The following corollary is immediate from Theorem 8.29.

Corollary 8.30. **VPV** *and $\mathbf{V}^1(\mathbf{VPV})$ have the same $\Sigma^1_1(\mathcal{L}_{\mathbf{FP}})$ theorems.*

In particular, every Σ^B_1 theorem of $\mathbf{V}^1$ is provable in **VPV**. From this and Corollary 8.22 and part (a) of Theorem 8.27 we have the following:

THEOREM 8.31 (Σ_1^1-Definability Theorem for $\mathbf{VPV}$). *A function is $\Sigma_1^1(\mathcal{L}_A^2)$-definable in $\mathbf{VPV}$ iff it is in $\mathbf{FP}$.*

Finally, from Corollary 8.30 and part b) of Theorem 8.27 we have

THEOREM 8.32. *Every $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-formula is provably equivalent in $\mathbf{VPV}$ to a $\Sigma_1^B(\mathcal{L}_A^2)$-formula.*

8B.2. VPV is conservative over VP.

THEOREM 8.33. *$\mathbf{VPV}$ is a conservative extension of $\widehat{\mathbf{VP}}$ and $\mathbf{VP}$.*

PROOF. By definition the vocabulary and axioms of $\mathbf{VPV}$ include the vocabulary and axioms of $\overline{\mathbf{V}}^0$. Also it is easy to see that F'_{MCV} can be defined from functions in $\mathcal{L}_{\mathbf{FAC}^0}$ using limited recursion (174), (175), and its defining axiom (170) is provable in $\mathbf{VPV}$ from these recursion equations using induction (Lemma 8.23). Therefore $\mathbf{VPV}$ is an extension of $\widehat{\mathbf{VP}}$ (and $\mathbf{VP}$).

We now show that $\mathbf{VPV}$ is conservative over $\widehat{\mathbf{VP}}$, and hence by Theorem 8.13 over $\mathbf{VP}$. The functions of $\mathcal{L}_{\mathbf{FP}}$ can be introduced successively, each one either by a Σ_0^B bit definition or by limited recursion, in terms of previously-defined functions. Thus $\mathbf{VPV}$ is the union of theories $\mathcal{T}_i$ satisfying

$$\mathcal{T}_0 \subset \mathcal{T}_1 \subset \mathcal{T}_2 \subset \cdots \tag{177}$$

where $\mathcal{T}_0$ is $\widehat{\mathbf{VP}}$ and for $i > 0$ each $\mathcal{T}_i$ is obtained from $\mathcal{T}_{i-1}$ by adding the defining equation for one new function F_i. We show by induction on i that each new string function F_i is definable in $\widehat{\mathbf{VP}}$ by a $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VP}}})$-formula $\alpha_{F_i}(\vec{x}, \vec{X}, Y)$ satisfying

$$Z = F_i(\vec{x}, \vec{X}) \leftrightarrow \alpha_{F_i}(\vec{x}, \vec{X}, Z) \tag{178}$$

Also $\mathcal{T}_{i-1}$ together with (178) prove the original defining axiom for F_i in $\mathcal{T}_i$.

This shows that each $\mathcal{T}_i$ is conservative over $\mathcal{T}_{i-1}$, and hence $\bigcup \mathcal{T}_i$ is conservative over $\widehat{\mathbf{VP}}$.

Setting $F \equiv F_i$, the formula α_F in (178) for a general string function $F(\vec{x}, \vec{X})$ is based on a family of Boolean circuits $C_F(n_1, n_2)$ which compute F, where n_1 is an upper bound on the length of each argument in $(\vec{x}, \vec{X})$ and n_2 is an upper bound on $|F(\vec{x}, \vec{X})|$. The circuit expects unary notation for the number inputs, so $n_1 \geq x_i$ for each x_i in $\vec{x}$ and $n_1 \geq |X_i|$ for each X_i in $\vec{X}$. $C_F(n_1, n_2)$ is described by a triple

$$(a_F(n_1, n_2), G_F(n_1, n_2), E_F(n_1, n_2))$$

of $\mathbf{FAC}^0$ functions, using the (a, G, E) notation explained in Section 8A. The circuit is monotone, and is based on the "double-rail logic" described in the proof sketch of Proposition 8.7, so each of the inputs in $(\vec{x}, \vec{X})$ must be presented twice; once using the expected bit string and once as the string of negations of those bits. In fact C_F expects its inputs to be the values of gate numbers $2, 3, \ldots, 2n_1k_F + 1$, where k_F is the number of input variables in $\vec{x}, \vec{X}$ (recall that gates 0 and 1 always have the constant values 0 and 1 respectively).

Let the $\mathcal{L}_A^2$ term $t_F(n_1)$ be an upper bound on $|F(\vec{x}, \vec{X})|$, when n_1 is an upper bound on each of the input lengths $\vec{x}, \vec{X}$, and let the $\mathcal{L}_A^2$ term $g_F(n_1)$ be an upper bound on the number of "computing" gates in $C_F(n_1, n_2)$, not counting gates used for inputs and outputs. Then there are $2n_2$ output gates right after the computing gates, which store both $F(\vec{x}, \vec{X})$ and the negations of these bits.

The $\mathbf{FAC}^0$ output function $Out(c, d, Y)$ extracts bits c through $d \dot{-} 1$ of Y, so

$$Out(c, d, Y) = Y(c)Y(c+1)\cdots Y(d \dot{-} 1)$$

The $\mathbf{FAC}^0$ input function $In_F(n_1, \vec{x}, \vec{X}, E) = E'$ augments the edge relation E for C_F, so E' is the same as E except edges from gates 0 and 1 to the input gates $2, 3, \ldots, 2n_1k_F + 1$ are set so that these gates code the values $\vec{x}, \vec{X}$. Thus

(179) $F(\vec{x}, \vec{X}) =$

$$Out\big(c, d, F_{MCV}(a_F(n_1, n_2), G_F(n_1, n_2), In_F(n_1, \vec{x}, \vec{X}, E_F(n_1, n_2)))\big)$$

where

$$\begin{aligned} n_1 &= max\{\vec{x}, |\vec{X}|\} \\ n_2 &= t_F(n_1) \\ a_F(n_1, n_2) &= 1 + 2n_1k_F + g_F(n_1) + 2n_2 \\ c &= 2n_1k_F + g_F(n_1) + 2 \\ d &= c + 2n_2 \end{aligned}$$

Notice that (179) (with the specified $\mathcal{L}_{\mathbf{FAC}^0}$ terms for the variables other than $\vec{x}, \vec{X}$) expresses $F(\vec{x}, \vec{X})$ as a term of $\mathcal{L}_{\widehat{\mathbf{VP}}}$.

Now the formula α_{F_i} in (178) (with $F \equiv F_i$) is given by

(180) $$\alpha_F(\vec{x}, \vec{X}, Z) \equiv \forall j < t, Z(j) \leftrightarrow T(j)$$

where the term T is the RHS of equation (179), and the quantifier bound $t(\vec{x}, \vec{X})$ is $t_F(max\{\vec{x}, |\vec{X}|\})$.

It remains to show that we can define the triple a_F, G_F, E_F of $\mathbf{FAC}^0$ functions specifying the circuits $C_F(n_1, n_2)$ for every function F (or f) in $\mathcal{L}_{\mathbf{FP}}$, in such a way that $\widehat{\mathbf{VP}}$ proves their defining axioms (in terms of earlier functions). In order to show this, we follow Definition 8.17, specifying $\mathcal{L}_{\mathbf{FP}}$. We start with $\mathcal{L}_{\mathbf{FAC}^0}$. The initial functions in $\mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}\}$ have straightforward circuits (recall that the number inputs for $+, \times, pd$ are given in unary notation). After that functions are introduced successively using parts (2) and (3) of the definitions of $\mathcal{L}_{\mathbf{FAC}^0}$ and $\mathcal{L}_{\mathbf{FP}}$, where part (2) introduces functions $F_{\varphi,t}$ and (in the case of $\mathcal{L}_{\mathbf{FAC}^0}$) $f_{\varphi,t}$, where φ is a a quantifier-free formula involving previously-defined functions, and part (3) introduces the function $F_{G,H,t}$ defined from G and H by limited recursion, where G, H are previously-defined.

To illustrate how to build circuits for new functions in terms of old functions we consider a simple example of composition. Suppose

$$F(\vec{x}, \vec{X}) = H(K(\vec{x}, \vec{X})) \tag{181}$$

and suppose that we have a circuits C_K specified by (a_K, G_K, E_K) computing K, and circuits C_H specified by (a_H, G_H, E_H) computing H, where all functions are in $\mathbf{FAC}^0$. Then we can combine these circuits to form C_F by placing C_K in its original position and adding $2n_1k_K + g_K(n_1)$ to each gate number of C_H, so that the input gates of the shifted C_H coincide with the output gates of C_K. Now $\mathbf{FAC}^0$ descriptor functions (a_F, G_F, E_F) for C_F are easily bit-defined by Σ_0^B formulas in terms of (a_K, G_K, E_K) and (a_H, G_H, E_H). In particular the size of C_F is given by

$$a_F(n_1, n_2) = a_K(n_1, t_K(n_1)) + a_H(t_K(n_1), n_2)$$

Note that if the composition (181) is more complicated, say $F = H(K_1, K_2)$, then to describe the circuit C_F for F the circuit C_{K_2} for K_2 needs to be shifted and the original inputs $\vec{x}, \vec{X}$ need to be copied to the input gates for the shifted C_{K_2}, and the outputs for both C_{K_1} and C_{K_2} need to be copied to the inputs for the shifted C_H. But all this is easily accomplished with $\mathbf{FAC}^0$ functions, using the techniques developed in Section 8A.

Each new function F introduced via circuits has a definition given by (178) and (180) and hence satisfies (179). However the theory $\mathcal{T}_i$ in the sequence (177) should be able to prove the defining axioms for F as given in Definition 8.18 for **VPV**. A simple example is when $F \equiv F_{\varphi,t}$ and

$$\varphi(z) \equiv H(K(\vec{x}, \vec{X}))(z)$$

In this case we may assume as an induction hypothesis that $\mathcal{T}_i$ proves (179) when F is replaced by either H or K, and since $\mathcal{T}_i(F)$ defines F by combining the circuits for H and K as explained above we may assume that $\mathcal{T}_i(F)$ proves (179) as it stands. We must show that $\mathcal{T}_i(F)$ proves (181), which amounts to showing that the combined circuits for H and K compute their composition, as intended.

The main lemma needed for this and similar correctness proofs is roughly that if C and C' are two circuits, and gates $a', \ldots, b'$ of C' are the same as gates $a, \ldots, b$ of C but with their numbers shifted by a constant c, and if Y and Y' are correct assignments to C and C' respectively (i,e. (156) holds), and if Y and Y' agree on the 'inputs' to C and C', then $Y(i) \leftrightarrow Y(i+c)$ for $a \le i \le b$. This kind of lemma can be proved in $\overline{\mathbf{V}}^0$ by induction on a $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$ formula.

In case the new function F is introduced by limited recursion, then $F \equiv F_{G,H,t}$, and F, G, H satisfy (174) and (175) (page 200). The circuit $C_F(n_1, n_2)$ for F is built by combining the circuit $C_G(n_1, n_2)$ for G with n_1 shifted copies of the circuit $C_H(max\{n_1, n_2\}, n_2)$ for H, interleaved with circuits computing the sequence of values $0, 1, \ldots, (n_1 - 1)$ for the first argument for H.

The output of C_G is $F(0)$ (i.e. $F(0, \vec{x}, \vec{X})$), and successive outputs of the shifted circuits C_H comprise the sequence $F(1), \ldots, F(n_1)$. The output gates for C_F select from this sequence of outputs the correct output $F(y)$ based on the input argument y. Thus the i-th output gate of C_F is an OR of AND-gates, where the j-th AND-gate has one input from the i-th bit of $F(j)$ and the other input from a selector gate which is on iff $y = j$. This selector gate is the AND of bit j of the input y (which is presented in unary) and bit $j + 1$ of the negated bits of y (which are also part of the input to C_F). ⊣

COROLLARY 8.34. *$\mathbf{V}^1$ is Σ_1^B-conservative over* $\mathbf{VP}$.

PROOF. This is immediate from Theorem 8.33 and Corollary 8.30. ⊣

8C. $\mathbf{TV}^0$ and the $\mathbf{TV}^i$ Hierarchy

We now introduce the $\mathbf{TV}^i$ hierarchy, where for $i > 0$ $\mathbf{TV}^i$ is the two-sorted version of Buss's [12] single-sorted theory $\mathbf{T}_2^i$. For $i = 0$ it turns out that $\mathbf{TV}^0 = \mathbf{VP}$, although the two theories have very different axioms.

For $i \geq 0$ the theory $\mathbf{TV}^i$ is the same as $\mathbf{V}^i$, except instead of the Σ_i^B-$\mathbf{COMP}$ axioms we introduce the Σ_i^B "string induction" axiom scheme. Here we view a string X as the number $\sum_i X(i)2^i$, and define the string zero $\varnothing$ (empty string) and string successor function $S(X)$ as in Example 5.42. Thus $S(X)$ has Σ_0^B-bit definition

$$S(X)(i) \leftrightarrow \varphi_S^{bit}(i, X) \tag{182}$$

where

$$\varphi_S^{bit}(i, X) \ \equiv \ i \leq |X| \wedge [(X(i) \wedge \exists j < i \neg X(j)) \vee (\neg X(i) \wedge \forall j < i X(j))]$$

DEFINITION 8.35 (String Induction Axiom). *If Φ is a set of formulas, then the string induction axiom scheme, denoted Φ-$\mathbf{SIND}$, is the set of all formulas*

$$[\varphi(\varnothing) \wedge \forall X(\varphi(X) \supset \varphi(S(X))] \ \supset \ \varphi(Y) \tag{183}$$

where $\varphi(X)$ is in Φ, and may have free variables other than X.

Since we want the theories $\mathbf{TV}^i$ to have underlying language $\mathcal{L}_A^2$, in case Φ has vocabulary $\mathcal{L}_A^2$ we will interpret (183) as a formula over $\mathcal{L}_A^2$, using the standard method of eliminating Σ_0^B-bit-definable function symbols (Lemma 5.40).

Definition 8.36. *For $i \geq 0$, $\mathbf{TV}^i$ is the theory over $\mathcal{L}_A^2$ with axioms those of $\mathbf{V}^0$ together with the Σ_i^B-**SIND** scheme.*

Although the induction scheme (183) has an unbounded string quantifier, it is easy to see that the theory $\mathbf{TV}^i$ remains the same if that quantifier $\forall X$ is replaced by the bounded quantifier $\forall X \leq |Y|$ (see Exercise 3.16). Hence $\mathbf{TV}^i$ is a polynomial-bounded theory, axiomatized by Σ_{i+1}^B-formulas.

Lemma 8.37. *For $i \geq 0$, $\mathbf{TV}^i$ proves Σ_i^B-**IND**.*

Proof. We are to show that $\mathbf{TV}^i$ proves

$$[\varphi(0) \wedge \forall x(\varphi(x) \supset \varphi(x+1))] \supset \varphi(z)$$

where $\varphi(x)$ is Σ_i^B.

We need the following easily verified fact:

$$\mathbf{V}^0 \vdash (|S(X)| = |X| \ \vee \ |S(X)| = |X| + 1) \tag{184}$$

Reasoning in $\mathbf{TV}^i$, assume

$$[\varphi(0) \wedge \forall x(\varphi(x) \supset \varphi(x+1))]$$

From this and (184) we conclude

$$[\psi(\varnothing) \wedge \forall X(\psi(X) \supset \psi(S(X)))]$$

where $\psi(X) \equiv \varphi(|X|)$. Hence $\psi(X_z)$ follows by Σ_i^B-**SIND**, where X_z is a string with length z. Hence $\varphi(z)$. ⊣

Theorem 8.38. *For $i \geq 0$, $\mathbf{V}^i \subseteq \mathbf{TV}^i$.*

Proof. We generalize Definition 6.33 to define $\tilde{\mathbf{V}}^i$ to be $\mathbf{V}^0 + \Sigma_i^B$-**IND**. The proof of Theorem 6.35 easily generalizes to show $\mathbf{V}^i = \tilde{\mathbf{V}}^i$. Hence the theorem follows from Lemma 8.37. ⊣

Just as $\mathbf{V}^i$ proves the number minimization and maximization axioms for Σ_i^B-formulas (Corollary 5.8), $\mathbf{TV}^i$ proves the stronger string minimization and maximization axioms for Σ_i^B-formulas. First, we define the ordering relation for strings.

Definition 8.39 (String Ordering). *The string relation $X \leq Y$ has defining axiom*

$$\begin{aligned} X \leq Y \leftrightarrow [X = Y \ \vee \ (&|X| \leq |Y| \wedge \\ &\exists z \leq |Y| \, (Y(z) \wedge \neg X(z) \wedge \forall u \leq |Y|, \, z < u \supset (X(u) \supset Y(u))))] \end{aligned} \tag{185}$$

Often our vocabularies do not contain extra relation symbols outside $\mathcal{L}_A^2$. Thus the syntactic formula $X \leq Y$ will be an abbreviation for the RHS of Equation (185).

EXERCISE 8.40. *Show that the following are theorems of* $\mathbf{V}^0$ *(where* $\varnothing, S, +$ *are defined in Example 5.42):*

(a) $X \le Y \vee Y \le X$ *($X \le Y$ is a total order).*
(b) $(X \le Y \wedge Y \le X) \supset X = Y$ *($X \le Y$ is irreflexive).*
(c) $\varnothing \le X$.
(d) $X \le Y \leftrightarrow X + Z \le Y + Z$.

For a string term T, we define $\exists X \le T\ \varphi(X)$ as an abbreviation for $\exists X(X \le T \wedge \varphi(X))$. Similarly, $\forall X \le T\ \varphi(X)$ is an abbreviation for $\forall X(X \le T \supset \varphi(X))$. Note that the bounding term T is for the *value* of X, while the bounding term t in $\exists X \le t \ldots$ or $\forall X \le t \ldots$ is for the length of X (Definition 4.13).

DEFINITION 8.41 (String Minimization and Maximization Axioms). *The string minimization axiom scheme for* Φ, *denoted* Φ-**SMIN**, *is*

$$\varphi(Y) \supset \exists X \le Y, \varphi(X) \wedge \neg\exists Z < X \varphi(Z)$$

where φ *is a formula in* Φ. *Similarly the string maximization axioms scheme for* Φ, *denoted* Φ-**SMAX**, *is*

$$\varphi(\varnothing) \supset \exists X \le Y, \varphi(X) \wedge \neg\exists Z \le Y(X < Z \wedge \varphi(Z))$$

where φ *is a formula in* Φ.

THEOREM 8.42. *For* $i \ge 0$, $\mathbf{TV}^i$ *proves the* Σ_i^B-**SMIN** *and* Σ_i^B-**SMAX** *axioms.*

PROOF. To prove Σ_i^B-**SMAX**, let $\varphi(X)$ be a Σ_i^B-formula. Let $\varphi'(X)$ be the Σ_i^B-formula obtained by taking a prenex form of

$$X \le Y \supset \exists U \le Y(X \le U \wedge \varphi(U))$$

Then the **SMAX** axiom for $\varphi(X)$ follows from the **SIND** axiom (183) applied to $\varphi'(X)$.

The proof of Σ_i^B-**SMIN** is similar, but uses the binary subtraction function $Z \dot{-} Y$. ⊣

EXERCISE 8.43. *Show that the limited subtraction function for string* $Z \dot{-} Y$ *is* Σ_0^B*-bit-definable, where the intended meaning of* $Z \dot{-} Y$ *is* $\varnothing$ *if* $Z \le Y$, *and* $(Z \dot{-} Y) + Y = Z$ *otherwise.*

We now concentrate on $\mathbf{TV}^0$.

THEOREM 8.44. $\mathbf{TV}^0 = \mathbf{VP}$.

PROOF. Subsection 8C.1 shows that $\mathbf{TV}^0 \subset \mathbf{VPV}$, and by Theorem 8.33 $\mathbf{VPV}$ is conservative over $\mathbf{VP}$. Hence $\mathbf{TV}^0 \subseteq \mathbf{VP}$. The reverse inclusion is shown in Subsection 8C.2. ⊣

By Theorem 8.44 we know the properties of $\mathbf{VP}$ proved in Section 8A also hold for $\mathbf{TV}^0$. In particular $\mathbf{TV}^0$ is finitely axiomatizable, the functions Σ_1^B-definable in $\mathbf{TV}^0$ comprise $\mathbf{FP}$, and by Corollary 8.34 $\mathbf{V}^1$ is Σ_1^B-conservative over $\mathbf{TV}^0$.

In the following corollary, $\mathbf{TV}^i(\mathbf{VPV})$ is defined analogously to $\mathbf{V}^i(\mathbf{VPV})$ in Definition 8.26, namely it has the vocabulary of $\mathbf{VPV}$ and the axioms are the union of the axioms for $\mathbf{TV}^i$ and $\mathbf{VPV}$. (See also Theorem 8.27 and Corollary 8.28).

Corollary 8.45. *For* $i \geq 0$, $\mathbf{TV}^i(\mathbf{VPV})$ *is a conservative extension of* $\mathbf{TV}^i$.

Proof. For $i = 0$ this follows from the fact that $\mathbf{VPV}$ is a conservative extension of $\mathbf{TV}^0$ (Theorems 8.33 and 8.44). For $i \geq 1$ we know $\mathbf{V}^1 \subseteq \mathbf{TV}^i$, and hence $\mathbf{TV}^i$ Σ_1^B-defines all functions in $\mathcal{L}_{\mathbf{FP}}$, and also $\mathbf{TV}^i$ proves Σ_1^B-$\mathbf{REPL}$ by Corollary 6.24. Therefore the corollary follows from Corollary 6.27. ⊣

8C.1. $\mathbf{TV}^0 \subseteq \mathbf{VPV}$. In this subsection we use the string addition function $X + Y$ introduced in Chapter 5 and use some of its simple properties stated in Exercise 5.44. We also need the string relation $X \leq Y$ (Definition 8.39) and the string function $POW2(x)$ defined below. The intended meaning of $POW2(x)$ is such that (see Notation on page 82) $bin(POW2(x)) = 2^x$.

Example 8.46. *The string function* $POW2(x)$, *also denoted by* $\{x\}$, *has bit defining axiom*

$$POW2(x)(i) \leftrightarrow i = x$$

Exercise 8.47. *Show that* $\overline{\mathbf{V}}^0$ *proves the following:*

$$X + POW2(0) = S(X)$$
$$X < POW2(|X|)$$
$$POW2(i) + POW2(i) = POW2(i+1)$$

The following theorem suffices to prove $\mathbf{TV}^0 \subset \mathbf{VPV}$. That $\mathbf{VPV}$ proves the open string induction axioms may seem surprising, since unwinding the induction requires exponentially many steps.

Theorem 8.48. $\mathbf{VPV}$ *proves the* $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-$\mathbf{SIND}$ *axioms.*

Proof. By Lemma 8.19 we may assume that $\varphi(X)$ in (183) is an open $\mathcal{L}_{\mathbf{FP}}$-formula. Let $\vec{y}, \vec{Y}$ be a list of the parameters in $\varphi(X)$. We use binary search to define in $\mathbf{VPV}$ an $\mathcal{L}_{\mathbf{FP}}$ function $G(\vec{y}, \vec{Y}, X)$ such that $\mathbf{VPV}$ proves

$$(186) \qquad (\varphi(\varnothing) \wedge \neg\varphi(X)) \supset ((\varphi(G(\vec{y}, \vec{Y}, X)) \wedge \neg\varphi(S(G(\vec{y}, \vec{Y}, X)))$$

from which (183) follows immediately.

In more detail, we use the string functions $X + Y$ and $POW2(x)$ and the string relation $X \leq Y$ defined above.

In the following we suppress mention of the parameters $\vec{y}, \vec{Y}$.

Define the formula

$$\varphi'(X, Z) \equiv \varphi(Z) \wedge Z \leq X$$

Now we use limited recursion (174), (175) (page 200 to define in **VPV** the binary search function $H(i, X)$, whose value is the left end of the interval $[A, B]$ of length $POW2(|X| \dot{-} i)$ satisfying $\varphi'(X, A) \wedge \neg\varphi'(X, B)$. (Recall the number function $x \dot{-} y$ (limited subtraction), Section 3C.3).

Let $n = |X|$.

$$H(0, X) = \varnothing$$

$$H(i+1, X) = \begin{cases} H(i, X) & \text{if } \neg\varphi'(X, H(i, X) + POW2(n \dot{-} (i+1))) \\ H(i, X) + POW2(n \dot{-} (i+1)) & \text{otherwise} \end{cases}$$

We can use $|X|$ as a bounding term to limit this recursion. Now define

$$G(X) = H(|X|, X)$$

The following two formulas can be proved in **VPV** by induction on i (Lemma 8.23), using Exercises 5.44 and 8.47. The first formula justifies $|X|$ as a length bound for the recursion.

$$X \neq \varnothing \supset (H(i, X) + POW2(0)) \leq X$$

$$(\varphi(\varnothing) \wedge \neg\varphi(X) \wedge i \leq n) \supset$$
$$(\varphi'(X, H(i, X)) \wedge \neg\varphi'(X, H(i, X) + POW2(n \dot{-} i)))$$

Then (186) follows from these two formulas and $X + POW2(0) = S(X)$ (Exercise 8.47). $\dashv$

Recall the notion of a Δ_i^B formula in a theory (Definition 8.24).

DEFINITION 8.49. *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L}$. Let* **AX** *denote any of the axiom schemes* **COMP**, **IND**, **SIND**, *etc. We say that $\mathcal{T}$ proves Δ_i^B-**AX** if for any $\Delta_i^B(\mathcal{L})$ formula φ in $\mathcal{T}$, $\mathcal{T}$ proves the* **AX** *axiom for φ.*

From Theorem 8.48 and Corollary 8.25 we have

COROLLARY 8.50. **VPV** *proves* Δ_1^B-**SIND**.

8C.2. Bit Recursion. In order to show that $\mathbf{VP} \subseteq \mathbf{TV}^0$ we introduce a bit-recursion scheme and show that it is provable in $\mathbf{TV}^0$.

For each formula $\varphi(i, X)$ (possibly with other free variables) we define a formula $\varphi^{rec}(y, X)$ which says that each bit i of X is defined in terms of the preceding bits of X using φ. That is, using the notation $X^{<i}$ for $Cut(i, X)$ (see (96) on page 133)

$$\varphi^{rec}(y, X) \equiv \forall i < y(X(i) \leftrightarrow \varphi(i, X^{<i}))$$

In case $\varphi(i, X)$ is an $\mathcal{L}_A^2$-formula we can interpret $\varphi^{rec}(y, X)$ as an $\mathcal{L}_A^2$-formula by eliminating occurrences of $Cut(i, X)$ using the standard method of eliminating Σ_0^B-bit-definable function symbols (Lemma 5.40).

If $\varphi(i, X)$ is in Σ_0^B it is easy to see that $\mathbf{V}^0$ can use induction on y to prove that the condition $\varphi^{rec}(y, X)$ uniquely determines bits $X(0), ..., X(y-1)$ of X.

DEFINITION 8.51. *If Φ is a set of formulas, then the bit recursion axiom scheme, denoted* Φ-**BIT-REC***, is the set of formulas*

$$\exists X \varphi^{rec}(y, X) \tag{187}$$

where $\varphi(i, X)$ is in Φ, and may have free variables other than X.

We will show that $\mathbf{TV}^0 = \mathbf{V}^0 + \Sigma_0^B$-**BIT-REC**.

THEOREM 8.52. $\mathbf{TV}^0$ *proves the* Σ_0^B-**BIT-REC***-scheme.*

PROOF. We use Σ_0^B-**SMAX** to prove the existence of X in (187). Informally, imagine computing the bits $X(0), \dots, X(y-1)$ of X in that order. Suppose that false negative is allowed, but there is no false positive. That is, we consider strings Y that satisfy

$$\forall i < y,\, Y(i) \supset \varphi(i, Y^{<i})$$

The idea is that the maximal string Y guaranteed by **SMAX** cannot have any false negative bit, and thus must be the correct string.

To actually use the **SMAX** principle we need a twist in the above argument. This is because we compute X in (187) from bit 0, while string comparison starts with high order bits. Thus, let the string reversal function $Rev(y, X)$ have bit-defining axiom

$$Rev(y, X)(i) \leftrightarrow i < y \wedge X(y \dot{-} i \dot{-} 1)$$

where $\dot{-}$ is limited subtraction (Section 3C.3). Then $Rev(y, X)$ is the reverse of the string $X(0) \dots X(y-1)$.

Let $\varphi'(y, Y)$ be the formula

$$\forall i < y,\, Rev(y, Y)(i) \supset \varphi(i, (Rev(y, Y))^{<i}) \tag{188}$$

We can tacitly assume that $\varphi'(y, Y)$ is Σ_0^B (by Lemma 5.40). It is easy to see that $\varphi'(y, \varnothing)$. Thus, by Σ_0^B-**SMAX**, there is a maximal string $X' \le POW2(y)$ that satisfies (188). It is also easy to show (in $\mathbf{V}^0$) that X' in fact satisfies

$$\forall i < y,\, Rev(y, X')(i) \leftrightarrow \varphi(i, (Rev(y, X'))^{<i})$$

As a result, the string $X = Rev(y, X')$ satisfies (187). ⊣

LEMMA 8.53. $\mathbf{VP} \subseteq \mathbf{V}^0 + \Sigma_0^B$-**BIT-REC**

PROOF. Observe that the axiom MCV for **VP** (Definition 8.1) is an instance of Σ_0^B-**BIT-REC**. ⊣

This lemma completes the proof of Theorem 8.44, showing that $\mathbf{VP} = \mathbf{TV}^0$.

COROLLARY 8.54. $\mathbf{TV}^0$ *proves its* Δ_1^B-**SIND** *axioms.* $\mathbf{V}^1$ *proves its* Δ_1^B-**SIND** *axioms.*

PROOF. The first sentence follows from $\mathbf{VP} = \mathbf{TV}^0$ and Corollary 8.50. The second sentence follows from the first, since by Corollary 8.34 any Σ_1^B-formula that is Δ_1^B in $\mathbf{V}^1$ is also Δ_1^B in $\mathbf{TV}^0$. ⊣

8D. The Theory $\mathbf{V}^1$-HORN

This section will not be needed for any later results, but it is interesting in that gives more evidence for the rubustness of **VP** by giving yet another axiomatization.

The theory $\mathbf{V}^1$-**HORN** [31], is the same as **VP** and $\mathbf{TV}^0$ but presented with very different axioms. The of ideal of $\mathbf{V}^1$-**HORN** comes from a theorem of Grädel in descriptive complexity theory, characterizing the class **P** as the sets of finite models of certain second-order formulas. We will formulate Grädel's theorem as a representation theorem over $\mathcal{L}_A^2$. We start with some definitions and examples.

Definition 8.55. *A* Horn formula *is a propositional formula in conjunctive normal form such that each clause (i.e. conjunct) is a* Horn clause*, i.e. it contains at most one positive occurrence of a variable.*

Horn formulas are important because the satisfiability problem HornSat (given a Horn formula, determine whether it is satisfiable) is complete for **P**. A polytime algorithm for HornSat can be described as follows.

HornSat Algorithm: To test whether a given Horn formula A is satisfiable, initialize a truth assignment τ by assigning $\perp$ to each atom of A. Now repeat the following until satisfiability is determined: If τ satisfies all clauses of A then decide that A is satisfiable. Otherwise select a clause C of A not satisfied by τ. If C has no positive occurrence of any atom then decide that A is unsatisfiable. Otherwise C has a unique positive occurrence of some atom p, in which case flip the value of τ on p from $\perp$ to $\top$.

Exercise 8.56. *Show that the above algorithm runs in polynomial time and correctly determines whether a given Horn formula A is satisfiable.*

The HornSat algorithm suggests that a Horn clause $(p \vee \neg q_1 \vee \cdots \vee \neg q_k)$ can be written as an assignment statement

$$p \leftarrow (q_1 \wedge \cdots \wedge q_k)$$

(In fact some logic-based programming languages such as Prolog use this idea.)

We now indicate why HornSat is complete for **P**. It suffices to show that a known complete problem CVP (Circuit Value Problem) can be reduced to HornSat. Given a Boolean circuit C with binary gates $\wedge, \vee$ and unary gates $\neg$, and given a value $v(x) \in \{0, 1\}$ for each input x to C, we want to find a Horn formula A which is satisfiable iff C has output 1 for the given inputs $v(x)$. The formula A uses double rail logic (see the proof of Proposition 8.7) to evaluate C: for each gate and each input x of C the formula has two atoms x^+ and x^- asserting that the gate or input is 1 or 0, respectively. For each such x, A has a Horn clause $(\neg x^+ \vee \neg x^-)$ to insure that not both atoms are true. For each input x, A has a unit clause x^+ if $v(x) = 1$ and unit clause x^- if $v(x) = 0$. For each gate in C, A has up to three Horn clauses which assert that the output of the gate has the appropriate value with respect to its inputs. For example, if x is the $\vee$ of inputs y, z, then the clauses are

$$(x^+ \leftarrow y^+) \wedge (x^+ \leftarrow z^+) \wedge (x^- \leftarrow (y^- \wedge z^-)) \tag{189}$$

Finally A has the unit clause x^+_{out}, where x_{out} is the output gate.

It turns out that the collection of propositional Horn formulas that correspond to a given polytime problem can be represented by single Σ^B_1 formula as follows.

DEFINITION 8.57. *A Σ^B_1-**Horn** formula is an $\mathcal{L}^2_A$-formula of the form*

$$\varphi \equiv \exists Z_1 \dots \exists Z_k \forall y_1 \le t_1 \dots \forall y_m \le t_m \psi \tag{190}$$

where $k, m \ge 0$ and ψ is quantifier-free in conjunctive normal form and each clause contains at most one positive occurrence of a literal of the form $Z_i(t)$. No term of the form $|Z_i|$ may occur in φ, although φ may contain free string variables X (and free number variables) with no restriction on occurrences of $|X|$, and any clause of ψ may contain any number of positive (or negative) literals of the form $X(t)$.

We will show that Σ^B_1-Horn formulas represent polynomial time relations in their free variables.

EXAMPLE 8.58 ($Parity(X)$). *This is a Σ^B_1-**Horn**-formula which holds iff the string X contains an odd number of 1's. $Parity(X)$ encodes a dynamic-programming algorithm for computing the parity of X: $Z_{odd}(i)$ is true (and $Z_{even}(i)$ is false) iff the prefix of X of length i contains an odd number of 1's.*

$$\begin{gathered}
\exists Z_{even} \exists Z_{odd} \forall i < |X| \\
Z_{even}(0) \wedge \neg Z_{odd}(0) \wedge Z_{odd}(|X|) \wedge \\
(\neg Z_{even}(i+1) \vee \neg Z_{odd}(i+1)) \wedge \\
(\neg Z_{even}(i) \vee \neg X(i) \vee Z_{odd}(i+1)) \wedge (\neg Z_{odd}(i) \vee \neg X(i) \vee Z_{even}(i+1)) \wedge \\
(\neg Z_{even}(i) \vee X(i) \vee Z_{even}(i+1)) \wedge (\neg Z_{odd}(i) \vee X(i) \vee Z_{odd}(i+1))
\end{gathered}$$

EXERCISE 8.59. *Prove that $Parity(X)$ has the stated property.*

In Section 4C.2 we showed how the complexity classes $\mathbf{AC}^0$ and the members Σ^P_i of the polynomial hierarchy can be characterized by representation theorems involving the formula classes Σ^B_i. Now we state a similar theorem characterizing $\mathbf{P}$.

THEOREM 8.60 (Grädel). *A relation $R(\vec{x}, \vec{X})$ is polynomial time iff it is represented by some Σ^B_1-**Horn**-formula.*

Proof sketch. ($\Longleftarrow$) Suppose that the formula $\varphi(\vec{x}, \vec{X})$ has the form (190). We outline an algorithm that runs in time polynomial in $(\vec{x}, |\vec{X}|)$ which, given values for $\vec{x}, \vec{X}$, determines whether $\varphi(\vec{x}, \vec{X})$ holds (in the standard model). First note that once values for $\vec{x}, \vec{X}$ are given, the bounding terms $t_i = t_i(\vec{x}, \vec{X})$ can be evaluated to numbers bounded by polynomials in $(\vec{x}, |\vec{X}|)$. We expand the quantifier prefix $\forall y_1 \leq t_1 \ldots \forall y_m \leq t_m$ by giving all possible m-tuples of values $(y_1, \ldots, y_m)$ satisfying the bounding terms, and form the conjunction $\Psi(Z_1, \ldots, Z_k)$ of all instances $\psi(\vec{y})$, as $\vec{y}$ ranges over all these tuples. (Note that the number of such tuples is bounded by a polynomial in $(\vec{x}, |\vec{X}|)$.)

Then $\Psi(Z_1, \ldots, Z_k)$ can be made into a propositional conjunctive normal form formula Ψ' involving only literals of the form $Z_i(j)$ and $\neg Z_i(j)$ for specific numbers j, since all terms and all other variables in ψ have been evaluated. (Here it is important that we have disallowed occurrences of $|Z_i|$ in φ.) The arguments j in $Z_i(j)$ and $\neg Z_i(j)$ are values of terms t, for each $Z_i(t)$ or $\neg Z_i(t)$ that is a literal in the original formula ψ. Let B be an upper bound on the possible values of j (so B is a polynomial in $(\vec{x}, \vec{X})$). Then Ψ' is a Horn formula whose propositional variables are all in the set $\{Z_i(j) \mid i \leq k, j \leq B\}$. Thus the problem of checking for the existence of $Z_1, \ldots, Z_k$ reduces to the polytime HornSat problem of deciding whether Ψ' is satisfiable.

($\Longrightarrow$) Let $R(\vec{x}, \vec{X})$ be a polytime relation and let M be a deterministic polytime Turing machine that recognizes R in time $t(\vec{x}, \vec{X})$. By choosing t large enough, the entire computation of M on input $\vec{x}, \vec{X}$ can be represented (using the pairing function) by an array $Z(i, j)$ with t rows and columns, where the i-th row specifies the tape configuration at time i. Thus $R(\vec{x}, \vec{X})$ is represented by the Σ_1^B-**Horn**-formula

$$\exists Z \exists \tilde{Z} \forall i \leq t \forall j \leq t \psi(i, j, \vec{x}, \vec{X}, Z, \tilde{Z})$$

Here the variable $\tilde{Z}$ is forced to be $\neg Z$ in the same way that Z_{even} and Z_{odd} are forced to be complementary in the parity example above. The formula ψ satisfies the conditions in Definition 8.57 and each clause specifies a local condition on the computation. $\dashv$

Definition 8.61. *The theory* $\mathbf{V}^1$-**HORN** *has vocabulary* $\mathcal{L}_A^2$ *and axioms those of* $\mathbf{V}^0$ *together with* Σ_1^B-**Horn-COMP**.

The original definition of $\mathbf{V}^1$-**HORN** in [31] was a little different. Recall that $\mathbf{V}^0$ has axioms 2-**BASIC** together with Σ_0^B-**COMP** (Definition 5.3). The original definition was essentially $\mathbf{V}^1$-**HORN** = 2-**BASIC** + Σ_1^B-**Horn-COMP**. It was shown with some effort that $\mathbf{V}^1$-**HORN** proves Σ_0^B-**COMP**, so the two definitions are equivalent.

The next theorem follows from results in [31].

Theorem 8.62. $\mathbf{V}^1$-**HORN** = **VP**.

PROOF SKETCH. $\mathbf{V}^1$-**HORN** $\subseteq$ **VP**: It suffices to show

$$\mathbf{VP} \vdash \Sigma_1^B\text{-}\mathbf{Horn\text{-}COMP}$$

Since **VPV** is a conservative extension of **VP** (Theorem 8.33), it suffices to show $\mathbf{VPV} \vdash \Sigma_1^B$-**Horn-COMP**. Since $\mathbf{VPV} \vdash \Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$-**COMP** (Lemma 8.23), it suffices to show that for every Σ_1^B-**Horn**-formula φ there is a $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}})$ formula φ' such that $\mathbf{VPV} \vdash \varphi \leftrightarrow \varphi'$.

So let φ be a Σ_1^B-**Horn**-formula as in (190), where we write $\psi(Z_1, \dots, Z_k)$ simply as ψ, and let $\vec{x}, \vec{X}$ be the free variables in φ. The idea is to find a "witnessing function" $F_i(\vec{x}, \vec{X})$ in $\mathcal{L}_{\mathbf{FP}}$ for each Z_i such that **VPV** proves $\varphi \leftrightarrow \varphi'$, where

$$\varphi' \equiv \forall y_1 \le t_1 \dots \forall y_m \le t_m \psi(F_1(\vec{x}, \vec{X}), \dots F_k(\vec{x}, \vec{X}))$$

To define F_i we refer to the direction $\Longleftarrow$ in the proof of Theorem 8.60. There the algorithm to evaluate $\varphi(\vec{x}, \vec{X})$ computes a propositional Horn formula Ψ' whose propositional variables have the form $Z_i(j)$, and then applies the HornSat algorithm to determine whether Ψ' is satisfiable. This algorithm computes a truth assignment τ to the atoms $Z_i(j)$ of Ψ' such that Ψ' is satisfiable iff τ satisfies Ψ'. Thus it suffices to define the string $F_i(\vec{x}, \vec{X})$ to be the array of truth values that τ gives to Z_i. That is, the the bit definition of each F_i is

$$F_i(\vec{x}, \vec{X})(j) \leftrightarrow j \le B \wedge \tau(Z_i(j))$$

The algorithm outlined to compute F_i is clearly polytime and hence corresponds to some function in **FP**. The missing details in the proof are to show that **VPV** proves the correctness of the algorithm; i.e. $\mathbf{VPV} \vdash \varphi \supset \varphi'$.

VP $\subseteq$ $\mathbf{V}^1$-**HORN**: By Definition 8.1 it suffices to show that

$$\mathbf{V}^1\text{-}\mathbf{HORN} \vdash MCV$$

We indicated earlier (189) how propositional Horn clauses can be used to evaluate circuit gates. Now we show how to use a Σ_1^B-**Horn** formula to evaluate the circuit C described by parameters a, G, E as described in Section 8A. In essence, the new atoms x^+, x^-, etc. in (189) are encoded by the (existentially quantified) string variables Z in the Σ_1^B-**Horn** formula. Note that the algorithm outlined on page 214 is for circuits with binary gates, while here the circuit may have unbounded fan-ins.

Thus we want to define an array $Z(x)$ (and its negation $\tilde{Z}(x)$) to evaluate gate x in C. We will put in the clause

$$\neg Z(x) \vee \neg \tilde{Z}(x)$$

to make sure that not both are true. For gates 0 and 1 (with constant values 0 and 1 respectively) we put in the four clauses

$$\tilde{Z}(0), \qquad \neg Z(0), \qquad \neg \tilde{Z}(1), \qquad Z(1) \tag{191}$$

Next, consider gate x. Suppose that this is an $\vee$-gate, i.e., $\neg G(x)$ holds. Then we need several clauses. The first is

$$\big(\neg G(x) \wedge y < x \wedge E(y,x) \wedge Z(y)\big) \supset Z(x)$$

which assures that $Z(x)$ holds if at least one of the inputs to gate x is 1. To ensure that $\tilde{Z}(x)$ holds if all inputs to gate x are 0 is more involved. In fact, we formalize a simple algorithm that runs through the inputs of gate x to check if all of them are 0. We use a string variable P, where $P(x,y)$ is intended to mean that all gates u which are input to x, where $u < y$, output 0. The formalization is as follows:

$$\begin{aligned} & P(x,0) \\ (P(x,y) \wedge \neg E(y,x)) \ &\supset\ P(x,y+1) \\ (P(x,y) \wedge \tilde{Z}(y)) \ &\supset\ P(x,y+1) \\ (\neg G(x) \wedge P(x,x)) \ &\supset\ \tilde{Z}(x) \end{aligned}$$

Let $\psi_\vee$ denote the set of the five clauses described above for the case where the gate (x) is an $\vee$-gate. Also, let ψ_I be the set of clauses in (191). The set $\psi_\wedge$ of clauses for handling the case where (x) is an $\wedge$-gate is similar to $\psi_\vee$, using an extra variable Q instead of P.

Exercise 8.63. *Give the five clauses of $\psi_\wedge$.*

Now we can show in $\mathbf{V}^0$ that a string Y that is computed by

$$\text{(192)}\quad Y(i) \leftrightarrow \exists Z \exists \tilde{Z} \exists P \exists Q \forall x < a \forall y < a, \\ (\neg Z(x) \vee \neg \tilde{Z}(x)) \wedge \psi_I \wedge \psi_\wedge \wedge \psi_\vee \wedge Z(i)$$

(for $i < a$) satisfies $\delta_{MCV}(a,G,E,Y)$. The following exercise is helpful.

Exercise 8.64. *Let the string variables $Z, \tilde{Z}, P, Q$ satisfy the RHS of* (192), *and Y' satisfy $\delta_{MCV}(a,G,E,Y')$. Show by induction on i that for $i < a$,*

$$\neg Z(i) \supset \neg Y'(i) \qquad \textit{and} \qquad \neg \tilde{Z}(i) \supset Y'(i)$$

Exercise 8.65. *Prove by number induction that the string Y described above satisfies the recursion in $\delta_{MCV}(a,G,E,Y)$.*

Finally, the existence of Y in MCV follows from the existence of Y that satisfies (192), and the latter follows from Σ_1^B-**Horn-COMP**. This completes the proof that $\mathbf{VP} \subseteq \mathbf{V}^1$-**HORN**. $\dashv$

8E. $\mathbf{TV}^1$ and Polynomial Local Search

It follows from Theorem 8.38 that $\mathbf{V}^1 \subseteq \mathbf{TV}^1$, and hence $\mathbf{TV}^1$ can Σ_1^B-define all polynomial time functions. But there is no known nice characterization of the set of *all* functions Σ_1^B-definable in $\mathbf{TV}^1$. There is however a nice characterization of the set of all *search problems* Σ_1^B-definable in $\mathbf{TV}^1$.

A search problem is essentially a multivalued function, and the associated computational problem is to find one of the possible values. Here we are concerned with *total* search problems, which means that the set of possible values is always nonempty. We present a search problem by its graph. The search problem is definable in a theory if the theory proves its totality. In the two-sorted setting the set of possible values is a set of strings.

DEFINITION 8.66. *A* search problem Q_R *is a multivalued function with graph* $R(\vec{x}, \vec{X}, Z)$*, so*

$$Q_R(\vec{x}, \vec{X}) = \{Z \mid R(\vec{x}, \vec{X}, Z)\}$$

Here the arity of either or both of $\vec{x}, \vec{X}$ *may be zero. The search problem is* total *if the set* $Q_R(\vec{x}, \vec{X})$ *is non-empty for all* $\vec{x}, \vec{X}$*. The search problem is a* function problem *if* $|Q_R(\vec{x}, \vec{X})| = 1$ *for all* $\vec{x}, \vec{X}$*. A function* $F(\vec{x}, \vec{X})$ solves Q_R *if*

$$F(\vec{x}, \vec{X}) \in Q_R(\vec{x}, \vec{X})$$

for all $\vec{x}, \vec{X}$*.*

Here we will be concerned only with total search problems. The following notion of reduction preserves totality.

DEFINITION 8.67. *A search problem* Q_{R_1} *is many-one reducible to a search problem* Q_{R_2}*, written* $Q_{R_1} \leq_{\mathbf{AC}^0} Q_{R_2}$*, provided there are* $\mathbf{FAC}^0$*-functions* $\vec{f}, \vec{F}, G$ *such that* $G(\vec{x}, \vec{X}, Z) \in Q_{R_1}(\vec{x}, \vec{X})$ *for all* $Z \in Q_{R_2}(\vec{f}(\vec{x}, \vec{X}), \vec{F}(\vec{x}, \vec{X}))$*.*

We note that the usual definition states the weaker requirement that $\vec{f}, \vec{F}, G$ are polytime functions. However experience shows that when reductions are needed they can be made to meet our stronger requirement.

EXERCISE 8.68. *Show that* $\leq_{\mathbf{AC}^0}$ *is a transitive relation. Also show that if* $Q_{R_1} \leq_{\mathbf{AC}^0} Q_{R_2}$ *and* Q_{R_2} *is solvable by a polytime function, then* Q_{R_1} *is solvable by a polytime function.*

Local search is a method of finding a local maximum of a function by starting at a point in the domain of the function, finding a neighbor of the point that increases the value of the function, and continuing this process until no such neighbor exists. Polynomial Local Search (**PLS**) formalizes this as a search problem in case the function is polytime and suitable neighboring points can be found in polynomial time. Recall that $\varnothing$ denotes the empty set (Example 5.42).

DEFINITION 8.69. *A* **PLS** *problem* Q *is specified by the following:*

1) *A polytime relation* $\varphi_Q(\vec{x}, \vec{X}, Z)$ *and an* $\mathcal{L}_A^2$*-term* $t(\vec{x}, \vec{X})$ *satisfying the two conditions*

$$\varphi_Q(\vec{x}, \vec{X}, \varnothing)$$
$$\varphi_Q(\vec{x}, \vec{X}, Z) \supset |Z| \leq t(\vec{x}, \vec{X})$$

($\{Z \mid \varphi_Q(\vec{x}, \vec{X}, Z)\}$ *is the set of* candidate solutions *for problem instance* $(\vec{x}, \vec{X})$*.)*

2) *Polytime string functions* $P_Q(\vec{x}, \vec{X}, Z)$ *and* $N_Q(\vec{x}, \vec{X}, Z)$ *satisfying the two conditions*

$$\varphi_Q(\vec{x}, \vec{X}, Z) \supset \varphi_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Z))$$
$$N_Q(\vec{x}, \vec{X}, Z) \neq Z \supset P_Q(\vec{x}, \vec{X}, Z) < P_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Z))$$

(N_Q *is a heuristic for finding a neighbor of* Z *which increases the profit* P_Q*.* $N_Q(\vec{x}, \vec{X}, Z) = Z$ *is taken to mean that* Z *is locally optimal. Here* $X < Y$ *stands for* $X \leq Y \wedge \neg X = Y$*, where* $X \leq Y$ *is defined in Definition 8.39.)*

Then

$$Q(\vec{x}, \vec{X}) = \{Z \mid \varphi_Q(\vec{x}, \vec{X}, Z) \wedge N_Q(\vec{x}, \vec{X}, Z) = Z\} \tag{193}$$

The problem Q *is an* $\mathbf{AC}^0$-**PLS** *problem if* φ_Q, N_Q, P_Q *are* $\mathbf{AC}^0$*-relations and functions.*

It is easy to see that a **PLS** problem is a total search problem. For fixed $\vec{x}, \vec{X}$, the set of candidate solutions Z (those satisfying $\varphi_Q(\vec{x}, \vec{X}, Z)$) is nonempty and bounded. Thus given $\vec{x}, \vec{X}$, any candidate solution Z that maximizes the profit $P_Q(\vec{x}, \vec{X}, Z)$ is a member of $Q(\vec{x}, \vec{X})$.

We will concentrate on a subclass of **PLS** called **ITERATION**, which is complete for **PLS**.

DEFINITION 8.70. *An* **ITERATION** *problem* $Q = Q_F$ *is specified by a polytime function* $F(\vec{x}, \vec{X}, Z)$ *and a bounding term* $t(\vec{x}, \vec{X})$*. The graph relation* R *is specified by a formula* $\psi_F(\vec{x}, \vec{X}, Z)$ *which is (suppressing the parameters* $\vec{x}, \vec{X}$*):*

$$\begin{aligned}\psi_F(Z) \;\equiv\; &(Z = \varnothing \wedge F(\varnothing) = \varnothing) \;\vee \\ &|Z| \leq t \wedge Z < F(Z) \wedge \big(t < |F(Z)| \vee F(F(Z)) \leq F(Z)\big)\end{aligned} \tag{194}$$

Then

$$Q_F(\vec{x}, \vec{X}) = \{Z \mid \psi_F(\vec{x}, \vec{X}, Z)\} \tag{195}$$

The problem Q_F *is an* $\mathbf{AC}^0$-**ITERATION** *problem if* F *is an* $\mathbf{AC}^0$*-function.*

To see that Q_F is a total search problem, note that the largest $Z \leq t$ such that $(Z = \varnothing \vee Z < F(Z))$ is always a solution.

LEMMA 8.71. *Every* **ITERATION** *problem is a* **PLS** *problem.*

PROOF. Let Q_F be an **ITERATION** problem as above. Then Q_F can be specified as a **PLS** problem using the following definitions:

$$\varphi_Q(Z) \equiv |Z| \le t \wedge (Z = \varnothing \vee Z < F(Z))$$

$$P_Q(Z) = Z$$

$$N_Q(Z) = \begin{cases} F(Z) \text{ if } |F(Z)| \le t \text{ and } Z < F(Z) < F(F(Z)) \\ Z \text{ otherwise} \end{cases}$$

Then (195) follows from (193). Notice that if Q_F is an $\mathbf{AC}^0$-**ITERATION** problem then the corresponding problem is an $\mathbf{AC}^0$-**PLS** problem. ⊣

THEOREM 8.72. *Every* **PLS** *problem is many-one reducible to some* **ITERATION** *problem. Every* $\mathbf{AC}^0$-**PLS** *problems is many-one reducible to some* $\mathbf{AC}^0$-**ITERATION** *problem.*

PROOF. Let Q be a **PLS** problem and let t, φ_Q, P_Q, N_Q be as in Definition 8.69.

We give the following Σ_0^B-definition of the concatenation function $X *_z Y$, which is the first z bits of X followed by Y:

$$(X *_z Y)(i) \leftrightarrow i < z + |Y| \wedge [i < z \wedge X(i) \ \vee \ z \le i \wedge Y(i \dot- z)]$$

We wish to define an **ITERATION** problem Q_F with bounding term t' whose solutions yield solutions of Q. The idea is to let the domain of F consist of concatenations $U *_t V$ where U is a candidate solution for Q and V is its profit. Note that if $V_1 < V_2$ then $U_1 *_t V_1 < U_2 *_t V_2$ for all U_1, U_2.

In the following we suppress the parameters $\vec{x}, \vec{X}$.

Let $u = u(\vec{x}, \vec{X})$ be an $\mathcal{L}_A^2$-term large enough so that $|P_Q(N_Q(Z))| \le u$ for $|Z| \le t$. Then define

$$t' = t + u$$

and

$$F(U *_t V) = \begin{cases} N_Q(U) *_t P_Q(N_Q(U)) \text{ if } V = P_Q(U) \text{ and } \varphi_Q(U) \\ U *_t V \text{ otherwise} \end{cases}$$

The term t' is chosen so that if U satisfies $\varphi_Q(U)$ then $|F(U *_t P_Q(U))| \le t'$.

Here we redefine P_Q so that $P_Q(\varnothing) = \varnothing$. Note that the result is a **PLS** problem with the same solutions as the original problem.

Now suppose Z is a solution to the **ITERATION** problem Q_F. We show how to obtain a solution $G(Z)$ $(= G(\vec{x}, \vec{X}, Z))$ to the original **PLS** problem Q. We write $Z = U *_t V$ where U, V are uniquely determined by Z. Then from (193), (195) and our definitions we see that $G(U *_t V) = N_Q(U)$ is a solution to Q.

Hence by Definition 8.67 we conclude $Q \le_{\mathbf{AC}^0} Q_F$, where $\vec{f}, \vec{F}$ take $\vec{x}, \vec{X}$ to itself and $G(\vec{x}, \vec{X}, Z) = N_Q(\vec{x}, \vec{X}, Z^{<t(\vec{x}, \vec{X})})$. ⊣

DEFINITION 8.73. *If* $\mathbf{S}$ *is a set of search problems, then* $\mathbf{CC}(\mathbf{S})$ *is the set of search problems many-one reducible to* $\mathbf{S}$.

THEOREM 8.74.

$$\mathbf{CC}(\mathbf{ITERATION}) = \mathbf{CC}(\mathbf{PLS}) =$$
$$\mathbf{CC}(\mathbf{AC}^0\textbf{-ITERATION}) = \mathbf{CC}(\mathbf{AC}^0\textbf{-PLS})$$

PROOF. The first and last equalities follow from the preceding definition and theorem. The middle equality follows from these and Theorem 8.76 below. ⊣

DEFINITION 8.75. *Let* $Q(\vec{x}, \vec{X})$ *be a search problem with graph* $R(\vec{x}, \vec{X}, Z)$. *We say that* Q *is* Φ*-definable in a theory* $\mathcal{T}$ *if there is a formula* $\psi_R(\vec{x}, \vec{X}, Z)$ *in* Φ *such that*

$$\psi_R(\vec{x}, \vec{X}, Z) \supset R(\vec{x}, \vec{X}, Z)$$

and

$$\mathcal{T} \vdash \exists Z \psi_R(\vec{x}, \vec{X}, Z)$$

THEOREM 8.76. *The following are equivalent for a search problem* Q:

(a) Q *is* Σ_1^B*-definable in* $\mathbf{TV}^1$.
(b) Q *is in* $\mathbf{CC}(\mathbf{PLS})$.
(c) Q *is in* $\mathbf{CC}(\mathbf{AC}^0\textbf{-PLS})$.

PROOF. (a) $\Longrightarrow$ (c) follows from Theorem 8.77 below (Witnessing for $\mathbf{TV}^1$) and Lemma 8.71. (c) $\Longrightarrow$ (b) is obvious. Hence it suffices to show (b) $\Longrightarrow$ (a).

By Theorems 8.72 and 8.32 and Corollary 8.45 it suffices to show that every problem in $\mathbf{CC}(\mathbf{ITERATION})$ is $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-definable in $\mathbf{TV}^1(\mathbf{VPV})$. We start by showing this for every **ITERATION** problem Q_F. Let $\psi_F(\vec{x}, \vec{X}, Z)$ be the formula (194) defining Q_F. We may assume that F is an $\mathcal{L}_{\mathbf{FP}}$-function, and hence ψ_F is a $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-formula. Let

$$\eta(\vec{x}, \vec{X}, Z) \equiv (Z = \varnothing \vee Z < F(\vec{x}, \vec{X}, Z))$$

Then **VPV** proves η is equivalent to a Σ_1^B-formula (Theorem 8.32), and hence by Σ_1^B-**SMAX** (Theorem 8.42), $\mathbf{TV}^1(\mathbf{VPV})$ proves the existence of a largest $Z \le t$ satisfying $\eta(Z)$. Thus $\mathbf{TV}^1(\mathbf{VPV})$ proves that this Z satisfies $\psi_F(Z)$.

This shows that every **ITERATION** problem is $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-definable in $\mathbf{TV}^1(\mathbf{VPV})$. Now suppose the search Q_{R_1} is many-one reducible to some **ITERATION** problem Q_{R_2}. Define the formula $\psi_{R_1}(\vec{x}, \vec{X}, Z)$ by (suppressing $\vec{x}, \vec{X}$)

$$\psi_{R_1}(Z) \equiv \exists W \le t(Z = G(W) \wedge \psi_{R_2}(\vec{f}, \vec{F}, W))$$

where t is the bounding term for Q_{R_2} and ψ_{R_2} is a $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-formula which defines Q_{R_2} in $\mathbf{TV}^1(\mathbf{VPV})$, and $\vec{f}, \vec{F}, G$ show $Q_{R_1} \le_{\mathbf{AC}^0} Q_{R_2}$

according to Definition 8.67. Then ψ_{R_1} is equivalent to a $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-formula, and by Definition 8.67

$$\psi_{R_1}(\vec{x}, \vec{X}, Z) \supset R_1(\vec{x}, \vec{X}, Z)$$

Since by assumption $\mathbf{TV}^1(\mathbf{VPV})$ proves $\exists W \leq u\, \psi_{R_2}(W)$ (where u is a bounding term from Parikh's Theorem) it follows that $\mathbf{TV}^1(\mathbf{VPV})$ proves $\exists Z \psi_{R_1}(Z)$, as required. ⊣

THEOREM 8.77 (Witnessing Theorem for $\mathbf{TV}^1$). *Suppose that $\varphi(\vec{x}, \vec{X}, Z)$ is a Σ_1^1-formula such that*

$$\mathbf{TV}^1 \vdash \exists Z \varphi(\vec{x}, \vec{X}, Z)$$

*Then there is an $\mathbf{AC}^0$-**ITERATION** problem Q_F with graph $\psi_F(\vec{x}, \vec{X}, Z)$ from* (194) *and an $\mathbf{FAC}^0$-function G such that*

$$\overline{\mathbf{V}}^0 \vdash \psi_F(\vec{x}, \vec{X}, Z) \supset \varphi(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}, Z))$$

PROOF. By using pairing functions we may assume that φ is Σ_0^B. The proof is similar to the proof of the Witnessing Theorem for $\mathbf{V}^1$ (Section 6D). Thus we define a sequent system $\mathbf{LK}^2$-$\mathbf{TV}^1$, which is the same as $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ except that we replace the **IND** Rule by the *single*-Σ_1^B-**SIND** Rule, defined below. Recall (Example 5.42) the $\mathbf{AC}^0$ functions $\varnothing$ (empty set) and $S(X)$ (successor of X). For the next definition, when Φ is $\Sigma_i^B(\mathcal{L}_A^2)$ (for $i \geq 0$) the formulas $A(S(\delta))$ and $A(\varnothing)$ are understood to be the equivalent $\Sigma_i^B(\mathcal{L}_A^2)$ formulas as stated by the $\mathbf{FAC}^0$ Elimination Lemma 5.74.

DEFINITION 8.78 (The **SIND** Rule). *For a set Φ of formulas, the* Φ-**SIND** rule *consists of the inferences of the form*

$$(196) \qquad \frac{\Gamma, A(\delta) \longrightarrow A(S(\delta)), \Delta}{\Gamma, A(\varnothing) \longrightarrow A(T), \Delta}$$

where A is a formula in Φ and T is a string term.
Restriction. *The variable δ is called an* eigenvariable *and does not occur in the bottom sequent.*

The proof that $\mathbf{LK}^2$-$\mathbf{TV}^1$ is a complete system for $\mathbf{TV}^1$ is the same as the proof that $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ is a complete system for $\tilde{\mathbf{V}}^1$, with obvious modifications. Further the proof of Theorem 6.42, Anchored Completeness for $\mathbf{LK}^2$+**IND**, works for $\mathbf{LK}^2$-$\mathbf{TV}^1$, so every theorem of $\mathbf{TV}^1$ has an anchored $\mathbf{LK}^2$-$\mathbf{TV}^1$ proof.

Now we proceed as in the proof of the Witnessing Theorem for $\mathbf{V}^1$ (Section 6D.2) and for $\mathbf{V}^0$ (Section 5E.2), with appropriate changes.

Suppose that $\exists Z \varphi(\vec{x}, \vec{X}, Z)$ is a Σ_1^1-theorem of $\mathbf{TV}^1$, where φ is a Σ_0^B-formula. Then there is an anchored $\mathbf{LK}^2$-$\mathbf{TV}^1$ proof π of $\longrightarrow \exists Z \varphi(\vec{a}, \vec{\alpha}, Z)$. We may assume that π is in free variable normal form. By the Subformula Property the formulas in π are Σ_1^1 formulas, and in fact they are Σ_0^B formulas or *single*-Σ_1^1 formulas. As a result, every sequent in π has the form

$$\mathcal{S} = \quad \underbrace{\exists X_i \theta_i(X_i)}_{i=1,\dots,m}, \Gamma \longrightarrow \Delta, \underbrace{\exists Y_j \eta_j(Y_j)}_{j=1,\dots,n} \tag{197}$$

for $m, n \geq 0$, where θ_i and η_j and all formulas in Γ and Δ are Σ_0^B.

We will prove by induction on the depth in π of the sequent $\mathcal{S}$ that there is an $\mathbf{AC}^0$-**ITERATION** problem Q_F with graph ψ_F and for $1 \leq i \leq n$ there are $\mathcal{L}_{\mathbf{FAC}^0}$-functions G_i such that $\overline{\mathbf{V}}^0$ proves (the semantic equivalent of) the sequent

$$\mathcal{S}' = \quad \underbrace{\theta_i(\beta_i)}_{i=1,\dots,m}, \Gamma, \psi_F(\vec{a}, \vec{\alpha}, \vec{\beta}, \gamma) \longrightarrow \Delta, \underbrace{\eta_j(G_j(\vec{a}, \vec{\alpha}, \vec{\beta}, \gamma))}_{j=1,\dots,n} \tag{198}$$

where $\vec{a}, \vec{\alpha}$ is a list of exactly those variables with free occurrences in $\mathcal{S}$. (This list may be different for different sequents.) Also $\beta_1, \dots, \beta_m$ are distinct new free variables corresponding to the bound variables $X_1, \dots, X_m$, although the latter variables may not be distinct. When $\mathcal{S}$ is the final sequent of π, note that Γ and Δ are empty, $i = 0$, $j = 1$, and $\vec{\beta}$ is empty, so the theorem follows.

Note that this induction hypothesis is the same as in the proof for $\mathbf{V}^1$ and $\mathbf{V}^0$, except now each witnessing function G_j is allowed to take the argument γ, which is a solution to the **ITERATION** problem Q_F. As before, the induction step has a case for Σ_0^B-**COMP** and for each rule. The argument for Σ_0^B-**COMP** is the same as for $\mathbf{V}^0$ (since the witnessing function G_j can ignore its argument γ). The argument for each rule except Σ_1^B-**SIND** is similar to that for $\mathbf{V}^0$ (Section 5E.2), and can be obtained using the following lemma, that shows how two **ITERATION** problems can be combined into one.

LEMMA 8.79 (Composition of **ITERATION** Problems). *Suppose that Q_{F_1} and Q_{F_2} are* **ITERATION** *problems with graphs $\psi_{F_1}(\vec{x}, \vec{X}, U)$ and $\psi_{F_2}(\vec{x}, \vec{X}, U, V)$. Then there is an* **ITERATION** *problem Q_F with graph $\psi_F(\vec{x}, \vec{X}, Z)$ such that F is Σ_0^B-bit-definable from F_1, F_2, and there are* $\mathbf{FAC}^0$*-functions $G_1(\vec{x}, \vec{X})$ and $G_2(\vec{x}, \vec{X})$ such that (suppressing $\vec{x}, \vec{X}$)*

$$\overline{\mathbf{V}}^0(F_1, F_2, F) \vdash \psi_F(Z) \supset \psi_{F_1}(G_1(Z)) \wedge \psi_{F_2}(G_1(Z), G_2(Z))$$

PROOF. Assume the hypotheses of the Lemma, and let t be the bounding term for Q_{F_1} and let u be the bounding term for Q_{F_2}. Using the notation $U *_t V$ in the proof of Theorem 8.72, we express the argument Z in $F(\vec{x}, \vec{X}, Z)$ in the form

$$Z = U *_t V *_{t+u} \delta$$

where δ is a binary string equal to 0,1,or 2. We abbreviate Z by

$$Z = U * V * \delta$$

Then we define F by (suppressing $\vec{x}, \vec{X}$)

$$F(U{*}V{*}\delta) = \begin{cases} U * V * 2 & \text{if } \psi_{F_1}(U) \wedge \psi_{F_2}(U,V) \wedge \delta \leq 1 \\ U * F_2(U,V) * 1 & \text{if } \psi_{F_1}(U) \wedge |V| \leq u \wedge \\ & \qquad V < F_2(U,V) \wedge \delta \leq 1 \\ F_1(U) * \varnothing * \varnothing & \text{if } V = \delta = \varnothing \wedge |U| \leq t \wedge U < F_1(U) \\ U * V * \delta & \text{otherwise} \end{cases}$$

Let the **ITERATION** problem Q_F have bounding term $t + u + 2$.

We claim that

(199) $\overline{\mathbf{V}}^0(F_1, F_2, F) \vdash \ \psi_F(U * V * \delta) \supset \delta = 2 \wedge \psi_{F_1}(U) \wedge \psi_{F_2}(U,V)$

To see this, note that by line 3 in the definition of F, $F(\varnothing) \neq \varnothing$, since if $F_1(\varnothing) = \varnothing$ then $\psi_{F_1}(\varnothing)$, and hence one of the first two lines applies. Hence assuming $\psi_F(U * V * \delta)$ we have by (194)

$$U * V * \delta < F(U * V * \delta) = F(F(U * V * \delta))$$

From the definitions of ψ_{F_1} and ψ_{F_2} we see that this can only happen if line 1 applies in evaluating $F(U * V * \delta)$.

This establishes (199). To prove the lemma, we define

$$G_1(U * V * \delta) = U \qquad G_2(U * V * \delta) = V$$

We can make these definitions explicit by defining

$$G_1(\vec{x}, \vec{X}, Z) = Z^{<t} \qquad G_2(\vec{x}, \vec{X}, Z) = Z[t, t+u]$$

$\dashv$

It remains to handle the case in which $\mathcal{S}$ is obtained by an application of the Σ_1^B-**SIND** rule. Then $\mathcal{S}$ is the bottom sequent of

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda, \exists X \leq r(\delta)\theta(\delta, X) \longrightarrow \exists X \leq r(S(\delta))\theta(S(\delta), X), \Pi}{\Lambda, \exists X \leq r(\varnothing)\theta(\varnothing, X) \longrightarrow \exists X \leq r(T)\theta(T, X), \Pi}$$

where δ does not occur in $\mathcal{S}$ and θ is Σ_0^B.

By the induction hypothesis for the top sequent $\mathcal{S}_1$ it follows that $\overline{\mathbf{V}}^0$ proves a sequent $\mathcal{S}_1'$ of the form

(200) $$\mathcal{S}_1' = \ \Lambda', \eta_1, \psi_F(\delta, \beta, \gamma) \longrightarrow \eta_2, \Pi'$$

where

(201) $$\eta_1 \ \equiv \ |\beta| \leq r(\delta) \wedge \theta(\delta, \beta)$$

(202) $$\eta_2 \ \equiv \ |G(\delta, \beta, \gamma)| \leq r(S(\delta)) \wedge \theta(S(\delta), G(\delta, \beta, \gamma))$$

and ψ_F defines the graph of an $\mathbf{AC}^0$-**ITERATION** problem Q_F and G is an $\mathcal{L}_{\mathbf{FAC}^0}$-function. Here δ, β, γ do not occur in Λ', but they may occur in Π' as arguments to the witnessing functions G_j.

Our task is to use Q_F and G to find $Q_{F'}$ and G' to find a witness for $\exists X \leq r(T)\theta(T, X)$, given a witness β_0 for $\exists X \leq r(\varnothing)\theta(\varnothing, X)$. We want $\overline{\mathbf{V}}^0$ to prove the following sequent $\mathcal{S}'$:

$$\mathcal{S}' = \quad \Lambda', \rho_1, \psi_{F'}(\beta_0, \gamma') \longrightarrow \rho_2, \Pi'' \tag{203}$$

where

$$\rho_1 \;\equiv\; |\beta_0| \leq r(\varnothing) \wedge \theta(\varnothing, \beta_0) \tag{204}$$

$$\rho_2 \;\equiv\; |G'(\beta_0, \gamma')| \leq r(T) \wedge \theta(T, G'(\beta_0, \gamma')) \tag{205}$$

and Π'' will be given later.

We will use the technique in the proof of Lemma 8.79 and assume that the search variable γ' for $Q_{F'}$ has the form

$$\gamma' = \beta *_{r(T)} \gamma *_{r(T)+t} \delta$$

where β, γ, δ are as in (200), and t an upper bound for γ based on the bounding term for Q_F. In the following we drop the subscripts to $*$ and write

$$\gamma' = \beta{*}\gamma{*}\delta$$

The idea is that $Q_{F'}$ uses F and G to find witnesses β for successive string values of $\delta = 1, 2, \ldots, T$ knowing that β_0 is a witness in case $\delta = \varnothing$. $Q_{F'}$ should succeed under the assumption that (200) holds for all $\delta < T$ and all β, assuming that the formulas in Λ' are true and those in Π' are false.

We define $F'(\beta_0, \beta * \gamma * \delta)$ by cases in such a way that if η_1 holds, then it continues to hold when F' is applied repeatedly, and progress is made toward finding β' such that $\theta(T, \beta')$.

$$F'(\beta_0, \beta{*}\gamma{*}\delta) = \begin{cases} \quad G(\delta, \beta, \gamma) * \varnothing * S(\delta) & \text{if } \eta_1 \wedge \delta < T \wedge \psi_F(\delta, \beta, \gamma) \\ \text{else } \beta * F(\beta, \delta, \gamma) * \delta & \text{if } \eta_1 \wedge \delta < T \wedge \gamma < F(\beta, \delta, \gamma) \\ \text{else } \beta_0{*}\varnothing{*}\varnothing & \text{if } \beta = \gamma = \delta = \varnothing \\ \text{else } \beta{*}\gamma{*}\delta & \end{cases}$$

We define the witness-extracting function $G'(\beta_0, \gamma')$ as follows:

$$G'(\beta_0, \beta{*}\gamma{*}\delta)) = \begin{cases} \beta_0 & \text{if } T = \varnothing \\ G(\delta, \beta, \gamma) & \text{if } T \neq \varnothing \end{cases}$$

The following Claim asserts that a witness for $\exists X \theta(T, X)$ can be obtained from a solution $\beta{*}\gamma{*}\delta$ to $Q_{F'}$, provided (200) holds with Λ' true and Π' false.

Claim. $\overline{\mathbf{V}}^0$ proves

$$T \neq \varnothing, \rho_1, \psi_{F'}(\beta_0, \beta{*}\gamma{*}\delta) \longrightarrow \eta_1 \wedge \psi_F(\delta, \beta, \gamma) \wedge (\neg\eta_2 \vee \rho_2)$$

Proof of the Claim: We argue in $\overline{\mathbf{V}}^0$. Assume $T \neq \varnothing, \rho_1, \psi_{F'}(\beta_0, \beta{*}\gamma{*}\delta)$. By $\psi_{F'}(\beta_0, \beta{*}\gamma{*}\delta)$ and (194) there are two possibilities. The first is that $F'(\varnothing) = \varnothing$. But this is impossible, because if $\beta = \gamma = \delta = \varnothing$

then either $\beta_0 \neq \varnothing$ and line 3 in the definition of F' applies, or $\beta_0 = \varnothing$ and one of the first two lines applies (by ρ_1 and the definition of ψ_F).

Therefore the second possibility in the definition of $\psi_{F'}(\beta_0, \beta{*}\gamma{*}\delta)$ applies, and we have

$$\beta{*}\gamma{*}\delta < F'(\beta{*}\gamma{*}\delta) = F'(F'(\beta{*}\gamma{*}\delta)) \tag{206}$$

Analyzing the definition of F' and our assumptions ($T \neq \varnothing, \rho_1$) shows that the only way that (206) can hold is if line 1 in the definition of F' applies when evaluating $F'(\beta{*}\gamma{*}\delta)$. Thus $\eta_1 \wedge \psi_F(\delta, \beta, \gamma)$. Also since line 1 applies, if $S(\delta) < T$ then $\neg\eta_2$, for otherwise line 1 or line 2 would apply when evaluating $F'(F'(\beta{*}\gamma{*}\delta))$, contradicting the second part of (206). This proves the Claim in case $S(\delta) < T$. Finally if $S(\delta) = T$ then $\eta_2 \supset \rho_2$, and the Claim follows.

To establish that $\overline{\mathbf{V}}^0$ proves (203) we need to specify Π'' by giving values (in terms of γ') for the variables δ, β, γ which occur as arguments to the functions G_j in Π'. Motivated by the Claim and (200) we define, for $\gamma' = \beta{*}\gamma{*}\delta$,

$$B(\gamma') = \beta, \qquad GA(\gamma') = \gamma, \qquad D(\gamma') = \delta$$

and define Π'' to be the result of replacing β, γ, δ in Π' by $B(\gamma')$, $GA(\gamma')$, $D(\gamma')$ respectively.

The fact that $\overline{\mathbf{V}}^0$ proves (203) now follows from the Claim and by (200) with β, γ, δ replaced by $B(\gamma')$, $GA(\gamma')$, $D(\gamma')$. (The case $T = \varnothing$ follows from $(T = \varnothing \wedge \rho_1) \supset \rho_2$, which holds by definition of G'.) $\dashv$

8F. KPT Witnessing and Replacement

Here we present a generalization of the Herbrand Theorem from Chapter 2 and show how it can be used to prove the independence of the Replacement Axiom Scheme (Section 6C) in some cases. In Section 8G.3 we use it to show how the collapse of the polynomial hierarchy follows from the collapse of the bounded arithmetic hierarchy $\mathbf{V}^i$.

Form 2 of the Herbrand Theorem (Corollary 2.68) applies to a $\forall\exists$ consequence of a universal theory. The next result is a generalization which applies to $\forall\exists\forall$ consequences. We call it the KPT Witnessing Theorem, after the authors of [58], who used it to prove the first part of Theorem 8.102.

Theorem 8.80 (KPT Witnessing). *Let $\mathcal{T}$ be a universal two-sorted theory whose vocabulary $\mathcal{L}$ includes at least one string constant or function symbol. Let φ be an open $\mathcal{L}$-formula and suppose*

$$\mathcal{T} \vdash \forall X \exists Y \forall Z \varphi(X, Y, Z)$$

Then there exists a finite sequence $T_1, \ldots, T_k$ *of string terms such that*

$$\mathcal{T} \vdash \varphi(X, T_1(X), Z_1) \vee \varphi(X, T_2(X, Z_1), Z_2) \\ \vee \cdots \vee \varphi(X, T_k(X, Z_1, \ldots, Z_{k-1}), Z_k).$$

where the notation $T_i(X, Z_1, \ldots, Z_{i-1})$ *means that only the displayed variables may occur in* T_i.

In our applications of this theorem each term T_i is a function

$$F_i(X, Z_1, \ldots, Z_{i-1})$$

in some complexity class such as $\mathbf{FAC}^0$ or $\mathbf{FP}$. The "student-teacher" interpretation of the theorem [57] is a useful way to think of it. The student is given X and wants to find Y satisfying $\forall Z \varphi(X, Y, Z)$, but has computing power limited to the relevant complexity class. The student starts by trying $Y = F(X)$. The teacher either approves, or comes up with a counter-example Z_1 such that $\neg\varphi(X, F(X), Z_1)$. The student next tries $Y = F(X, Z_1)$, and the teacher either agrees or supplies a counter-example Z_2. This process continues for at most k steps after which the student finds a value of Y that works for all Z.

Proof of Theorem 8.80. Let $B, C_1, C_2, ...$ be a list of new string constants, and let $U_1, U_2, ...$ be an enumeration of all terms built from symbols of $\mathcal{L}$ together with $B, C_1, C_2, ...$, where the only new constants in U_k are among $\{B, C_1, ..., C_{k-1}\}$. It suffices to show that

$$\mathcal{T} \cup \{\neg\varphi(B, U_1, C_1), \neg\varphi(B, U_2, C_2), \ldots, \neg\varphi(B, U_k, C_k)\}$$

is unsatisfiable for some k.

Suppose otherwise. Then by compactness

$$\mathcal{T} \cup \{\neg\varphi(B, U_1, C_1), \neg\varphi(B, U_2, C_2), ...\} \tag{207}$$

has a model $\mathcal{M}$. Since $\mathcal{T}$ is universal, the substructure $\mathcal{M}'$ consisting of the denotations of the terms $U_1, U_2, ...$ is also a model for (207). It is easy to see that

$$\mathcal{M}' \models \mathcal{T} + \forall Y \exists Z \neg\varphi(B, Y, Z)$$

and hence $\mathcal{T} \not\vdash \forall X \exists Y \forall Z \varphi(X, Y, Z)$. ⊣

8F.1. Applying KPT Witnessing. Following [36] we now outline the method for using the KPT Witnessing Theorem to show that a universal theory $\mathcal{T}$ which extends $\overline{\mathbf{V}}^0$ and has a vocabulary $\mathcal{L}$ associated with certain complexity classes cannot prove the $\Sigma_0^B(\mathcal{L})$-**REPL** axioms (sometimes subject to complexity assumptions). Our main examples are $\mathcal{T} = \overline{\mathbf{V}}^0$ and $\mathcal{T} = \mathbf{VPV}$. That $\mathbf{VPV}$ is unlikely to prove Σ_0^B-Replacement may seem surprising, since $\mathbf{V}^1$ (which has the same Σ_1^B-theorems) even proves Σ_1^B-Replacement (Corollary 6.24).

Choose a function F which is in the relevant complexity class but whose inverse probably is not. Suppose $\mathcal{T}$ proves the following instance of replacement (which has W as a parameter, and $t = t(W)$ and $u = u(W)$ as terms):

$$(\forall i < t\, \exists Z < u\, F(Z) = W^{[i]}) \supset \exists Y \forall j < u\, F(Y^{[j]}) = W^{[j]}. \tag{208}$$

We can rewrite this as

$$\exists i < t\, \exists Y \forall Z < u\big(F(Z) = W^{[i]} \supset \forall j < t\, F(Y^{[j]}) = W^{[j]}\big)$$

Applying the KPT Witnessing Theorem we get a positive integer k and functions $g_1, \ldots, g_k, H_1, \ldots, H_k$ such that $\mathcal{T}$ proves

$$\begin{aligned}
&(F(Z_1) = W^{[g_1(W)]} \supset \forall j < t\, F(H_1(W)^{[j]}) = W^{[j]})\\
&\vee (F(Z_2) = W^{[g_2(W,Z_1)]} \supset \forall j < t\, F(H_2(W, Z_1)^{[j]}) = W^{[j]})\\
&\vee \ldots\\
&\vee (F(Z_k) = W^{[g_k(W,Z_1,\ldots,Z_{k-1})]} \supset \forall j < t\, F(H_k(W, Z_1, \ldots, Z_{k-1})^{[j]}) = W^{[j]})
\end{aligned}$$

This allows the "student", given an input W (considered as a sequence $W^{[0]}, \ldots, W^{[t-1]}$), to compute Y coding a sequence of pre-images of F of all t elements of W, by asking the "teacher" for pre-images of at most k elements of W.

The student proceeds as follows. Let $Y = H_1(W)$. If $\forall j < t\, F(Y^{[j]}) = W^{[j]}$ then output Y and halt. Otherwise compute $g_1(W)$ and ask the teacher for a pre-image Z_1 of $W^{[g_1(W)]}$. Let $Y = H_2(W, Z_1)$. If $\forall j < t\, F(Y^{[j]}) = W^{[j]}$ then output Y and halt. Otherwise compute $g_2(W, Z_1)$ and ask the teacher for a pre-image Z_2 of $W^{[g_2(W,Z_1)]}$, and so on. By our assumption the algorithm will run for at most k steps of this form before it outputs a suitable Y.

THEOREM 8.81 ([36]). *$\mathbf{V}^0$ and $\overline{\mathbf{V}}^0$ do not prove Σ_0^B-**REPL**.*

PROOF. Since $\overline{\mathbf{V}}^0$ extends $\mathbf{V}^0$ and every $\Sigma_i^B(\mathcal{L}_{\mathbf{FAC}^0})$-formula is provably equivalent to a Σ_i^B-formula (Lemma 5.74), it suffices to prove the theorem for the case $\overline{\mathbf{V}}^0$.

Recall that $PARITY(X)$ holds iff the string X has an odd number of ones. We have pointed out that $PARITY$ is not an $\mathbf{AC}^0$ relation, but in fact is is known [40, 2] that $PARITY$ is not even in nonuniform $\mathbf{AC}^0$; i.e. it cannot be computed by any polynomial size bounded depth family of Boolean circuits. We will show using the student-teacher method outlined above that if $\mathbf{V}^0$ proves Σ_0^B-**REPL** then there is a randomized $\mathbf{AC}^0$ algorithm which on each input X, with probability at least one-half, correctly outputs $PARITY(X)$, and if it does not output $PARITY(X)$ it outputs an 'abort' message, meaning the computation failed. From this it follows using a standard argument that $PARITY$ is in nonuniform $\mathbf{AC}^0$. For each input length n, the circuit for computing $PARITY(X)$ for $|X| = n$ is obtained by repeating the randomized computation $n + 1$ times with indepentent random bits, to obtain a randomized $\mathbf{AC}^0$ algorithm that computes $PARITY(X)$ with abort probability at most 2^{-n-1}. Hence there must be some fixed setting of the random bits which aborts on at most a fraction 2^{-n-1} inputs X of length n; which means this setting of random bits allows the circuit to correctly compute $PARITY(X)$ on all inputs X of length n.

Let PAR be the function that maps a binary string of length m to its parity vector. That is, $PAR(m,X) = Y$ if $|Y| \leq m$ and, for each $i < m$, $Y(i)$ is the parity of the string $X(0) \dots X(i)$. In what follows we take m to be a parameter, assume X is a string of length at most m, and suppress the argument m from $PAR(m,X)$. Note that for fixed m, PAR is a bijection from the set of strings of length at most m to itself.

Although $PAR(X)$ cannot be computed in $\mathbf{AC}^0$, its inverse, which we will call $UNPAR$, is in (uniform) $\mathbf{FAC}^0$: the ith bit of $UNPAR(Y)$ is given by the Σ_0^B-formula

$$(i = 0 \wedge Y(i)) \vee (i > 0 \wedge Y(i-1) \oplus Y(i))$$

Here $UNPAR$ has an argument m, which we suppress. Then

$$UNPAR(PAR(X)) = X$$

and

$$PAR(UNPAR(Y)) = Y$$

Notice also that for all m-bit strings A, B, C, writing $\oplus$ for bitwise XOR, if $A = B \oplus C$ then $PAR(A) = PAR(B) \oplus PAR(C)$.

Assuming that $\overline{\mathbf{V}}^0$ proves Σ_0^B-**REPL** we can apply the argument of Section 8F.1 and assume that $\overline{\mathbf{V}}^0$ proves (208) for the case in which F is $UNPAR$. We can assume that the parameter m is coded by the parameter W; specifically $m = |W^{[0]}|$, where $W^{[0]}$ is a string of 0's except bit $m-1$ is 1. (Note that $PAR(W^{[0]}) = W^{[0]}$.) Also we define the terms $t = m+1$, and $u = m$. Then for some fixed k there is a uniform $\mathbf{AC}^0$ algorithm which, for any sequence $W^{[1]}, \dots, W^{[m]}$ of binary strings of length at most m makes k queries of the form "what is $PAR(W^{[i]})$?" and outputs the sequence of parity vectors of W.

Suppose $m \geq 2k$. We will show how to use this algorithm to compute the parity of a single string I, $|I| \leq m$, in uniform randomized $\mathbf{AC}^0$.

Choose m strings $U_1, \dots, U_m$ of m bits each uniformly at random, and for each i compute $V_i = UNPAR(U_i)$. Choose a number r, $1 \leq r \leq m$, uniformly at random. For $1 \leq i \leq m$ define $W^{[i]}$ by the condition

$$W^{[i]} = \begin{cases} V_i & \text{if } i \neq r \\ I \oplus V_r & \text{if } i = r. \end{cases}$$

Since for each m the function $UNPAR$ defines a bijection from the set $\{0,1\}^m$ to itself, and since for each I with $|I| < m$ the map $X \mapsto I \oplus X$ also defines a bijection from that set to itself, it follows that the string W defined above, interpreted as an $m \times m$ bit matrix, is uniformly distributed over all such matrices.

Now run our student-teacher $\mathbf{AC}^0$ algorithm on W. If the student asks "what is $PAR(W^{[i]})$?" for $i \neq r$, reply with U_i (or $W^{[0]}$ if $i = 0$) (which is the correct answer). If the algorithm queries "what is $PAR(Y^{[r]})$?", then abort the computation.

Since $PAR(W^{[i]})$ is queried for at most k different values of i and since for each input I each pair (W, r) is equally likely to have been chosen, it follows that the computation will be aborted with probability at most $k/m \leq 1/2$.

Hence with probability at least 1/2 the algorithm is not aborted, we are able to answer all the queries correctly, and we obtain Y such that $Y^{[r]} = PAR(W^{[r]}) = PAR(I \oplus V_r)$. But $I = V_r \oplus (I \oplus V_r)$ and hence

$$\begin{aligned} PAR(I) &= PAR(V_r) \oplus PAR(I \oplus V_r) \\ &= U_r \oplus Y^{[r]} \end{aligned}$$

We use this to compute $PAR(I)$ and use bit $m-1$ of $PAR(I)$ to determine $PARITY(I)$.

For each input I the algorithm succeeds with probability at least 1/2, where the probability is taken over its random input bits. If the algorithm aborts, this is reflected in the output. As explained earlier, this implies the existence of a nonuniform $\mathbf{AC}^0$ algorithm for $PARITY(I)$. Since no such algorithm exists, it follows that $\mathbf{V}^0$ does not prove the Σ_0^B-Replacement scheme. ⊣

We now show that **VPV** seems unlikely to prove Σ_0^B-**REPL** because a consequence would be that integer factoring is easy. This constrasts with $\mathbf{V}^1$, which proves the stronger $\mathbf{g}\Sigma_1^B$-**REPL** scheme (Corollary 6.24).

We adapt the proof [73] that cracking Rabin's cryptosystem based on squaring modulo N is as hard as factoring. Let N be the product of distinct odd primes P and Q. Suppose $0 < X_1 < N$ and $\gcd(X_1, N) = 1$. Let $C = X_1^2$. Then C has precisely four square roots X_1, X_2, X_3, X_4 modulo N. This can be seen as follows: let $X_P = (X_1 \bmod P)$ and $X_Q = (X_1 \bmod Q)$. By the Chinese remainder theorem there are uniquely determined numbers X_1, X_2, X_3, X_4 with $0 < X_i < N$ such that

$$\begin{array}{ll} X_1 \equiv X_P \pmod{P} & X_1 \equiv X_Q \pmod{Q} \\ X_2 \equiv X_P \pmod{P} & X_2 \equiv -X_Q \pmod{Q} \\ X_3 \equiv -X_P \pmod{P} & X_3 \equiv X_Q \pmod{Q} \\ X_4 \equiv -X_P \pmod{P} & X_4 \equiv -X_Q \pmod{Q} \end{array}$$

Now $X_1 - X_2 \equiv 0 \pmod{P}$ and $X_1 - X_2 \equiv 2X_Q \not\equiv 0 \pmod{Q}$, so $\gcd(X_1 - X_2, N) = P$. So from X_1 and X_2 we can recover P, and similarly from X_1 and X_3 we can recover Q.

Hence if we have one square root of C, and are then given a square root at random, we can factor N with probability $\frac{1}{2}$.

THEOREM 8.82 ([36]). *If* **VPV** *proves* Σ_0^B-**REPL** *then factoring (of products of two odd primes) is possible in probabilistic polynomial time.*

PROOF. We will argue as in the proof of the previous theorem, this time taking squaring modulo N as our function F (so F has N as a parameter).

If **VPV** proves Σ_0^B-**REPL** then there is polynomial time algorithm which, for some fixed k, given any sequence $W^{[0]}, \ldots, W^{[m-1]}$ of squares (modulo N) (where $m = |N|$), makes at most k queries of the form "what is the square root of $W^{[i]}$?" and, if these are answered correctly, outputs square roots of all the $W^{[i]}$s.

Now suppose N is large enough that $m = |n| > k$. Choose numbers $X_0, \ldots, X_{m-1}$ uniformly at random with $0 < X_i < N$. We may assume that $\gcd(X_i, N) = 1$ for all i, since otherwise we can immediately find a factor of N.

Choose W so that for each i, $W^{[i]} = (X_i^2 \bmod N)$. Notice that each X_i is distributed uniformly among the four square roots of $W^{[i]}$.

Run our algorithm, and to each query "what is the square root of $W^{[i]}$?", answer with X_i. We will get as output Y coding a sequence $Y^{[0]}, \ldots, Y^{[m-1]}$ of square roots of $W^{[0]}, \ldots, W^{[m-1]}$.

If we think of N as fixed, the value of Y depends only on the inputs given to the algorithm, namely W and the k many numbers X_i that we gave as replies. Let i be some index for which X_i was not used. Then X_i is distributed at random among the square roots of $W^{[i]}$, and $Y^{[i]}$ is a square root of $W^{[i]}$ that was chosen without using any information about which square root X_i is. Hence $\gcd(X_i - Y^{[i]}, N)$ is a factor of N with probability $\frac{1}{2}$. $\dashv$

8G. More on $\mathbf{V}^i$ and $\mathbf{TV}^i$

8G.1. Finite Axiomatizability. $\mathbf{V}^0$ is finitely axiomatizable by Theorem 5.76. By the discussion following Theorem 8.44 we know that $\mathbf{TV}^0$ is finitely axiomatizable, as are the $\forall\Sigma_1^B$-consequences of $\mathbf{V}^1$. Here we show that $\mathbf{V}^i$ and $\mathbf{TV}^i$ are finitely-axiomatizable for all $i \geq 0$. We start by proving the existence of a universal polynomial time function.

THEOREM 8.83 (Universal Function). *There is an* $\mathcal{L}_{\mathbf{FP}}$ *function*

$$Univ(X, W)$$

such that for every $\mathcal{L}_{\mathbf{FP}}$*-function* $F(X)$ *there is an* $\mathcal{L}_{\mathbf{FAC}^0}$*-function* $H_F(n)$ *such that*

$$\mathbf{VPV} \ \vdash \ |X| < n \supset F(X) = Univ(X, H_F(n))$$

In particular **VPV** *proves* $F(X) = Univ(X, H_F(|X|))$.

PROOF. We use the machinery of (179) (page 205). The value of $Univ(X, W)$ is the output of the circuit C described by W, where C

expects an input string of length at most n (specified by W), and (assuming $|X| < n$) $Univ(X, W)$ supplies X to the input gates of C. Then $H_F(n)$ describes a circuit which computes $F(X)$ for $|X| < n$. $\dashv$

To help prove the next result, we introduce a string pairing function.

Definition 8.84. $\langle X, Y\rangle$ *is the* $\mathcal{L}_{\mathbf{FAC}^0}$*-function defined by*

$$\langle X, Y\rangle(i) \leftrightarrow \exists j \le i,\ (i = \langle 0, j\rangle \wedge X(j)) \vee (i = \langle 1, j\rangle \wedge Y(j))$$

More generally $\langle X_1, \dots, X_n\rangle$ *is defined inductively by*

$$\langle X_1, \dots, X_{n+1}\rangle = \langle\langle X_1, \dots, X_n\rangle, X_{n+1}\rangle$$

Finally we define

$$\langle x_1, \dots, x_k, \vec{X}\rangle = \langle POW2(x_1), \dots, POW2(x_k), \vec{X}\rangle$$

Note that $\overline{\mathbf{V}}^0$ proves

$$\langle X, Y\rangle = Z \supset (X = Z^{[0]} \wedge Y = Z^{[1]})$$

Theorem 8.85. $\mathbf{V}^i$ *and* $\mathbf{TV}^i$ *are finitely axiomatizable for all* $i \ge 0$.

Proof. We have already proved this for $i = 0$. For the general case we start with the finitley axiomatizable theory **VP** and add one Σ_i^B-**COMP**-axiom to get $\mathbf{V}^i$ and one Σ_i^B-**SIND**-axiom to get $\mathbf{TV}^i$. The axioms in question involve universal formulas. For notational simplicity we treat the case $i = 1$; the general case will be clear.

For $\mathbf{V}^1$ we define the $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$ formula

$$UV(i, a, X, W) \equiv \exists Y \le a\, Univ(\langle i, X, Y\rangle, W)(0)$$

and let $UV'(i, a, X, W)$ be the equivalent Σ_1^B formula according to Theorem 8.32 (so $\mathbf{VPV} \vdash UV \leftrightarrow UV'$).

Let $\mathcal{T}$ be the finitely axiomatizable theory extending **VP** by the comprehension axiom for formula $UV'(i, a, X, W)$ (where a, X, W are parameters). Obviously $\mathcal{T} \subseteq \mathbf{V}^1$. To prove the reverse inclusion, since **VPV** is conservative over **VP**, it suffices to show $\mathbf{V}^1 \subseteq \mathcal{T} + \mathbf{VPV}$.

Let $\varphi(i, \vec{x}, \vec{X})$ be a Σ_1^B-formula. Then there is an $\mathcal{L}_{\mathbf{FP}}$-function F such that (using Theorem 8.83) **VPV** proves

$$\begin{aligned}\varphi(i, \vec{x}, \vec{X}) &\leftrightarrow \exists Y \le t\, F(\langle i, \langle \vec{x}, \vec{X}\rangle, Y\rangle)(0)\\ &\leftrightarrow \exists Y \le t\, Univ(\langle i, \langle \vec{x}, \vec{X}\rangle, Y\rangle, H_F(|\langle i, \langle \vec{x}, \vec{X}\rangle, Y\rangle|))(0)\\ &\leftrightarrow UV'(i, t, \langle \vec{x}, \vec{X}\rangle, H_F(|\langle i, \langle \vec{x}, \vec{X}\rangle, Y\rangle|))\end{aligned}$$

Hence **VPV** proves the comprehension for φ from the comprehension axiom for UV'. It follows that $\mathbf{V}^1 = \mathcal{T}$.

The argument is similar for $\mathbf{TV}^1$. This time we define

$$UT(a, X, Z, W) \equiv \exists Y \le a\, Univ(\langle X, Z, Y\rangle, W)(0)$$

and axiomatize $\mathbf{TV}^1$ by the string induction axiom for $UT'(X)$, where a, Z, W are parameters and UT' is a Σ_1^B-formula equivalent to UT. $\dashv$

Since $\mathbf{VP} \subseteq \mathbf{V}^1$ (Theorem 8.3) and $\mathbf{TV}^0 = \mathbf{VP}$ (Theorem 8.44) it follows that $\mathbf{TV}^0 \subseteq \mathbf{V}^1$. This is generalized in the following result.

Theorem 8.86. *For $i \geq 0$*

$$\mathbf{V}^i \subseteq \mathbf{TV}^i \subseteq \mathbf{V}^{i+1}$$

Proof. The first inclusion is Theorem 8.38. For the second inclusion, by definition 8.36 it suffices to show that $\mathbf{V}^{i+1}$ proves the Σ_i^B-**SIND** induction scheme

$$[\varphi(\varnothing) \wedge \forall X(\varphi(X) \supset \varphi(S(X))] \supset \varphi(Y)$$

where $\varphi(X)$ is a Σ_i^B-formula. Reasoning in $\mathbf{V}^{i+1}(\mathbf{VPV})$ (which by Theorem 8.27 is conservative over $\mathbf{V}^{i+1}$) assume

$$\varphi(\varnothing) \wedge \forall X(\varphi(X) \supset \varphi(S(X))) \tag{209}$$

Define the Π_{i+1}^B-formula

$$\psi(i) \equiv \forall Z \leq i \forall W \leq |Y| \big((|W+Z| \leq |Y| \wedge \varphi(W)) \supset \varphi(W+Z)\big)$$

By Corollary 6.4 applied to $\mathbf{V}^{i+1}$ we are justified in using number induction on $\psi(i)$. The base case $\psi(0)$ is easy since $\varphi(W) \supset \varphi(S(W))$ by assumption (209). The induction step $\psi(i) \supset \psi(i+1)$ is proved by using the hypothesis $\psi(i)$ twice, first with (W, Z) set to (W, Z') with $Z' = \lfloor \frac{1}{2} Z \rfloor$ and then with (W, Z) set to $(W+Z', Z')$ to infer $\varphi(W+2Z')$, from which $\psi(i+1)$ follows, (using the assumption (209)) if $2Z' \neq Z$).

Finally $\varphi(Y)$ follows from $\psi(|Y|+1)$ and $\varphi(\varnothing)$. ⊣

From the above theorem we have the hierarchy

$$\mathbf{V}^0 \subset \mathbf{TV}^0 \subseteq \mathbf{V}^1 \subseteq \mathbf{TV}^1 \subseteq \mathbf{V}^2 \subseteq \mathbf{TV}^2 \subseteq \mathbf{V}^{\mathbf{3}} \subseteq \dots$$

so the unions of $\mathbf{V}^i$ and $\mathbf{TV}^i$ are the same. We use the notation

$$\mathbf{V}^\infty = \bigcup_i \mathbf{V}^i = \bigcup_i \mathbf{TV}^i \tag{210}$$

The next result follows from Theorem 8.85 and compactness.

Corollary 8.87. *$\mathbf{V}^\infty$ is finitely axiomatizable iff $\mathbf{V}^\infty$ collapses to $\mathbf{V}^i$ or $\mathbf{TV}^i$ for some i.*

See Section 8G.3 for consequences of the collapse of $\mathbf{V}^\infty$.

8G.2. Definability in the $\mathbf{V}^\infty$ hierarchy. See Table 1 page 239 for a partial summary of the results in this section.

Recall that for $i \geq 1$, Σ_i^P is the set of (two-sorted) relations in level i of the polynomial hierarchy, and that these are precisely the relations represented by Σ_i^B-formulas (Theorem 4.19). Also $\mathbf{FP}^{\Sigma_i^P}$ is the set of functions computable by a polynomial time Turing machine with a Σ_i^P oracle. For $i = 0$, we will take $\mathbf{FP}^{\Sigma_0^P}$ to be simply $\mathbf{FP}$ (this is consistent with taking Σ_0^P to be either $\mathbf{P}$ or $\mathbf{AC}^0$). We will show that for $i \geq 0$, $\mathbf{FP}^{\Sigma_i^P}$ is the class of functions Σ_{i+1}^B-definable in $\mathbf{TV}^i$, and also in $\mathbf{V}^{i+1}$. (We have already shown this for $i = 0$.)

We start by generalizing the universal theory $\mathbf{VPV}$ to $\mathbf{VPV}^i$, for $i \geq 1$. Here $\mathbf{VPV}^1 = \mathbf{VPV}$, and for $i \geq 0$ $\mathbf{VPV}^{i+1}$ has function symbols for all functions in $\mathbf{FP}^{\Sigma_i^P}$. We use $\mathcal{L}_{\mathbf{FP}^i}$ to denote the vocabulary of $\mathbf{VPV}^i$.

Since $\mathcal{L}_{\mathbf{FP}^i}$ includes the vocabulary $\mathcal{L}_{\mathbf{FAC}^0}$ of $\overline{\mathbf{V}}^0$, it includes symbols for the string functions $\varnothing, S, +$ defined using Σ_0^B-formulas in Example 5.42. Since we want the theories $\mathbf{VPV}^i$ to be universal we take the defining axioms for these functions to be the quantifier-free axioms for these functions in $\overline{\mathbf{V}}^0$. Also for the present purposes string ordering $X \leq Y$ as given in Definition 8.39 is replaced by its equivalent quantifier-free definition in $\overline{\mathbf{V}}^0$. See Example 8.46 for the definition of $POW2(x)$.

We need functions to witness bounded existential string quantifiers, just as $f_{\varphi(z),t}$ as defined in (86) is used to witness bounded existential number quantifiers. Thus let $G_{\varphi,t}(\vec{x},\vec{X})$ be the least Y with $|Y| < t(\vec{x},\vec{X})$ such that $\varphi(\vec{x},\vec{X},Y)$ holds, or $POW2(t)$ if there is no such Y. Then $G_{\varphi,t}$ has defining axiom (suppressing $\vec{x},\vec{X}$)

(211)
$$G_{\varphi,t} \leq POW2(t) \wedge \big(G_{\varphi,t} < POW2(t) \supset \varphi(G_{\varphi,t})\big) \wedge \big(Y < G_{\varphi,t} \supset \neg\varphi(Y)\big)$$

The definition of vocabularies $\mathcal{L}_{\mathbf{FP}^i}$ is similar to Definition 8.17 for $\mathcal{L}_{\mathbf{FP}}$.

DEFINITION 8.88. *The vocabularies $\mathcal{L}_{\mathbf{FP}^1} \subset \mathcal{L}_{\mathbf{FP}^2} \subset \dots$ are defined as follows.*

(i) $\mathcal{L}_{\mathbf{FP}^1} = \mathcal{L}_{\mathbf{FP}}$.
(ii) *For $i \geq 1$ $\mathcal{L}_{\mathbf{FP}^{i+1}}$ is the smallest set that satisfies*
   (1) $\mathcal{L}_{\mathbf{FP}^{i+1}} \supseteq \mathcal{L}_{\mathbf{FP}^i}$.
   (2) *For each open formula $\varphi(\vec{x},\vec{X},Y)$ over $\mathcal{L}_{\mathbf{FP}^i}$ and term $t(\vec{x},\vec{X})$ of $\mathcal{L}_A^2$ there is a string function $G_{\varphi,t}$ in $\mathcal{L}_{\mathbf{FP}^{i+1}}$.*
   (3) *For each open formula $\varphi(z,\vec{x},\vec{X})$ over $\mathcal{L}_{\mathbf{FP}^{i+1}}$ and term $t = t(\vec{x},\vec{X})$ of $\mathcal{L}_A^2$ there is a string function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FP}^{i+1}}$.*
   (4) *For each triple G, H, t, where $G(\vec{x},\vec{X})$ and $H(y,\vec{x},\vec{X},Z)$ are functions in $\mathcal{L}_{\mathbf{FP}^{i+1}}$ and $t = t(y,\vec{x},\vec{X})$ is a term in $\mathcal{L}_A^2$, there is a function $F_{G,H,t}$ in $\mathcal{L}_{\mathbf{FP}^{i+1}}$.*

DEFINITION 8.89. *For $i \geq 1$ the universal theory $\mathbf{VPV}^i$ has vocabulary $\mathcal{L}_{\mathbf{FP}^i}$ and (i) $\mathbf{VPV}^1 = \mathbf{VPV}$ and (ii) for $i \geq 1$ $\mathbf{VPV}^{i+1}$ contains $\mathbf{VPV}^i$ and has as (sometimes) additional defining axioms* (211) *for each function $G_{\varphi,t}$ in $\mathcal{L}_{\mathbf{FP}^{i+1}}$ and* (176) *for each function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FP}^{i+1}}$ and* (174), (175) *(page 200) for each function $F_{G,H,t}$ in $\mathcal{L}_{\mathbf{FP}^{i+1}}$.*

LEMMA 8.90. *For all $i \geq 0$ for every Σ_i^B-formula ψ there is an open formula φ over $\mathcal{L}_{\mathbf{FP}^{i+1}}$ such that $\mathbf{VPV}^{i+1}$ proves $\psi \leftrightarrow \varphi$.*

PROOF. We use induction on i. For $i = 0$ this is clear. Now suppose $i > 0$ and ψ is a Σ_i^B-formula. Then we may assume that $\psi \equiv \exists Y < t\,\eta(Y)$, where η is Σ_{i-1}^B. By the induction hypothesis there is an open formula φ over $\mathcal{L}_{\mathbf{FP}^i}$ such that $\mathbf{VPV}^i$ proves $\eta \leftrightarrow \varphi$. Then $\mathbf{VPV}^{i+1}$ proves

$$\varphi(G_{\varphi,t}) \leftrightarrow \exists Y < t\varphi(Y) \leftrightarrow \psi$$

so $\varphi(G_{\varphi,t})$ satisfies the Lemma. ⊣

THEOREM 8.91. *For $i \geq 0$ the string function symbols F of $\mathcal{L}_{\mathbf{FP}^{i+1}}$ represent precisely the string functions in $\mathbf{FP}^{\Sigma_i^P}$ and terms of the form $|F(\vec{x}, \vec{X})|$ represent precisely the number functions of $\mathbf{FP}^{\Sigma_i^P}$.*

PROOF. The part about number functions follows from the part about string functions, so we prove the latter. We use induction in i. For $i = 0$ this was observed when introducing $\mathcal{L}_{\mathbf{FP}}$. For the induction step the proof is similar to the proof of Cobham's Theorem. To see that every string function in $\mathcal{L}_{\mathbf{FP}^{i+1}}$ is in $\mathbf{FP}^{\Sigma_i^P}$ it suffices to show that this is true for each of the cases in part (ii) of Definition 8.88. For 3) and 4) this is true because the functions computable by a polynomial time Turing machine with a Σ_i^P oracle are closed under composition and limited recursion, and such a machine can evaluate an open formula whose functions are so computable. For 2), observe that such a machine can query its Σ_i^P oracle to find out for $W \leq \mathit{POW2}(t)$ whether there is $Y \leq W$ satisfying $\varphi(\vec{x}, \vec{X}, Y)$, and hence use binary search to find the least such Y (if any).

Conversely, to see that every string function in $\mathbf{FP}^{\Sigma_i^P}$ is represented by a function symbol in $\mathcal{L}_{\mathbf{FP}^{i+1}}$, use limited recursion to define a function like $\mathit{Conf}_{\mathsf{M}}$ in the proof of Cobham's Theorem 6.16 to compute the configurations of the oracle Turing machine, where now the Σ_i^P oracle queries are answered with the help of open formulas in Lemma 8.90. Now the value of F can be extracted using the output function $\mathit{Out}_{\mathsf{M}}$ as in (99), so $F(\vec{x}, \vec{X}) = T(\vec{x}, \vec{X})$ for some term T of $\mathcal{L}_{\mathbf{FP}^{i+1}}$. Then $F \equiv F_{\varphi(z),t}$ where $\varphi(z) \equiv T(z)$ and t is a bounding term for F. ⊣

The next result generalizes Theorem 8.23.

THEOREM 8.92. *For $i \geq 1$ $\mathbf{VPV}^i$ proves the $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}^i})$-$\mathbf{COMP}$, $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}^i})$-$\mathbf{IND}$, $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}^i})$-$\mathbf{MIN}$, and $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}^i})$-$\mathbf{MAX}$ axiom schemes.*

PROOF. By Corollary 5.8 it suffices to prove this for the case of $\mathbf{COMP}$. For every $\Sigma_0^B(\mathcal{L}_{\mathbf{FP}^i})$-formula φ there is an open $\mathcal{L}_{\mathbf{FP}^i}$-formula φ^+ such that $\mathbf{VPV}^i$ proves $\varphi \leftrightarrow \varphi^+$ (see the proof of Lemma 5.70). The function $F_{\varphi,y}$ is easily used to prove the comprehension axiom for an open formula φ. ⊣

THEOREM 8.93. *For $i \geq 0$, every function in $\mathcal{L}_{\mathbf{FP}^{i+1}}$ is Σ_{i+1}^B-definable in $\mathbf{TV}^i$, and $\mathbf{VPV}^{i+1}$ is a conservative extension of $\mathbf{TV}^i$.*

PROOF. For $i = 0$ this follows from Theorems 8.27 a), 8.33, 8.44, and Corollary 8.34.

In general we show $\mathbf{VPV}^{i+1}$ extends $\mathbf{TV}^i$, by showing $\mathbf{VPV}^{i+1}$ proves the Σ_i^B-**SIND** axioms (183). By Lemma 8.90 we may assume that φ is an open $\mathcal{L}_{\mathbf{FP}^{i+1}}$-formula. Now proceed exactly as in the proof of Theorem 8.48, replacing $\mathbf{VPV}$ by $\mathbf{VPV}^{i+1}$.

To show that the extension is conservative, and to show that every $\mathcal{L}_{\mathbf{FP}^{i+1}}$-function is Σ_{i+1}^B-definable in $\mathbf{TV}^i$, by Theorem 8.91 it suffices to show that all functions in $\mathbf{FP}^{\Sigma_i^P}$ are Σ_{i+1}^B-definable in $\mathbf{TV}^i$, and (for conservativity) that this can be done in such a way that the defining axioms for the $\mathcal{L}_{\mathbf{FP}^{i+1}}$-functions are provable. We omit the latter (which amounts to formalizing in $\mathbf{TV}^i$ part of the proof of Theorem 8.91), and concentrate on the former.

Let $F(\vec{x}, \vec{X})$ be a function in $\mathbf{FP}^{\Sigma_i^P}$, where $i \geq 1$. Then some polynomial time oracle Turing machine M computes F using an oracle $\varphi(W)$, where φ is (represented by) a Σ_i^B-formula.

Let $t = t(\vec{x}, \vec{X})$ be a suitable $\mathcal{L}_A^2$ bounding term and let

$$Comp_{\mathsf{M}}(\vec{x}, \vec{X}, U, W, Z)$$

be a Σ_0^B-formula which asserts that U codes a computation of M on input $\vec{x}, \vec{X}$ where for all $i < t$, $W^{[i]}$ is the ith oracle query (if any) and $Z(t \dot{-} i)$ is the answer to this query.

Define $\psi(\vec{x}, \vec{X}, Z, Y)$ to be

$$\exists U \leq t \exists W \leq t$$

$$Comp_{\mathsf{M}}(\vec{x}, \vec{X}, U, W, Z) \wedge Y = Ans_{\mathsf{M}}(U) \wedge \forall i < t(Z(t \dot{-} i) \supset \varphi(W^{[i]}))$$

where $Ans_{\mathsf{M}}(U)$ is the output of the computation coded by U. Let $\psi'(\vec{x}, \vec{X}, Z)$ be $\exists Y < t\psi(\vec{x}, \vec{X}, Z, Y)$. Then ψ' is a $\mathbf{g}\Sigma_i^B$-formula, which $\mathbf{TV}^i$ proves equivalent to a Σ_i^B-formula by the Replacement scheme (Corollary 6.24). Note that if $\psi'(\vec{x}, \vec{X}, Z)$ holds then the 'true' query answers coded by Z must be correct, but 'false' answers may not be correct. However the largest Z satisfying ψ' must code all correct answers, since if the ith query is the first incorrect answer then changing $Z(t \dot{-} i)$ from 'false' to 'true' would increase Z no matter how the subsequent answers are changed.

Since $\mathbf{TV}^i$ proves the Σ_i^B-**SMAX** axioms (Theorem 8.42) and **VPV** proves $\psi'(\vec{x}, \vec{X}, \varnothing)$ it follows that $\mathbf{TV}^i$ proves the existence of a largest Z, $|Z| < t$, satisfying $\psi'(\vec{x}, \vec{X}, Z)$.

Thus we may use the following definition for $F(\vec{x}, \vec{X})$.

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \exists Z < t\big(\psi(\vec{x}, \vec{X}, Z, Y) \wedge \forall Z' < t(Z < Z' \supset \neg\psi'(\vec{x}, \vec{X}, Z'))\big)$$

The formula $\eta(\vec{x}, \vec{X}, Y)$ on the RHS is equivalent to a Σ_{i+1}^B-formula. Also $\mathbf{TV}^i$ proves the existence of a largest $Z < t$ satisfying $\psi'(\vec{x}, \vec{X}, Z)$

and hence the existence of Y satisfying $\eta(\vec{x}, \vec{X}, Y)$. Finally $\mathbf{V}^0$ (and hence $\mathbf{TV}^i$) proves the uniqueness of Y, since obviously there is at most one largest Z satisfying ψ', and by Σ_0^B-**IND** this Z uniquely determines U, W in $Comp_{\mathsf{M}}$ and hence Y. ⊣

Theorem 8.94. *For $i \geq 0$ the following are equivalent for a string function F:*

(i) *F is in $\mathbf{FP}^{\Sigma_i^P}$.*
(ii) *F is Σ_{i+1}^B-definable in $\mathbf{TV}^i$.*
(iii) *F is Σ_{i+1}^B-definable in $\mathbf{V}^{i+1}$.*
(iv) *F is Σ_{i+1}^B-definable in $\mathbf{VPV}^{i+1}$.*
(v) *F is $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}^{i+1}})$-definable in $\mathbf{VPV}^{i+1}$.*

Similarly for a number function f.

Proof. (i) $\Longrightarrow$ (ii) by Theorems 8.91 and 8.93. (ii) $\Longrightarrow$ (iii) by Theorem 8.3. (iii) $\Longrightarrow$ (ii) by Theorem 8.95. (ii) $\Longrightarrow$ (iv) by Theorem 8.93. (iv) $\Longrightarrow$ (v) by Lemma 8.90. (v) $\Longrightarrow$ (i) by Theorems 8.20 and 8.91. ⊣

Theorem 8.95. *For $i \geq 0$ $\mathbf{V}^{i+1}$ is Σ_{i+1}^B-conservative over $\mathbf{TV}^i$.*

Proof. By Lemma 8.90 every Σ_{i+1}^B-formula φ is provably equivalent in $\mathbf{VPV}^{i+1}$ to a $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}^{i+1}})$-formula φ'. Thus if $\mathbf{V}^{i+1}$ proves φ then $\mathbf{V}^{i+1} + \mathbf{VPV}^{i+1}$ proves φ', and, arguing as in the proof of Theorem 8.29, $\mathbf{VPV}^{i+1}$ proves that φ' can be witnessed by functions in $\mathcal{L}_{\mathbf{FP}^{i+1}}$. Thus $\mathbf{VPV}^{i+1}$ proves φ' and φ, and so $\mathbf{TV}^i$ proves φ by Theorem 8.93. ⊣

The next result generalizes Theorem 8.76. We define a $\mathbf{PLS}^{\Sigma_i^P}$ problem Q to be the same as in Definition 8.69 except now the relation φ_Q and the functions N_Q and P_Q are allowed to be polynomial time with a Σ_i^P-oracle.

Theorem 8.96. *For $i \geq 1$ a search problem Q is Σ_i^B-definable in $\mathbf{TV}^i$ iff $Q \leq_{\mathbf{AC}^0} Q'$ for some $\mathbf{PLS}^{\Sigma_{i-1}^P}$ problem Q'.*

Proof. The proof is very similar to the proof of Theorem 8.76. Instead of an $\mathbf{AC}^0$-**ITERATION** problem we need a $\mathbf{FP}^{\Sigma_{i-1}^P}$-**ITERATION** problem, in which the function F is allowed to be in $\mathbf{FP}^{\Sigma_{i-1}^P}$. To see that every $\mathbf{PLS}^{\Sigma_{i-1}^P}$ problem is many-one reducible to some $\mathbf{FP}^{\Sigma_{i-1}^P}$-**ITERATION** problem, we need to slightly alter the proof of Theorem 8.72. The difficulty in that proof is that the reducing function G is defined by $G(U *_t V) = N_Q(U)$, where the neighborhood function N_Q is now allowed to be in $\mathbf{FP}^{\Sigma_{i-1}^P}$ instead of in $\mathbf{FAC}^0$. To fix this, we change the iterating function F in the proof to a function F'. The idea behind F was to let its domain be concatinations $U *_t V$ where U is a candidate solution for Q and V is its profit. The idea behind F' is that its domain consists of concatenations $U *_t W *_t V$ where U and V are as before, and $W = N_Q(U)$. Now we can define the reducing function G by $G(U *_t W *_t V) = W$.

EXERCISE 8.97. *Work out the details in the definition of F'.*

Continuing the proof of Theorem 8.96, it remains to generalize the witnessing theorem 8.77 so that the assumption is

$$\mathbf{TV}^i \vdash \exists Z \varphi(\vec{x}, \vec{X}, Z)$$

where now φ is Π^B_{i-1} and Q_F is a $\mathbf{FP}^{\Sigma^P_{i-1}}$-**ITERATION**-problem. By Theorems 8.93 and 8.94 it suffices to replace $\mathbf{TV}^i$ by $\mathbf{TV}^i + \mathbf{VPV}^i$ and φ by an open formula in $\mathcal{L}_{\mathbf{FP}^i}$, and $\overline{\mathbf{V}}^0$ by $\mathbf{VPV}^i$. The proof is a straightforward modification of the proof of Theorem 8.77, where now the string induction rule (196) applies to $\Sigma^B_1(\mathcal{L}_{\mathbf{FP}^i})$-formulas A. ⊣

The previous results characterize the search problems Σ^B_i-definable in $\mathbf{V}^j$ and $\mathbf{TV}^j$ when $i = j$ and sometimes when i and j differ by one. In order to specify these search problems for more general i and j we need to define a generalization of oracle Turing machines.

DEFINITION 8.98. *A* witness query Y *to an oracle* $\exists W \le t\, R(Y, W)$ *returns a witness* $W \le t$ *satisfying* $R(Y, W)$ *if such exists, and otherwise returns "NO". For* $i \ge 1$, $\mathbf{FP}^{\Sigma^P_i}[wit, O(g(n)]$ *is the class of search problems* Q *solvable by a polynomial time Turing machine that makes* $O(g(n)$ *witness queries to a* Σ^P_i *oracle* $\exists W \le t\, R(Y, W)$ *for* $R \in \Pi^P_{i-1}$.

Notice that a witness query can be simulated by polynomial many Boolean queries, using binary search. However as far as we know, the class $\mathbf{FP}^{\Sigma^P_i}[wit, O(g(n)]$ cannot be specified in terms of Boolean queries when $g(n)$ is $O(\log n)$.

THEOREM 8.99. (i) *For* $i \ge 1$*, a search problem* Q *is* Σ^B_{i+1}*-definable in* $\mathbf{V}^i$ *iff* $Q \in \mathbf{FP}^{\Sigma^P_i}[wit, O(\log n)]$.

(ii) *For* $j \ge 2$ *and* $\mathbf{V}^0 \subseteq \mathcal{T} \subseteq \mathbf{TV}^{j-2}$*, a search problem* Q *is* Σ^B_j*-definable in* $\mathcal{T}$ *iff* $Q \in \mathbf{FP}^{\Sigma^P_{j-1}}[wit, O(1)]$.

PROOF OF (II). By Theorem 8.93 $\mathbf{VPV}^{j-1}$ extends $\mathbf{TV}^{j-2}$ and hence the 'only if' direction is an easy consequence of the KPT Witnessing Theorem (see Exercise 8.101 below).

For the 'if' direction it suffices to show that for $j \ge 2$, $\mathbf{V}^0$ can Σ^B_j-define every search problem Q in $\mathbf{FP}^{\Sigma^P_{j-1}}[wit, O(1)]$. For concreteness we show this for $j = 2$; the general argument is essentially the same.

Let M be a polytime Turing machine which solves $Q(\vec{x}, \vec{X})$ by making at most a constant q number of queries to a Σ^P_1 witness oracle represented by a Σ^B_1-formula

$$\psi(Y) \equiv \exists W < t\, \eta(Y, W)$$

	$\mathbf{V}^0$	$\mathbf{TV}^0$	$\mathbf{V}^1$	$\mathbf{TV}^1$
Σ_1^B	$\mathbf{FAC}^0$	$\mathbf{FP}$	$\mathbf{FP}$	$\mathbf{CC}(\mathbf{PLS})$
Σ_2^B	$\mathbf{FP}^{\Sigma_1^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\mathbf{NP}}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\mathbf{NP}}[wit, \mathcal{O}(\log n)]$	$\mathbf{FP}^{\mathbf{NP}}$
Σ_3^B	$\mathbf{FP}^{\Sigma_2^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_2^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_2^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_2^P}[wit, \mathcal{O}(1)]$
Σ_4^B	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(1)]$
Σ_5^B	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$

	$\mathbf{V}^2$	$\mathbf{TV}^2$	$\mathbf{V}^3$	$\mathbf{TV}^3$
Σ_1^B	$\mathbf{CC}(\mathbf{PLS})$			
Σ_2^B	$\mathbf{FP}^{\mathbf{NP}}$	$\mathbf{CC}(\mathbf{PLS})^{\mathbf{NP}}$	$\mathbf{CC}(\mathbf{PLS})^{\mathbf{NP}}$	
Σ_3^B	$\mathbf{FP}^{\Sigma_2^P}[wit, \mathcal{O}(\log n)]$	$\mathbf{FP}^{\Sigma_2^P}$	$\mathbf{FP}^{\Sigma_2^P}$	$\mathbf{CC}(\mathbf{PLS})^{\Sigma_2^P}$
Σ_4^B	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_3^P}[wit, \mathcal{O}(\log n)]$	$\mathbf{FP}^{\Sigma_3^P}$
Σ_5^B	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$	$\mathbf{FP}^{\Sigma_4^P}[wit, \mathcal{O}(1)]$

Table 1. Definable search problems.

where η is in Σ_0^B. It is straightforward to give a Σ_2^B-formula $\varphi(\vec{x}, \vec{X}, Z)$ which asserts that there is some computation C of M on input $\vec{x}, \vec{X}$ with correct answers to all oracle queries such that C outputs Z. However it is more difficult to find such a formula such that $\mathbf{V}^0$ proves $\exists Z\varphi$.

To do this we first observe that we can find a machine M' which is equivalent to M but such that M' makes all of its queries in parallel, so that the answer to a query does not depend on the witness answer to any other query. The machine M' makes a witness query for each of the 2^q binary strings of length q, asking whether there is an apparently-corrrect computation of M on input $\vec{x}, \vec{X}$ such that the YES-NO answers to the $\leq q$ queries correspond to the bits of the string (or an initial segment). Here 'apparently-correct' means that for each query Y to $\psi(Y)$ a YES answer must be supplied with a witness W satisfying $\eta(Y, W)$, although NO answers need not be verified. Each YES answer to a query to M' must include a witness which codes an apparently-correct computation of M. From these witnesses M' can find a truely correct computation C of M where no initial segment S of C ending in a NO answer coincides with an initial segment S' of another computation except S' ends in a YES answer to the same query.

The parallel queries made by M' all have the form

$$\psi'(Y) \equiv \exists W < t\, \eta'(\vec{x}, \vec{X}, Y, W)$$

where η' is Σ_0^B-formula and Y is simply a bit string of length q. Now we describe a Σ_2^B-formula $\alpha_{\mathsf{M}}(\vec{x}, \vec{X}, Z)$ which asserts that Z is a possible output of a correct computation of M' on input $\vec{x}, \vec{X}$, and such that $\mathbf{V}^0$ proves $\exists Z \alpha_{\mathsf{M}}$. It suffices to describe $\alpha_{\mathsf{M}}(Z)$ as a disjunction of two Σ_2^B-formulas $\alpha^1_{\mathsf{M}}(Z) \vee \alpha^2_{\mathsf{M}}$, where $\alpha^1_{\mathsf{M}}(Z)$ makes the assertion as just stated and α^2_{M} is false.

Let $ApCor(\vec{x}, \vec{X}, C)$ be a Σ_0^B-formula which asserts that C codes an apparently-correct computation of M' on input $\vec{x}, \vec{X}$. Then $\alpha^1_{\mathsf{M}}(Z)$ asserts the intended meaning of $\alpha_{\mathsf{M}}(Z)$ in the obvious way:

$$\alpha^1_{\mathsf{M}}(Z) \;\equiv\; \exists C \forall W,\, ApCor(C) \wedge \neg\, Wit(W, C) \wedge Z = Out(C)$$

where we have omitted the bounds on the quantifiers and suppressed the arguments $\vec{x}, \vec{X}$, and $Wit(W, C)$ is a Σ_0^B-formula asserting that W is a witness for some query in C which was (incorrectly) answered NO, and $Out(C)$ is the output of the computation C.

Presumably $\mathbf{V}^0$ does not prove $\exists Z \alpha^1_{\mathsf{M}}(Z)$, so we need the false disjunct α^2_{M}, which asserts that there is no correct computation of M' on input $\vec{x}, \vec{X}$. Specifically α^2_{M} is a Σ_2^B-formulas which asserts that there exists a sequence $W_1, \ldots, W_\ell$ of potential witnesses to the $\ell = 2^q$ parallel queries made by a computation of M' such that for all apparently-correct computations C, one of the NO queries in C is in fact witnessed by some W_i. It suffiece to prove the following:

Claim. $\mathbf{V}^0$ proves $\exists Z \big(\alpha^1_{\mathsf{M}}(Z) \vee \alpha^2_{\mathsf{M}}\big)$.
Reasoning in $\mathbf{V}^0$, if α^2_{M} then (since Z does not occur in α^2_{M}) we can conclude $\exists Z\, \alpha^2_{\mathsf{M}}$ and we are done.

So assume $\neg\alpha^2_{\mathsf{M}}$. For each of the $\ell = 2^q$ queries Y_i (which are simply strings of length q), if $\exists W \eta'(Y_i, W)$ then let W_i be a witness satisfying $\eta'(Y_i, W_i)$; otherwise let $W_i = \varnothing$. Since $\neg\alpha^2_{\mathsf{M}}$ there exists an apparently-correct computation C such that none of the NO queries in C is witnessed by any W_i. But by the way we chose W_i, this means that all NO queries are correct, and hence C is a correct computation. Let $Z = Out(C)$. Then $\alpha^1_{\mathsf{M}}(Z)$. Hence $\exists Z\, \alpha_{\mathsf{M}}(Z)$. ⊣

EXERCISE 8.100. *Explain what goes wrong if we try to extend the above proof to the case that* M *makes more than a constant number of witness oracle queries.*

PROOF OF (I). The ‘only if’ direction can be proved using the same method as for Theorem 6.28 (witnessing for $\mathbf{V}^1$); see [54, 55] for details. For the proof of the ‘if’ direction let M be a polytime Turing machine which solves $Q(\vec{x}, \vec{X})$ by making $O(\log n)$ queries to a witness oracle represented by a formula $\psi(Y) \equiv \exists W < t\, \eta(Y, W)$, where η is in Π^B_{i-1}. It is easy to see that there is a Σ^B_{i+1}-formula $\varphi(\vec{x}, \vec{X}, Z)$ which asserts that $Z \in Q(\vec{x}, \vec{X})$ by asserting that there is a computation of M on input $\vec{x}, \vec{X}$ with output Z such that if $Y^{[i]}$ codes the ith query and $W^{[i]}$ codes the ith answer, then either $\eta(Y^{[i]}, W^{[i]})$ or $\neg\psi(Y^{[i]})$ and $W^{[i]} = $ ‘NO’.

In order to show that $\mathbf{V}^i$ proves $\exists Z\varphi(\vec{x}, \vec{X}, Z)$ we use the fact that $\mathbf{V}^i$ proves the Σ_i^B-**MAX** axiom scheme (Corollary 6.4) and argue as in the proof of Theorem 8.93. Thus $\mathbf{V}^i$ proves there is a largest $n < t$ (for suitable t) satisfying the Σ_i^B-formula $\alpha(n, \vec{x}, \vec{X})$, which asserts that there exists a computation of M as above and there exists the query sequence Y and answer sequence W such that the bits of the reverse binary notation for n code the Boolean answers to the successive queries of the computation, and for all i, if the ith query answer is positive, then $\eta(Y^{[i]}, W^{[i]})$. $\dashv$

Exercise 8.101. *Show using the KPT Witnessing Theorem 8.80 that for $i \geq 1$ if a search problem Q is Σ_{i+1}^B-definable in $\mathbf{VPV}^i$ then Q is in $\mathbf{FP}^{\Sigma_i^P}[wit, O(1)]$.*

8G.3. Collapse of $\mathbf{V}^\infty$ vs collapse of PH. It is an open question whether $\mathbf{V}^\infty$ (the union of the theories $\mathbf{V}^i$) collapses to some particular $\mathbf{V}^i$. Since each $\mathbf{V}^i$ is finitely axiomatizable, this question is equivalent to asking whether $\mathbf{V}^\infty$ is finitely axiomatizable. As far as we know it is possible that the polynomial hierarchy **PH** could collapse without $\mathbf{V}^\infty$ collapsing. (For example there might be a polynomial time algorithm for propositional satisfiability whose correctness is not provable in $\mathbf{V}^\infty$.) However if some $\mathbf{V}^i$ proves that **PH** collapses, then $\mathbf{V}^\infty$ collapses to $\mathbf{V}^i$. That is, if for every Σ_{i+1}^B-formula φ there is a Σ_i^B-formula φ' such that $\mathbf{V}^i$ proves $\varphi \leftrightarrow \varphi'$, then $\mathbf{V}^i$ proves Σ_{i+1}^B-**COMP**, so $\mathbf{V}^{i+1} = \mathbf{V}^i$. But the same assumption shows that $\mathbf{V}^{i+2} = \mathbf{V}^{i+1}$, and so on, so $\mathbf{V}^\infty = \mathbf{V}^i$.

The following theorem is an application of KPT Witnessing, and shows that the converse also holds: if $\mathbf{V}^\infty$ collapses to $\mathbf{V}^i$ then $\mathbf{V}^i$ proves that **PH** collapses. (This would be obvious if a function is in $\mathbf{FP}^{\Sigma_{i-1}^P}$ iff it is Σ_1^B-definable in $\mathbf{V}^i$, as opposed to Σ_i^B-definable in $\mathbf{V}^i$ as stated in Theorem 8.94.)

Theorem 8.102 ([58, 18, 85]). *For $i \geq 0$ if $\mathbf{TV}^i = \mathbf{V}^{i+1}$ then $\mathbf{TV}^i = \mathbf{V}^\infty$ and $\Sigma_{i+2} = \Pi_{i+2} = \mathbf{PH}$ and $\mathbf{TV}^i$ proves $\Sigma_{i+3} = \Pi_{i+3} = \mathbf{PH}$.*

Partial proof. For readability we treat the case $i = 0$; the general case is similar. (See the remark at the end of this proof.) Assuming $\mathbf{TV}^0 = \mathbf{V}^1$ we show that **PH** collapses to $\mathbf{P}/poly$ and provably collapses to $\mathbf{NP}/poly$, where *poly* refers to polynomial "advice", as explained below. It follows from the methods of Karp and Lipton [50] that **PH** collapses to $\Sigma_2^P = \Pi_2^P$ and provably collapses to $\Sigma_3^P = \Pi_3^P$. The proof that $\mathbf{TV}^0 = \mathbf{V}^1$ implies $\mathbf{TV}^0 = \mathbf{V}^\infty$ can be obtained from [18].

Since **VPV** is a conservative extension of $\mathbf{TV}^0$ (Theorem 8.93) and $\mathbf{V}^1(\mathbf{VPV}) = \mathbf{V}^1 + \mathbf{VPV}$ (Section 8B.1) our assumption $\mathbf{TV}^0 = \mathbf{V}^1$ is equivalent to $\mathbf{VPV} = \mathbf{V}^1(\mathbf{VPV})$.

The assertion "Every sequence $\alpha_1, \ldots, \alpha_m$ of propositional formulas has an initial sequence of maximal length ℓ of satisfiable formulas" is expressible by a formula

$$\psi \equiv \forall X \exists Y \forall Z \varphi(X, Y, Z)$$

where φ is an open $\mathcal{L}_{\mathbf{FP}}$-formula. Here X codes the sequence $\alpha_1, \ldots, \alpha_m$ and φ asserts that Y codes a sequence of satisfying assignments to the the first ℓ formulas of X for some $\ell \leq m$, and also that if $\ell < m$ then Z codes an assignment which falsifies $\alpha_{\ell+1}$.

$\mathbf{V}^1(\mathbf{VPV})$ proves ψ by applying the Σ_1^B-**MAX** axioms (Corollary 8.28) to the $\Sigma_1^B(\mathcal{L}_{\mathbf{FP}})$-formula expressing the condition that the first ℓ formulas coded by X are satisfiable. Hence by our assumption, **VPV** proves ψ, so by the KPT Witnessing Theorem there are polytime functions $F_1, \ldots, F_k$ such that **VPV** proves

(212)
$$\varphi(X, F_1(X), Z_1) \vee \varphi(X, F_2(X, Z_1), Z_2) \vee \cdots \vee \varphi(X, F_k(X, Z_1, \ldots, Z_{k-1}), Z_k)$$

Note that each function F_i plays the role of Y, and hence should code a sequence of assignments satisfying some initial segment of the formulas coded by X. From these functions $F_1, \ldots, F_k$ we obtain polytime functions $G_1, \ldots, G_k$ such that **VPV** proves that for every sequence $\alpha_1, \ldots, \alpha_k$ of propositional formulas with satisfying assignments $Z'_1, \ldots, Z'_k$ there is an i, $1 \leq i \leq k$, such that $G_i(X, Z'_1, \ldots Z'_{i-1})$ codes a satisfying assignment for α_i (we say that G_i 'wins' in this case).

The algorithm for evaluating $G_1(X)$ proceeds by computing $W_1 = F_1(X)$. If the sequence W_1 begins with an assignment satisfying α_1, then $G_1(X)$ is set to that assignment, so G_1 wins. Otherwise the algorithm for $G_2(X, Z'_1)$ sets $Z_1 = Z'_1$, so the first disjunct $\varphi(F_1(X), Z_1)$ in (212) is false (since by assumption Z'_1 satisfies α_1). Now the G_2 algorithm computes $W_2 = F_2(X, Z_1)$. If W_2 includes an assignment satisfying α_2, then $G_2(X, Z'_1)$ is set to that assignment, and G_2 wins. Otherwise the algorithm for $G_3(X, Z'_1, Z'_2)$ sets Z_2 to either Z'_1 or Z'_2 depending on $F_2(X, Z_1)$, so that the second conjunct $\varphi(X, F_2(X, Z_1), Z_2)$ in (212) is false. Then G_3 is set to an assignment in $F_3(X, Z_1, Z_2)$ which satisfies α_3, if one exists. In general, if none of $G_1, \ldots, G_{i-1}$ wins, then the algorithm for G_i chooses $Z_1, \ldots Z_{i-1}$ so the first $i - 1$ disjuncts in (212) are false, and evaluates $F_i(X, Z_1, \ldots, Z_{i-1})$, looking for an assignment that satisfies α_i. At least one of $G_1, \ldots, G_k$ must win, since otherwise $Z_1, \ldots, Z_k$ can be chosen so as to falsify (212).

$\mathbf{P}/poly$ is the class of problems solvable by a polynomial size family of Boolean circuits, or equivalently the class of problems solvable by a polynomial time Turing machine which is allowed a polynomial length advice string A_n for each input length n. In order to show that $\mathbf{NP} \subseteq \mathbf{P}/poly$ it suffices to define a polytime relation $R(X, Y)$ such that for each n there is an advice string A_n of length bounded by a polynomial in n so that for every propositional formula α of length n, $R(\alpha, A_n)$ holds iff α is satisfiable.

We now explain how to use the functions $G_1, \ldots, G_k$ to define R and A_n. For each satisfiable propositional formula α of length n we associate a fixed assignment Z_α which satisfies α. We define a map H which takes a k-tuple $X = (\alpha_1, \ldots, \alpha_k)$ of distinct satisfiable propositional formulas of length n, where the formulas in X are ordered lexicographically, to a $k-1$-tuple of such formulas, where $H(X)$ is obtained from X by deleting the first formula α_i such that the assignment $G_i(X, Z_{\alpha_1}, \ldots, Z_{\alpha_{i-1}})$ satisfies α_i. Then the ratio of domain size to range size of H is at least $(C - k + 1)/k$, where C is the number of satisfiable formulas of length n. Hence there is a $k-1$-tuple $(\beta_1, \ldots, \beta_{k-1})$ of formulas which is the image under H of at least $(C - k + 1)/k$ different k-tuples. Part of the advice string A_n codes the sequence $\beta_1, \ldots, \beta_{k-1}, Z_{\beta_1}, \ldots Z_{\beta_{k-1}}$. Each of the k-tuples mapping to $(\beta_1, \ldots, \beta_{k-1})$ consists of $(\beta_1, \ldots, \beta_{k-1})$ with a new formula α inserted somewhere. Further distinct such k-tuples have distinct inserted formulas α, since the fromulas in the tuples are ordered lexicographically. Hence there are at least $(C - k + 1)/k$ such formulas α, and each such α has a satisfying assignment which can be computed from the advice string A_n using $G_1, \ldots, G_k$.

Now delete this set of at least $(C - k + 1)/k$ fromulas α from the set of satisfiable formulas of length n, and apply the above process to the set of remaining formulas, obtaining another $k-1$-tuple of formulas and satisfying assignments to add to the advice string A_n. After $O(\log n)$ such iterations, an advice string A_n of length $O(n \log n)$ is obtained which, using the functions $G_1, \ldots, G_k$ suffices to compute a satisfying assignment to any satisfiable formula of length n. This yields the required polynomial time procedure with advice for solving the satisfiability problem.

The correctness proof for the above $\mathbf{P}/poly$ procedure seems to require a counting argument which cannot (as far as we know) be formalized in $\mathbf{V}^\infty$. We now show how to define an advice string A'_n which can be used to put the satisfiability problem in *co*-$\mathbf{NP}/poly$, provably in $\mathbf{V}^2$. Again we use the functions $G_1, \ldots, G_k$ described above. The idea is to find the smallest $\ell, 1 \leq \ell \leq k$, such that there exists formulas $\alpha_{\ell+1}, \ldots, \alpha_k$ of length n (not necessarily satisfiable) such that for all tuples $(\alpha_1, \ldots, \alpha_\ell)$ of satisfiable formulas of length n, and all tuples $(Z_1, \ldots, Z_\ell)$ of satisfying assingments for $\vec{\alpha}$, there exists $i \leq \ell$ such that $G_i(\alpha_1, \ldots, \alpha_k, Z_1, \ldots, Z_{i-1})$ codes a satisfying assignment for α_i. $\mathbf{V}^2$ proves the existence of ℓ and $\alpha_{\ell+1}, \ldots, \alpha_k$ by the Σ_2^B-$\mathbf{MIN}$ axioms (note that k is a candidate for ℓ). The advice A'_n is the tuple $\alpha_{\ell+1}, \ldots, \alpha_k$. Then an arbitrary formula α_ℓ of length n is satisfiable iff for all tuples $(\alpha_1, \ldots, \alpha_{\ell-1})$ and satisfying assignments $(Z_1, \ldots, Z_{\ell-1})$ there exists $i \leq \ell$ such that $G_i(\alpha_1, \ldots, \alpha_k, Z_1, \ldots, Z_{i-1})$ codes a satisfying assignment for α_i. (The 'if' direction follows from the minimality of

ℓ.) Hence we have expressed length n satisfiability with a Π_1^B formula involving the advice A'_n, which shows that the satisfiability problem is in *co*-**NP**/*poly*, and hence **PH** collapses to **NP**/*poly* = *co*-**NP**/*poly*, as desired.

To prove this theorem for $i > 0$, replace **VPV** by $\mathbf{VPV}^i$, and replace the propositional formulas α by qunatified propositional formulas with a quantifier prefix limited to i alternations beginning with $\exists$. $\dashv$

8H. RSUV Isomorphism

Recall the hierarchies of single-sorted theories $\mathbf{S}_2^i$ and $\mathbf{T}_2^i$ (for $i \geq 1$) from Section 3E. In particular, $\mathbf{S}_2^1$ characterizes the class single-sorted **P** in much the same way as $\mathbf{V}^1$ characterizes the class (two-sorted) **P** (Theorem 6.6 and Corollary 6.8). Here we will show that each theory $\mathbf{S}_2^i$ is essentially a single-sorted version of $\mathbf{V}^i$ (for $i \geq 1$), i.e., they are "RSUV isomorphic" (the same is true for $\mathbf{T}_2^i$ and $\mathbf{TV}^i$).

This section is organized as follows. First we formally define $\mathbf{S}_2^i$ and $\mathbf{T}_2^i$. Then in Section 8H.2 we define the notion of an RSUV isomorphism as a bijection between classes of single-sorted and two-sorted models. These are associated with the syntactical translations of single-sorted and two-sorted formulas, defined in Subsections 8H.3 and 8H.4. Finally we sketch a proof of the RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$.

8H.1. The Theories $\mathbf{S}_2^i$ and $\mathbf{T}_2^i$. For this subsection it might be helpful to revisit Sections 3A and 3E, and Subsection 4C.2. Recall that the vocabulary for $\mathbf{S}_2^1$ is

$$\mathcal{L}_{\mathbf{S}_2} = [0, S, +, \cdot, \#, |x|, \lfloor \frac{1}{2}x \rfloor;\ =, \leq]$$

where $|x|$ is the length of the binary representation of x, and the function $x \# y = 2^{|x|\cdot|y|}$ provides the polynomial growth in length for the terms of $\mathcal{L}_{\mathbf{S}_2}$.

The *sharply bounded quantifiers* are bounded quantifiers (Definition 3.6) which are of the form $\exists x \leq |t|$ and $\forall x \leq |t|$. The syntactic classes of bounded formulas of $\mathcal{L}_{\mathbf{S}_2}$ are defined as follows.

DEFINITION 8.103 (Bounded Formulas of $\mathcal{L}_{\mathbf{S}_2}$). $\Delta_0^b = \Sigma_0^b = \Pi_0^b$ *is the set of formulas whose quantifiers are sharply bounded. For* $i \geq 0$, Σ_{i+1}^b *and* Π_{i+1}^b *are the smallest sets of formulas that satisfy:*

1) $\Pi_i^b \subseteq \Sigma_{i+1}^b$, $\Sigma_i^b \subseteq \Pi_{i+1}^b$.
2) *If* $\varphi, \psi \in \Sigma_{i+1}^b$ *(or* Π_{i+1}^b*), then so are* $\varphi \wedge \psi$, $\varphi \vee \psi$.
3) *If* $\varphi \in \Sigma_{i+1}^b$ *(resp.* $\varphi \in \Pi_{i+1}^b$*), then* $\neg\varphi \in \Pi_{i+1}^b$ *(resp.* $\neg\varphi \in \Sigma_{i+1}^b$*).*
4) *If* $\varphi \in \Sigma_{i+1}^b$ *(resp.* $\varphi \in \Pi_{i+1}^b$*), then* $\exists x \leq t\ \varphi$ *and* $\forall x \leq |t|\ \varphi$ *are in* Σ_{i+1}^b *(resp.* $\forall x \leq t\ \varphi$ *and* $\exists x \leq |t|\ \varphi$ *are in* Π_{i+1}^b*).*

Notice that different from Σ_i^B and Π_i^B (Definition 4.14), here the formulas in Σ_i^b and Π_i^b are not required to be in prenex form, and any bounded quantifier can occur in the scope of a sharply bounded quantifier. Nevertheless, it can be shown that for $i \geq 1$, a single-sorted relation is in the (single-sorted) class Σ_i^P if and only if it is represented by a Σ_i^b formula. In particular, a single-sorted relation is in **NP** if and only if it is represented by a Σ_1^b formula. (See Definition 4.15 and the Σ_i^B and Σ_1^1 Representation Theorem 4.19.)

The set **BASIC** of the defining axioms for symbols in $\mathcal{L}_{\mathbf{S}_2}$ are given in Figure 3. There 1 and 2 are the numerals $S0$ and $SS0$, respectively. Note that **BASIC** is by no means optimal, i.e., it is possible to derive some of its axioms from others. Here we are not concerned with its optimality.

1. $x \leq y \supset Sx \leq Sy$	17. $\lvert x\rvert = \lvert y\rvert \supset x\#z = y\#z$
2. $x \neq Sx$	18. $\lvert x\rvert = \lvert u\rvert + \lvert v\rvert \supset$ $x\#y = (u\#y)\cdot(v\#y)$
3. $0 \leq x$	19. $x \leq x + y$
4. $(x \leq y \wedge x \neq y) \leftrightarrow Sx \leq y$	20. $x \leq y \wedge x \neq y \supset$ $S(2\cdot x) \leq 2\cdot y \wedge S(2\cdot x) \neq 2\cdot y$
5. $x \neq 0 \supset 2\cdot x \neq 0$	21. $x + y = y + y$
6. $x \leq y \vee y \leq x$	22. $x + 0 = x$
7. $(x \leq y \wedge y \leq x) \supset x = y$	23. $x + Sy = S(x+y)$
8. $(x \leq y \wedge y \leq z) \supset x \leq z$	24. $(x+y)+z = x+(y+z)$
9. $\lvert 0\rvert = 0$	25. $x + y \leq x + z \leftrightarrow y \leq z$
10. $\lvert S0\rvert = S0$	26. $x \cdot 0 = 0$
11. $x \neq 0 \supset (\lvert 2\cdot x\rvert = S(\lvert x\rvert)\wedge$ $\lvert S(2\cdot x)\rvert = S(\lvert x\rvert))$	27. $x \cdot Sy = (x\cdot y) + x$
12. $x \leq y \supset \lvert x\rvert \leq \lvert y\rvert$	28. $x \cdot y = y \cdot x$
13. $\lvert x\#y\rvert = S(\lvert x\rvert\cdot\lvert y\rvert)$	29. $x\cdot(y+z) = (x\cdot y)+(x\cdot z)$
14. $0\#x = S0$	30. $1 \leq x \supset (x\cdot y \leq x\cdot z \leftrightarrow y \leq z)$
15. $x \neq 0 \supset (1\#(2\cdot x) = 2\cdot(1\#x)$ $\wedge 1\#S(2\cdot x)) = 2\cdot(1\#x))$	31. $x \neq 0 \supset \lvert x\rvert = S(\lvert\lfloor\frac{1}{2}x\rfloor\rvert)$
16. $x\#y = y\#x$	32. $x = \lfloor\frac{1}{2}y\rfloor \leftrightarrow$ $(2\cdot x = y \vee S(2\cdot x) = y)$

Figure 3. **BASIC**

Recall the definition of an induction scheme Φ-**IND** (Definition 3.4). For formulas of $\mathcal{L}_{\mathbf{S}_2}$ there are other kinds of induction, namely *length induction* and *polynomially induction*, which are defined below.

Definition 8.104 (**LIND** and **PIND**). *Let $\mathcal{L}$ be a vocabulary which extends $\mathcal{L}_{\mathbf{S}_2}$, and Φ be a set of $\mathcal{L}$-formulas. Then Φ-**LIND** is the set of formulas of the form*

$$\big(\varphi(0) \wedge \forall x(\varphi(x) \supset \varphi(x+1))\big) \supset \forall z\varphi(\lvert z\rvert) \tag{213}$$

*and Φ-**PIND** is the set of formulas of the form*

$$\big(\varphi(0) \wedge \forall x(\varphi(\lfloor\tfrac{1}{2}x\rfloor) \supset \varphi(x))\big) \supset \forall z\varphi(z) \tag{214}$$

where φ is a formula in Φ, $\varphi(x)$ is allowed to have free variables other than x.

DEFINITION 8.105 ($\mathbf{S}_2^i$ and $\mathbf{T}_2^i$). *For $i \geq 1$, $\mathbf{S}_2^i$ is the theory axiomatized by* **BASIC** *and Σ_i^b-***PIND***; $\mathbf{T}_2^i$ is the theory axiomatized by* **BASIC** *and Σ_i^b-***IND**.

We leave as an exercise the following interesting results:

EXERCISE 8.106. *Show that for $i \geq 1$:*
(a) $\mathbf{S}_2^i$ *can be axiomatized by* **BASIC** *together with Σ_i^b-***LIND**.
(b) $\mathbf{S}_2^i \subseteq \mathbf{T}_2^i \subseteq \mathbf{S}_2^{i+1}$.

$\mathbf{S}_2^1$ and $\mathbf{V}^1$ turn out to be essentially the same, as explained in the next subsection.

8H.2. RSUV Isomorphism. Here we define the notion of RSUV isomorphism model-theoretically by defining the $^\flat$ and $^\sharp$ *mappings* between single-sorted and two-sorted models. These (semantic) mappings are associated with the syntactical translations between of single-sorted and two-sorted formulas, to be defined in later sections.

Recall that $BIT(i, x)$ is the relation which holds if and only if the i-th lower-order bit in the binary representation of x is 1. It is left as an exercise to show that this relation is definable in $\mathbf{S}_2^1$. It follows that $\mathbf{S}_2^1(BIT)$ is a conservative extension of $\mathbf{S}_2^1$.

EXERCISE 8.107. *Show that $BIT(i, x)$ is definable in $\mathbf{S}_2^1$, and that*

$$\mathbf{S}_2^1(BIT) \vdash \forall x \forall y, x = y \leftrightarrow (|x| = |y| \wedge \forall i \leq |x|, BIT(i, x) \leftrightarrow BIT(i, y))$$

Now let $\mathcal{M}$ be a model of $\mathbf{S}_2^1$ with universe U. We can construct from $\mathcal{M}$ a two-sorted $\mathcal{L}_A^2$-structure $\mathcal{N}$ as follows. First, expand $\mathcal{M}$ to include the interpretation of BIT. The universe $\langle U_1, U_2\rangle$ of $\mathcal{N}$ is defined to be

$$U_2 = U, \qquad \text{and} \qquad U_1 = \{|u| : u \in U\}$$

The constants 0 and 1 are interpreted as 0 and $S0$ respectively (which are in U_1, by the axioms 9 and 10 of **BASIC**). The interpretations of the other symbols of $\mathcal{L}_A^2$ (except for $\in$) in $\mathcal{N}$ are exactly as in $\mathcal{M}$. (Note that by this definition, $|\;|$ is clearly a function from U_2 to U_1.) Finally $\in$ is interpreted as

$$i \in^{\mathcal{N}} x \Leftrightarrow BIT(i, x) \text{ holds in } \mathcal{M}, \qquad \text{for all } i \in U_1, x \in U_2$$

DEFINITION 8.108. *For a model $\mathcal{M}$ of $\mathbf{S}_2^1$, denote by $\mathcal{M}^\sharp$ the two-sorted structure $\mathcal{N}$ obtained as described above.*

Conversely, suppose that $\mathcal{N}$ is a model of $\mathbf{V}^1$ with universe $\langle U_1, U_2\rangle$. We can construct from $\mathcal{N}$ a (single-sorted) $\mathcal{L}_{\mathbf{S}_2}$-structure $\mathcal{M}$ with universe $U = U_2$ where each bounded set X in U_2 is interpreted as the number $bin(X)$ (see (46)):

$$bin(X) = \sum_i X(i) 2^i$$

In order to interpret the symbols of $\mathcal{L}_{\mathbf{S}_2}$ in $\mathcal{M}$, we need the fact that the functions and predicates of $\mathcal{L}_{\mathbf{S}_2}$ when interpreted as taking string arguments are respectively provably total and definable in $\mathbf{V}^1$.

In fact, by Exercise 6.11 the string multiplication function $X \times Y$ is Σ_1^1-definable in $\mathbf{V}^1$. Also, using the fact that $BIT(i,x)$ is definable in $\mathbf{I}\Delta_0$ (Subsection 3C.3) and that $\mathbf{V}^0$ is a conservative extension of $\mathbf{I}\Delta_0$ (Theorem 5.9), we have $BIT(i,x)$ is Σ_0^B-definable in $\mathbf{V}^0$:

Corollary 8.109. *The relation $BIT(i,x)$ is Σ_0^B-definable in $\mathbf{V}^0$.*

Thus the string function $|X|_2$ whose bit-graph is

$$|X|_2(i) \leftrightarrow (i \le |X| \wedge BIT(i,|X|))$$

is provably total in $\mathbf{V}^0$.

The string relation $X \le Y$ is defined in Definition 8.39. The constant 0 is interpreted as the empty set $\varnothing$, which is defined in $\mathbf{V}^0$ by Exercise 5.44. The successor and addition functions on strings are also definable in $\mathbf{V}^0$ (Exercise 5.44). Finally, the functions $X\#Y$ and $\lfloor \frac{1}{2}X \rfloor$ can be defined in $\mathbf{V}^0$ using Σ_0^B-$\mathbf{COMP}$ as follows:

$$(X\#Y)(z) \leftrightarrow z = |X|\cdot|Y|, \qquad \lfloor \tfrac{1}{2}X \rfloor(z) \leftrightarrow z \le |X| \wedge z+1 \in X$$

Definition 8.110. *For a model $\mathcal{N}$ of $\mathbf{V}^1$, let $\mathcal{N}^\flat$ denote the single-sorted $\mathcal{L}_{\mathbf{S}_2}$-structure $\mathcal{M}$ constructed as above.*

Formal definition of RSUV isomorphism is given below.

Definition 8.111 (RSUV Isomorphism). *Let $\mathcal{T}_1$ be a single-sorted theory over $\mathcal{L}_{\mathbf{S}_2}$ and $\mathcal{T}_2$ be a two-sorted theory over $\mathcal{L}_A^2$ so that $\mathbf{S}_2^1 \subseteq \mathcal{T}_1$ and $\mathbf{V}^1 \subseteq \mathcal{T}_2$. Then $\mathcal{T}_1$ and $\mathcal{T}_2$ are said to be RSUV isomorphic (denoted by $\mathcal{T}_1 \overset{\mathrm{RSUV}}{\simeq} \mathcal{T}_2$) if (i) for every model $\mathcal{M}$ of $\mathcal{T}_1$, $\mathcal{M}^\sharp \models \mathcal{T}_2$, and (ii) for every model $\mathcal{N}$ of $\mathcal{T}_2$, $\mathcal{N}^\flat \models \mathcal{T}_1$.*

Note that we can loosen the restrictions that $\mathbf{S}_2^1 \subseteq \mathcal{T}_1$ and $\mathbf{V}^1 \subseteq \mathcal{T}_2$ by, for example, imposing that BIT is definable in $\mathcal{T}_1$, and $X \times Y$ is definable in $\mathcal{T}_2$ (while maintaining that $\mathcal{T}_1$ extends a certain subtheory of $\mathbf{S}_2^1$, and $\mathcal{T}_2$ extends $\mathbf{V}^0$). This allows us to speak of the RSUV isomorphism between subtheories of $\mathbf{S}_2^1$ and $\mathbf{V}^1$.

The main result of this section is stated below.

Theorem 8.112. *For $i \ge 1$, $\mathbf{S}_2^i$ and $\mathbf{V}^i$ are RSUV isomorphic, and $\mathbf{T}_2^i$ and $\mathbf{TV}^i$ are RSUV isomorphic.*

Associated with the $^\sharp$ and $^\flat$ mappings defined above are respectively the $^\flat$ and $^\sharp$ translations of formulas that we will introduce shortly. For example, one direction of Theorem 8.112 (for $i=1$) requires showing that $\mathcal{M}^\sharp \models \mathbf{V}^1$ for every model $\mathcal{M}$ of $\mathbf{S}_2^1(BIT)$. Thus we will translate syntactically an $\mathcal{L}_A^2$ formula φ into an $\mathcal{L}_{\mathbf{S}_2}(BIT)$ formula $\varphi^\flat$ (the $^\flat$ translation) so that

$$\mathcal{M}^\sharp \models \forall\varphi \text{ if and only if } \mathcal{M} \models \forall\varphi^\flat$$

(Recall that $\forall\varphi$ is the universal closure of φ. See Definition 2.41.) Then we will prove that $\mathbf{S}_2^1(BIT) \vdash \varphi^\flat$ for each axiom φ of $\mathbf{V}^1$.

The $^\sharp$ translation is essentially the inverse of the $^\flat$ translation. The RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$ is pictured below (Figure 4).

$$
\begin{array}{ccc}
\mathbf{S}_2^1 & \overset{\text{RSUV}}{\simeq} & \mathbf{V}^1 \\
\hline
\mathcal{M} & \rightharpoonup & \mathcal{M}^\sharp \\
\varphi^\flat & \leftharpoonup & \varphi \\
\hline
\mathcal{N}^\flat & \leftharpoonup & \mathcal{N} \\
\psi & \rightharpoonup & \psi^\sharp
\end{array}
$$

FIGURE 4. The RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$.

In the next two subsections we define the $^\flat$ and $^\sharp$ translations. The proof of Theorem 8.112 will be given in Subsection 8H.5.

8H.3. The $^\sharp$ Translation. The sharply bounded quantifiers in a bounded $\mathcal{L}_{\mathbf{S}_2}$-formula are translated into bounded number quantifiers, and other bounded quantifiers are translated into bounded string quantifiers. In other words, a bound variable is translated into a bound number variable if it is sharply bounded. (Note that the bounding term of a bounded string quantifier bounds the *length* of the quantified variable, while in single-sorted logic the bounding terms are for the *values* of the variables.)

It can be easily seen that simply translating bounded quantifiers as above results in bounded (two-sorted) formulas over the vocabulary that extends $\mathcal{L}_A^2$ by allowing the functions (except 0) and predicates of $\mathcal{L}_{\mathbf{S}_2}$ to be two-sorted functions and predicates whose arguments can be of either sort. For example, there are formally four $+$ functions: one with arity $\langle 2, 0\rangle$, two with arity $\langle 1, 1\rangle$ and one with arity $\langle 0, 2\rangle$. Also, it is straightforward to determine the sorts to which these functions belong. Thus $x + Y$ and $X + Y$ are string functions, while $|x|$ is a number function.

Notation. Let $\mathcal{L}^+$ denote the extension of $\mathcal{L}_A^2$ described above.

The functions of $\mathcal{L}^+$ can be shown to be Σ_1^b-definable in $\mathbf{V}^1$. In fact, the number functions and most of the string functions of $\mathcal{L}^+$ (except for the string multiplication function, or the multiplication functions of "mixed" sorts) are respectively Σ_0^B-definable (in $\mathbf{V}^0$) and Σ_0^B-bit-definable. For example, the number functions $|x|$ and $x\#y$ are Σ_0^B-bit-definable due to the fact that the predicate $BIT(i, x)$ is Δ_0-definable in $\mathbf{I}\Delta_0$ (Subsection 3C.3). For the fact that the afore-mentioned multiplication functions are Σ_1^B-definable in $\mathbf{V}^1$, see Exercise 6.11 and the discussion in the previous subsection about the $^\flat$ mapping.

Now it follows from Corollary 6.27 and Corollary 6.24 that $\mathbf{V}^1(\mathcal{L}^+)$ proves both the $\mathbf{g}\Sigma_1^B(\mathcal{L}^+)$-**COMP** and $\mathbf{g}\Sigma_1^B(\mathcal{L}^+)$-**IND** axiom schemes.

Corollary 8.113. $\mathbf{V}^1(\mathcal{L}^+) \vdash \mathbf{g}\Sigma_1^B(\mathcal{L}^+)$-$\mathbf{IND}$.

Formally we define for each bounded $\mathcal{L}_{\mathbf{S}_2}$ formula $\psi(\vec{x}, \vec{y})$ a bounded $\mathcal{L}^+$ formula $\psi^\sharp(\vec{x}, \vec{Y})$ (i.e., the subset $\vec{y}$ of the free variables of ψ is selected to be translated into the free string variables of $\psi^\sharp$) so that for every model $\mathcal{N}$ of $\mathbf{V}^1$,

$$\mathcal{N}^\flat \models \forall\vec{x}\forall\vec{y}\psi(|\vec{x}|, \vec{y}) \qquad \text{if and only if} \qquad \mathcal{N}[\mathcal{L}^+] \models \forall\vec{x}\forall\vec{Y}\psi^\sharp(\vec{x}, \vec{Y})$$

(where $\mathcal{N}[\mathcal{L}^+]$ denote the expansion of $\mathcal{N}$ by the interpretations for $\mathcal{L}^+$). We will focus on the case where all bounding terms of ψ are of the form $t(\vec{x}, \vec{y})$ (i.e., they involve only the free variables of ψ). We need the following result whose proof is left as an exercise.

Exercise 8.114. *Let $t(\vec{x}, \vec{y})$ be an $\mathcal{L}_{\mathbf{S}_2}$ term. Let $T(\vec{x}, \vec{Y})$ be the $\mathcal{L}^+$ term obtained from $t(\vec{x}, \vec{y})$ by replacing the variables $\vec{y}$ by new string variables $\vec{Y}$, and treating the functions occurring in t as the corresponding functions of $\mathcal{L}^+$. Then there is an $\mathcal{L}_A^2$ term $t'(\vec{x}, |\vec{Y}|)$ so that $\mathbf{V}^1(\mathcal{L}^+) \vdash |T(\vec{x}, \vec{Y})| \le t'(\vec{x}, |\vec{Y}|)$.*

The formula $\psi^\sharp(\vec{x}, \vec{Y})$ is constructed inductively as follows. First if $\psi(\vec{x}, \vec{y})$ is an atomic formula, then $\psi^\sharp(\vec{x}, \vec{Y})$ is the atomic formula obtained from $\psi(\vec{x}, \vec{y})$ by translating the free variables $\vec{y}$ into free string variables $\vec{Y}$, and translating the symbols of $\mathcal{L}_{\mathbf{S}_2}$ into the appropriate symbols of $\mathcal{L}^+$.

Next, if ψ is $\psi_1 \wedge \psi_2$ (resp. $\psi_1 \vee \psi_2$), then $\psi^\sharp$ is $\psi_1^\sharp \wedge \psi_2^\sharp$ (resp. $\psi_1^\sharp \vee \psi_2^\sharp$). If $\psi \equiv \neg\psi_1$, then $\psi^\sharp$ is obtained from of $\neg\psi_1^\sharp$ by pushing the $\neg$ to the atomic subformulas.

Now consider the case where $\psi(\vec{x}, \vec{y}) \equiv \exists z \le t\, \psi_1(z, \vec{x}, \vec{y})$. Let $T(\vec{x}, \vec{Y})$ and $t'(\vec{x}, |\vec{Y}|)$ be as in Exercise 8.114. Then

$$\psi^\sharp(\vec{x}, \vec{Y}) \equiv \exists Z \le 1 + t'(\vec{x}, |\vec{Y}|),\, Z \le T(\vec{x}, \vec{Y}) \wedge \psi_1^\sharp(Z, \vec{x}, \vec{Y})$$

Finally suppose that $\psi(\vec{x}, \vec{y}) \equiv \exists z \le |t|\, \psi_1(z, \vec{x}, \vec{y})$. Then

$$\psi^\sharp(\vec{x}, \vec{Y}) \equiv \exists z \le t'(\vec{x}, |\vec{Y}|),\, z \le |T(\vec{x}, \vec{Y})| \wedge \psi_1^\sharp(z, \vec{x}, \vec{Y})$$

The cases where $\psi(\vec{x}, \vec{y}) \equiv \forall z \le t\, \psi_1(z, \vec{x}, \vec{y})$ or $\psi(\vec{x}, \vec{y}) \equiv \forall z \le |t|\, \psi_1(z, \vec{x}, \vec{y})$ are handled similarly. This completes our description of the $^\sharp$ translation. The proof of its desired properties are left as an exercise.

Exercise 8.115. *Let $\psi(\vec{x}, \vec{y})$ be an $\mathcal{L}_A^2$-formula.*

(a) *Show that if ψ is in Σ_i^b (resp. Π_i^b) for some $i \ge 0$, then $\psi^\sharp(\vec{Y})$ is in $\mathbf{g}\Sigma_i^B(\mathcal{L}^+)$ (resp. $\mathbf{g}\Pi_i^B(\mathcal{L}^+)$).*

(b) *Let $\mathcal{N}$ be a model of $\mathbf{V}^1$. Show that*

$$\mathcal{N}^\flat \models \forall\vec{x}\forall\vec{y}\psi(|\vec{x}|, \vec{y}) \qquad \textit{if and only if} \qquad \mathcal{N}[\mathcal{L}^+] \models \forall\vec{x}\forall\vec{Y}\psi^\sharp(\vec{x}, \vec{Y})$$

8H.4. The $^\flat$ Translation. The $^\flat$ translation is essentially a syntactical counter-part of the $^\sharp$ mapping. In general we will translate bounded string quantifiers into bounded quantifiers, and bounded number quantifiers into sharply bounded quantifiers. Thus we need to find the translation t' for each bounding term t. This task is left as an exercise (see also Exercise 8.114).

Exercise 8.116. *Let $t(\vec{x}, |\vec{Y}|)$ be an $\mathcal{L}_A^2$-term, and $t_1(\vec{x}, |\vec{y}|)$ be the $\mathcal{L}_{\mathbf{S}_2}$-term obtained from t by replacing each the string variables $\vec{Y}$ by new variables $\vec{y}$, and replacing each occurrence of 1 by $S0$. Then there is an $\mathcal{L}_{\mathbf{S}_2}$-term $t'(\vec{x}, \vec{y})$ so that $\mathbf{S}_2^1 \vdash t_1(|\vec{x}|, |\vec{y}|) \leq |t'(\vec{x}, \vec{y})|$.*

We also need the following results, which follows from the fact that BIT is Σ_1^b-definable in $\mathbf{S}_2^1$.

Notation. Let $\mathcal{L}_{\mathbf{S}_2}^+$ stand for $\mathcal{L}_{\mathbf{S}_2} \cup \{BIT\}$.

Exercise 8.117. *Show that $\mathbf{S}_2^1(BIT)$ proves both axiom schemes*

$$\Sigma_1^b(BIT)\text{-}\mathbf{LIND} \text{ and } \Sigma_1^b(BIT)\text{-}\mathbf{IND}.$$

As in the $^\sharp$ translation, we will consider only those formulas whose bounding terms involve only the free variables. Thus suppose that $\varphi(\vec{x}, \vec{Y})$ is such a formula, i.e., all the bounding terms in φ are of the form $t(\vec{x}, |\vec{Y}|)$ (with all variables displayed). Then the $\mathcal{L}_{\mathbf{S}_2}^+$ formula $\varphi^\flat(\vec{x}, \vec{y})$, which has the same set of variables as that of φ (where each string variable Y is replaced by a new variable y), satisfies

$$\mathcal{M}^\sharp \models \forall\vec{x}\forall\vec{Y}\varphi(\vec{x}, \vec{Y}) \quad \text{if and only if} \quad \mathcal{M} \models \forall\vec{x}\forall\vec{y}\varphi^\flat(|\vec{x}|, \vec{y})$$

for any model $\mathcal{M}$ of $\mathbf{S}_2^1(BIT)$.

The formula $\varphi^\flat(\vec{x}, \vec{y})$ is defined inductively as follows. First, if $\varphi(\vec{x}, \vec{Y})$ is an atomic formula, then let $\varphi^\flat(\vec{x}, \vec{y})$ be obtained from $\varphi(\vec{x}, \vec{y})$ by

- replacing each occurrence of 1 by $S0$,
- replacing each occurrence of $Y(t)$ by $BIT(t, Y)$, and
- replacing each occurrence of a string variable Y by the corresponding new variable y.

For the induction step, if $\varphi \equiv (\varphi_1 \wedge \varphi_2)$ (resp. $(\varphi_1 \vee \varphi_2)$, $\neg\varphi_1$), then define $\varphi^\flat \equiv (\varphi_1^\flat \wedge \varphi_2^\flat)$ (resp. $(\varphi_1^\flat \vee \varphi_2^\flat)$, $\neg\varphi_1^\flat$).

Next consider the case where $\varphi(\vec{x}, \vec{Y}) \equiv \exists Z \leq t(\vec{x}, |\vec{Y}|)\, \varphi_1(\vec{x}, \vec{Y}, Z)$. Let $t'(\vec{x}, \vec{y})$ be as in Exercise 8.116. Then

$$\varphi^\flat(\vec{x}, \vec{y}) \equiv \exists z \leq S0 + t'(\vec{x}, \vec{y}),\, |z| \leq t(\vec{x}, |\vec{y}|) \wedge \varphi_1^\flat(\vec{x}, \vec{y}, z)$$

Now consider the case where $\varphi(\vec{x}, \vec{Y}) \equiv \exists u \leq t(\vec{x}, |\vec{Y}|)\, \varphi_1(u, \vec{x}, \vec{Y})$. Let $t'(\vec{x}, \vec{y})$ be as before. Then define

$$\varphi^\flat(\vec{x}, \vec{y}) \equiv \exists u \leq |t'(\vec{x}, \vec{y})|,\, u \leq t(\vec{x}, |\vec{y}|) \wedge \varphi_1^\flat(u, \vec{x}, \vec{y})$$

The cases where $\varphi(\vec{x}, \vec{Y}) \equiv \forall Z \le t(\vec{x}, |\vec{Y}|)\, \varphi_1(\vec{x}, \vec{Y}, Z)$ or $\varphi(\vec{x}, \vec{Y}) \equiv \forall u \le t(\vec{x}, |\vec{Y}|)\, \varphi_1(u, \vec{x}, \vec{Y})$ are handled analogously. This completes our description of the $^\flat$ translation.

The desired properties of $\varphi^\flat$ can be proved by structural induction on φ. Details are left as an exercise.

Exercise 8.118. *Let $\varphi(\vec{x}, \vec{Y})$ be an $\mathcal{L}_A^2$-formula.*

(a) *Show that if φ is in Σ_i^B (resp. Π_i^B) for some $i \ge 0$, then $\varphi^\flat(|\vec{x}|, \vec{y})$ is in $\Sigma_i^b(BIT)$ (resp. $\Pi_i^b(BIT)$).*
(b) *Let $\mathcal{M}$ be a model of $\mathbf{S}_2^1(BIT)$. Show that*

$$\mathcal{M}^\sharp \models \forall \vec{x} \forall \vec{Y} \varphi(\vec{x}, \vec{Y}) \qquad \textit{if and only if} \qquad \mathcal{M} \models \forall \vec{x} \forall \vec{Y} \varphi^\flat(|\vec{x}|, \vec{y})$$

8H.5. The RSUV Isomorphism between $\mathbf{S}_2^i$ and $\mathbf{V}^i$. In this subsection we will sketch the proof of the RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$. The proof of the RSUV isomorphism between $\mathbf{S}_2^i$ and $\mathbf{V}^i$ for $i \ge 2$ is similar, and is left as an exercise.

First, the next theorem is useful in proving RSUV isomorphism.

Notation. We will assume that the theories mentioned here are axiomatized by set of formulas whose bounding terms do not contain any bound variable.

Theorem 8.119. *Let $\mathcal{T}_1$ be a single-sorted theory over $\mathcal{L}_{\mathbf{S}_2}$ such that $\mathbf{S}_2^1 \subseteq \mathcal{T}_1$, and $\mathcal{T}_2$ be a two-sorted theory over $\mathcal{L}_A^2$ such that $\mathbf{V}^1 \subseteq \mathcal{T}_2$. Suppose that (i) $\mathcal{T}_1(BIT) \vdash \varphi^\flat$ for every axiom φ of $\mathcal{T}_2$, and (ii) $\mathcal{T}_2(\mathcal{L}^+) \vdash \psi^\sharp$ for every axiom ψ of $\mathcal{T}_1$. Then $\mathcal{T}_1 \overset{\text{RSUV}}{\simeq} \mathcal{T}_2$.*

Proof. We show that $\mathcal{M}^\sharp \models \mathcal{T}_2$ for every model $\mathcal{M}$ of $\mathcal{T}_1$. The other half (that $\mathcal{N}^\flat \models \mathcal{T}_1$ for every model $\mathcal{N}$ of $\mathcal{T}_2$) is similar.

Thus suppose that $\mathcal{M} \models \mathcal{T}_1(BIT)$. Then by (i) we have $\mathcal{M} \models \varphi^\flat$ for every axiom φ of $\mathcal{T}_2$. By Exercise 8.118 (b) it follows that $\mathcal{M}^\sharp \models \mathcal{T}_2$. $\dashv$

Exercise 8.120. *Show that $\mathbf{S}_2^1(BIT) \vdash \psi \leftrightarrow (\psi^\sharp)^\flat$ and $\mathbf{V}^1(\mathcal{L}^+) \vdash \varphi \leftrightarrow (\varphi^\flat)^\sharp$ for every bounded $\mathcal{L}_{\mathbf{S}_2}$ formula ψ and bounded $\mathcal{L}_A^2$ formula φ.*

Notice that it follows from Theorem 8.112 that if $\mathcal{M}$ is a model of $\mathbf{S}_2^1$, then $\mathcal{M}^\sharp$ is a model of $\mathbf{V}^1$. Hence we can define $(\mathcal{M}^\sharp)^\flat$. Similarly, if $\mathcal{N}$ is a model of $\mathbf{V}^1$, then $(\mathcal{N}^\flat)^\sharp$ is well-defined. The $^\sharp$ and $^\flat$ operations turn out to define a bijection between isomorphism classes of models of $\mathbf{S}_2^1$ and $\mathbf{V}^1$, as shown in the next corollary.

Corollary 8.121. *Let $\mathcal{T}_1$ be a single-sorted theory that extends $\mathbf{S}_2^1$. Then $(\mathcal{M}^\sharp)^\flat$ and $\mathcal{M}$ are same for every model $\mathcal{M}$ of $\mathcal{T}_1$. Similarly, suppose that $\mathcal{T}_2$ is a two-sorted theory that extends $\mathbf{V}^1$. Then $(\mathcal{N}^\flat)^\sharp$ is isomorphic to $\mathcal{N}$ for every model $\mathcal{N}$ of $\mathcal{T}_2$.*

PROOF SKETCH. First, let $\mathcal{M}$ be a model of $\mathcal{T}_1$. Clearly $\mathcal{M}$ and $(\mathcal{M}^\sharp)^\flat$ have the same universe. Indeed, the mappings

$$U(\mathcal{M}) \longrightarrow U_2(\mathcal{M}^\sharp) \longrightarrow U((\mathcal{M}^\sharp)^\flat)$$

are all identity maps. (Here $U(\mathcal{M})$ and $U((\mathcal{M}^\sharp)^\flat)$ denote respectively the universe of $\mathcal{M}$ and $(\mathcal{M}^\sharp)^\flat$, and $U_2(\mathcal{M}^\sharp)$ denotes the second-sort universe of $\mathcal{M}^\sharp$.) So we need to show that the symbols of $\mathcal{L}_{\mathbf{S}_2}$ have the same interpretations in $\mathcal{M}$ and $(\mathcal{M}^\sharp)^\flat$. This essentially follows from the fact that $\mathcal{M}^\sharp \models \mathbf{V}^1$, the functions and relations of $\mathcal{L}^+$ are definable in $\mathbf{V}^1$, and that the "extension axiom" is provable in $\mathbf{S}^1_2$ (Exercise 8.107).

The second statement is proved similarly. (Here $(\mathcal{N}^\flat)^\sharp$ and $\mathcal{N}$ might have different first-sort universes, but they are isomorphic.) ⊣

The next corollary provides the converse of Theorem 8.119 above.

COROLLARY 8.122. *Let $\mathcal{T}_1$ be a single-sorted theory over $\mathcal{L}_{\mathbf{S}_2}$ and $\mathcal{T}_2$ be a two-sorted theory over $\mathcal{L}^2_A$ such that $\mathcal{T}_1 \overset{\text{RSUV}}{\simeq} \mathcal{T}_2$. Then*
 (i) *$\mathcal{T}_1(BIT) \vdash \varphi^\flat$ for every axiom φ of $\mathcal{T}_2$, and*
 (ii) *$\mathcal{T}_2(\mathcal{L}^+) \vdash \psi^\sharp$ for every axiom ψ of $\mathcal{T}_1$.*

PROOF. For (i), let $\mathcal{M}$ be a model of $\mathcal{T}_1$ and φ be an axiom of $\mathcal{T}_2$. Then $\mathcal{M}^\sharp \models \mathcal{T}_2$. Therefore by Exercise 8.118 (b) $(\mathcal{M}^\sharp)^\flat \models \varphi^\flat$. Since $(\mathcal{M}^\sharp)^\flat$ and $\mathcal{M}$ are the same structure (Corollary 8.121), it follows that $\mathcal{M} \models \varphi^\flat$. Hence $\mathcal{T}_1 \vdash \varphi^\flat$.

(ii) is proved similarly using Exercise 8.115 (b). ⊣

THEOREM 8.123. *Suppose that $\mathcal{T}_1$ and $\mathcal{T}_2$ are RSUV isomorphic. Then $\mathcal{T}_1$ is finitely axiomatizable if and only if $\mathcal{T}_2$ is.*

PROOF. Suppose that $\mathcal{T}_1$ is a finitely axiomatizable single-sorted theory. Note that by the Σ^B_1-Transformation Lemma 6.25, for each $\mathcal{L}^+$ formula φ there is an $\mathcal{L}^2_A$ formula φ' so that $\mathbf{V}^1(\mathcal{L}^+) \vdash \varphi \leftrightarrow \varphi'$. We will use this notation in the following definition. Let $\mathcal{T}$ denote the union of the following set

$$\{(\psi^\sharp)' : \psi \text{ is an axiom of } \mathcal{T}_1(BIT)\}$$

and the set of the sentences of the form

$$\forall \vec{x} \forall \vec{Y} \exists ! z \varphi(\vec{x}, z, \vec{Y})$$

or

$$\forall \vec{x} \forall \vec{Y} \exists ! Z \varphi(\vec{x}, Z, \vec{Y})$$

where φ the the formula in the defining axiom of a function symbol of $\mathcal{L}^+$.

We show that $\mathcal{T}_2$ can be axiomatized by $\mathcal{T}$. First, let ψ be an axiom of $\mathcal{T}_1$. By Corollary 8.122 (ii) above, $\mathcal{T}_2(\mathcal{L}^+) \vdash \psi^\sharp$. Consequently (since $\mathcal{T}_2$ extends $\mathbf{V}^1$, and $\mathcal{T}_2(\mathcal{L}^+)$ is conservative over $\mathcal{T}_2$) $\mathcal{T}_2 \vdash (\psi^\sharp)'$. The defining axioms for symbols of $\mathcal{L}^+$ are in $\mathcal{T}_2$ because $\mathbf{V}^1 \subseteq \mathcal{T}_2$.

It remains to show that $\mathcal{T} \vdash \varphi$ for each axiom φ of $\mathcal{T}_2$.

Claim. For each model $\mathcal{N}$ of $\mathcal{T}$, there is a model $\mathcal{M}$ of $\mathcal{T}_1(BIT)$ so that $\mathcal{M}^\sharp = \mathcal{N}$.

Let φ be an axiom of $\mathcal{T}_2$. Let $\mathcal{N}$ be any model of $\mathcal{T}$, and let $\mathcal{M}$ be as in the Claim. Since $\mathcal{M} \models \mathcal{T}_1(BIT)$ and $\mathcal{T}_1(BIT) \models \varphi^\flat$ we have $\mathcal{M} \models \varphi^\flat$. By Exercise 8.118 (b) we have $\mathcal{N} \models \varphi$.

Finally, the Claim follows from part (a) of the exercise below and the fact that $\mathcal{T} \vdash (\psi^\sharp)' \leftrightarrow \psi^\sharp$ for every axiom ψ of $\mathcal{T}_1$. The latter follows from a careful examination of the proof of part (c) of the Σ_1^B-Transformation Lemma 6.25. (Here we do not require that $\mathcal{T}$ proves the Replacement axiom scheme.) ⊣

Exercise 8.124. (a) *Suppose that $\mathcal{T}_1$ is a single-sorted theory that extends $\mathbf{S}_2^1$. Show that for every two-sorted model $\mathcal{N}$ of the set $\{\psi^\sharp : \psi$ is an axiom of $\mathcal{T}_1\}$ there is a model $\mathcal{M}$ of $\mathcal{T}_1$ so that $\mathcal{M}^\sharp = \mathcal{N}$.*

(b) *Similarly, let $\mathcal{T}_2$ be a two-sorted theory that extends $\mathbf{V}^1$, and $\mathcal{T}_2' = \{\varphi^\flat : \varphi$ is an axiom of $\mathcal{T}_2\}$. Show that for every model $\mathcal{M}$ of $\mathcal{T}_2'$ there is a model $\mathcal{N}$ of $\mathcal{T}_2$ so that $\mathcal{M} = \mathcal{N}^\flat$.*

Proof sketch of $\mathbf{S}_2^1 \overset{\text{RSUV}}{\simeq} \mathbf{V}^1$. We need to show that $\mathbf{V}^1(\mathcal{L}^+)$ proves the $^\sharp$ translations of the axioms in **BASIC** as well as Σ_1^b-**LIND** (see Exercise 8.106). The former is straightforward and is left as an exercise.

Exercise 8.125. *Show that $\mathbf{V}^1(\mathcal{L}^+)$ proves the $^\sharp$ translations of the* **BASIC** *axioms.*

Now we consider the Σ_1^b-**LIND** axiom scheme. We will show that $\mathcal{N}$ satisfies the $^\sharp$ translations of the following *bounded length induction* for Σ_1^b formulas, which logically imply Σ_1^b-**LIND**:

$$(215)\qquad [\varphi(0) \wedge \forall x \leq |z|,\ \varphi(x) \supset \varphi(x+1)] \supset \forall z \varphi(|z|)$$

(where φ is a Σ_1^b formula).

Using Exercise 8.115 (a) it is easy to see that instances of (215) translate into $\mathbf{g}\Sigma_1^B(\mathcal{L}^+)$-**IND**. Hence the conclusion follows from Corollary 8.113.

Now consider the next half of the RSUV isomorphism. By Theorem 6.35 it suffices to show that $\mathbf{S}_2^1(BIT)$ satisfies the $^\flat$ translations of the 2-**BASIC** axioms and Σ_1^B-**IND** axioms. The latter translate into $\Sigma_1^b(BIT)$-**LIND** which is provable in $\mathbf{S}_2^1(BIT)$ by Exercise 8.117. Thus the following simple exercise completes our proof of the RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$. ⊣

Exercise 8.126. *Show that $\mathbf{S}_2^1(BIT)$ proves the $^\flat$ translation of the* 2-**BASIC** *axioms.*

Exercise 8.127. *Complete the proof of Theorem 8.112 by showing that $\mathbf{S}_2^i \overset{\text{RSUV}}{\simeq} \mathbf{V}^i$ for $i \geq 2$.*

8I. Notes

The theory **VP** in Section 8A is from [63], but the theory $\widehat{\mathbf{VP}}$ is new.

The theory **VPV** defined in Section 8B is based on the single-sorted equational theory **PV** [28]. The results in Section 8B.1 were first proved in single-sorted versions in Chapter 6 of [12].

In Section 8C the $\mathbf{TV}^i$ hierarchy for $i \geq 1$ is the two-sorted version of Buss's [12] $\mathbf{T}^i_2$ hierarchy. The theory $\mathbf{TV}^0$ was introduced in [30] where the results of Section 8C are outlined, except Theorem 8.44 is from [63].

The theory $\mathbf{V}^1$-**HORN** was introduced in [31], where versions of the results of Section 8D are proved.

The **PLS** problems were introduced in [49]. The results in Section 8E are mostly two-sorted versions of results from [21]. However our Witnessing Theorem 8.77 is stronger than the one in [21], in that our witnessing function G is in the small class $\mathbf{FAC}^0$, and the weak theory $\overline{\mathbf{V}}^0$, as opposed to $\mathbf{TV}^1$, proves the witnessing.

The results from Section 8F.1 are from [36].

Results and definitions in Section 8G have single-sorted precursors as follows. Theorem 8.86 is from [12]. The theories $\mathbf{VPV}^i$ are (for $i \geq 2$) two-sorted versions of the theories $\mathbf{PV}^i$ introduced in [58]. Theorems 8.94 and 8.95 are from [12, 15, 58]. Theorem 8.96 is from [21, 24]. Definition 8.98 (witness oracles) is from [22]. Theorem 8.99 (i) is from [54] and (ii) is from [72] and [61].

Table 1 is inspired by Table 2.1 in [61].

Buss [12] introduced the hierarchies $\mathbf{S}_2$, $\mathbf{T}_2$, and more generally, $\mathbf{S}_k$, $\mathbf{T}_k$ (for $k \geq 2$). (The index k indicates the presence of the function $\#_k$, where $\#_2 = \#$, and $x \#_{k+1} y = 2^{|x| \#_k |y|}$.) He also introduced the hierarchy $\mathbf{U}_2, \mathbf{V}_2$, where $\mathbf{U}^1_2$ and $\mathbf{V}^1_2$ capture **PSPACE** and **EXPTIME**, respectively. (The theories $\mathbf{V}^i$ in this book is sometimes called $\mathbf{V}^i_1$.) The equivalence between $\mathbf{S}^i_{k+1}$ and $\mathbf{V}^i_k$ was first realized in [53, 81]. The name "RSUV isomorphism" was introduced by Takeuti in [82], where he also introduced the hierarchies $\mathbf{R}_k$, and proved the equivalences between $\mathbf{R}^i_{k+1}$ and $\mathbf{U}^i_k$ and between $\mathbf{S}^i_{k+1}$ and $\mathbf{V}^i_k$. The $\mathbf{S} - \mathbf{V}$ equivalence was also proved in [74]. The syntactic translations $^\flat$ and $^\sharp$ are called *interpretations* in [81, 74] (the symbols $^\flat$ and $^\sharp$ were introduced in [74]).

Chapter 9

THEORIES FOR SMALL CLASSES

In this chapter we develop subtheories of **VP** that are associated with the following subclasses of **P**:

$$\mathbf{AC}^0(m) \subseteq \mathbf{TC}^0 \subseteq \mathbf{NC}^1 \subseteq \mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{NC}$$

For each class **C** we will obtain a theory **VC** in the style of **VP** (Section 8A). Here each theory **VC** is axiomatized by the axioms of $\mathbf{V}^0$ and a single axiom that, roughly speaking, asserts the existence of a solution for a complete problem of **C**. (Thus, since $\mathbf{V}^0$ is finitely axiomatizable, all theories **VC** are finitely axiomatizable.) In this chapter completeness is with respect to $\mathbf{AC}^0$-Turing reductions which are more general than the $\mathbf{AC}^0$-many-one reductions used in Section 8A (see Proposition 8.7). Therefore our results apply to classes such as $\mathbf{TC}^0$ that are not known to have any $\mathbf{AC}^0$-many-one complete problem.

The theory **VP** in Section 8A can be seen as a member of the family **VC** here. In general we consider classes **C** that are the $\mathbf{AC}^0$-closure of a polytime function $F_{\mathbf{C}}$. Together with **VC** we will obtain two universal theories $\widehat{\mathbf{VC}}$ and $\overline{\mathbf{VC}}$ whose classes of provably total functions both are equal to **FC**. First, following the development in Section 8A we will introduce a universal theory $\widehat{\mathbf{VC}}$ and show that it is a conservative extension of **VC**. The vocabulary of $\widehat{\mathbf{VC}}$ is $\mathcal{L}_{\mathbf{FAC}^0} \cup \{F_{\mathbf{C}}\}$, and the terms of $\widehat{\mathbf{VC}}$ represent precisely the functions in **FC**. Then using the Herbrand Theorem it follows that both the Σ_1^B-definable functions of **VC** and of $\widehat{\mathbf{VC}}$ are **FC** (and hence all relations in **C** are Δ_1^B-definable in both theories). Second, the theory $\overline{\mathbf{VC}}$ is obtained from $\widehat{\mathbf{VC}}$ by including symbols for all string functions in **FC**, similar to the way in which $\mathbf{V}^0$ is extended to $\overline{\mathbf{V}}^0$ in Section 5F. The defining axioms for functions in $\overline{\mathbf{VC}}$ are based on the $\mathbf{AC}^0$-reductions to the function $F_{\mathbf{C}}$. We will show that $\overline{\mathbf{VC}}$ is indeed a conservative extension of $\widehat{\mathbf{VC}}$, and conclude from this that $\overline{\mathbf{VC}}$ characterizes **FC** as mentioned.

For some subclasses **C** of **L** we are able to obtain universal theories **VCV** using recursion schemes similar to the limited recursion scheme given in Definition 6.15. Here **VCV** has symbols for all string functions in **FC** but their defining axioms are based on a particular recursion scheme rather than $\mathbf{AC}^0$ reduction as in the case of $\overline{\mathbf{VC}}$. We will prove that in each case **VCV** is conservative over **VC** and this shows the robustness of our definition of **VC**. The conservativity results also justify the "minimality" of our theories for characterizing **C**: here the universal theories have functions (or terms) to represent all functions in **FC** and the functions have straightforward defining axioms (either using $\mathbf{AC}^0$-reductions to the complete problem of **C** or using the recursion scheme that characterizes **FC**).

The fact that a theory **VCV** is a universal conservative extension of **VC** also implies that our theory **VC** proves the recursion scheme for the functions in **FC**. We will formalize in our theories proofs of a number of other mathematical theorems, such as the Pigeonhole Principle (PHP) or the discrete version of Jordan Curve Theorem. Some other theorems are of the form $\mathbf{C}_1 \subseteq \mathbf{C}_2$; for these we need to show that the defining axioms of $\mathbf{VC}_1$ are provable in $\mathbf{VC}_2$. We identify this research area of formalizing mathematical results in theories of Bounded Arithmetic as "Bounded Reverse Mathematics", and we mention some open problems in this area in Section 9G.

The chapter is organized as follows. We start by formally defining the notion of $\mathbf{AC}^0$ reduction in Section 9A. Then in Section 9B we introduce the families **VC**, $\widehat{\mathbf{VC}}$ and $\overline{\mathbf{VC}}$. In the subsequent sections we will define the theories for the classes mentioned above and carry out several formalizations in these theories: theories for $\mathbf{TC}^0$ are presented in Section 9C, theories for $\mathbf{AC}^0(m)$ are presented in Section 9D, theories for the **NC** hierarchy are presented in Section 9E, and theories for **NL** and **L** are given in Section 9F. For each of these sections, it is helpful to revise the meta-theorems that we prove for **VC**, $\widehat{\mathbf{VC}}$ and $\overline{\mathbf{VC}}$ in Section 9B. Finally, some open problems are listed in Section 9G.

9A. $\mathbf{AC}^0$ reductions

Roughly speaking a function F is $\mathbf{AC}^0$-reducible to a collection $\mathcal{L}$ of functions if F can be computed by a uniform polynomial size constant depth family of circuits which have unbounded fan-in gates computing functions from $\mathcal{L}$, in addition to Boolean gates (see for example [7]). This is a Turing style reduction, and generalizes the more restrictive many-one style. The class **P** and all classes that we consider in this chapter are closed under $\mathbf{AC}^0$ reductions. Below we will formalize the notion of $\mathbf{AC}^0$-reducible and show that in standard settings the $\mathbf{FAC}^0$ closure of a set of functions is the same as closure under composition and a comprehension operator.

Recall that a function F (resp. f) is Σ_0^B-definable from $\mathcal{L}$ if it is polynomially bounded, and its bit graph (resp. graph) is represented by a $\Sigma_0^B(\mathcal{L})$ formula (Definition 5.37). The following definition generalizes the notion of Σ_0^B-definability.

DEFINITION 9.1 ($\mathbf{AC}^0$ Reduction). *We say that a string function F (resp. a number function f) is $\mathbf{AC}^0$-reducible to $\mathcal{L}$ if there is a sequence of string functions $F_1, \ldots, F_n$ ($n \geq 0$) such that*

(216) $\quad F_i$ *is Σ_0^B-definable from $\mathcal{L} \cup \{F_1, \ldots, F_{i-1}\}$, for $i = 1, \ldots, n$;*

and that F (resp. f) is Σ_0^B-definable from $\mathcal{L} \cup \{F_1, \ldots, F_n\}$. A relation R is $\mathbf{AC}^0$-reducible to $\mathcal{L}$ if there is a sequence $F_1, \ldots, F_n$ as above, and R is represented by a $\Sigma_0^B(\mathcal{L} \cup \{F_1, \ldots, F_n\})$ formula.

EXERCISE 9.2. *Show that a number function f is $\mathbf{AC}^0$-reducible to $\mathcal{L}$ if and only if $f = |F|$ for some string function F which is $\mathbf{AC}^0$-reducible to $\mathcal{L}$.*

If in the above definition $\mathcal{L}$ consists only of functions in $\mathbf{FAC}^0$, then a single iteration ($n = 1$) is enough to obtain any function in $\mathbf{FAC}^0$, and by Corollary 5.41 no more functions are obtained by further iterations. However, as we shall see in the next section, if we start with a function such as *numones*, then repeated iterations generate the complexity class $\mathbf{TC}^0$. It is an open question whether there is a bound on the number of iterations needed.

DEFINITION 9.3 ($\mathbf{FAC}^0$-and $\mathbf{AC}^0$-Closure). *For a language $\mathcal{L}$, the* $\mathbf{FAC}^0$ closure *of $\mathcal{L}$ is the class of functions which are $\mathbf{AC}^0$-reducible to $\mathcal{L}$. The* $\mathbf{AC}^0$ closure *of $\mathcal{L}$ is the class of relations which are $\mathbf{AC}^0$-reducible to $\mathcal{L}$.*

All complexity classes of interest here are closed under $\mathbf{AC}^0$ reductions, because the corresponding function classes are closed under Σ_0^B-definability. For the case of $\mathbf{FAC}^0$, this follows from Corollary 5.41.

COROLLARY 9.4. *The $\mathbf{FAC}^0$ closure of $\mathbf{FAC}^0$ is $\mathbf{FAC}^0$. The $\mathbf{AC}^0$ closure of $\mathbf{AC}^0$ is $\mathbf{AC}^0$.*

For a complexity class $\mathbf{C}$, recall that $\mathbf{FC}$ is the corresponding function class (Definition 5.15). The following lemma is straightforward consequence of the definitions involved.

LEMMA 9.5. *A complexity class $\mathbf{C}$ is the $\mathbf{AC}^0$ closure of a language $\mathcal{L}$ iff $\mathbf{FC}$ is the $\mathbf{FAC}^0$ closure of $\mathcal{L}$.*

The composition of two functions is $\mathbf{AC}^0$ reducible to the functions, because a term representing the composition can be used in a $\Sigma_0^B(\mathcal{L})$-formula defining the composition. We now define another operation which preserves $\mathbf{AC}^0$ reducibility and which will be used together with composition to give a characterization of $\mathbf{AC}^0$ reducibility. The new operation takes a number function and collects a bounded number of its values in a set to form a string function. This notion and Theorem 9.7 below will be useful in Section 9C.3.

DEFINITION 9.6 (String Comprehension). *For a number function $f(x)$ (which may contain other arguments), the* string comprehension of f *is the string function $F(y)$ such that*

$$F(y) = \{f(x) : x \leq y\}$$

(See (49) for this set-theoretic notation.) Note that if f is polynomially bounded, then so is F.

For example, recall that the Σ_0^B formula $\varphi_{parity}(X, Y)$ (80) on page 114 asserts that for $0 \leq i < |X|$, bit $Y(i+1)$ is 1 iff the number of 1's among bits $X(0), ..., X(i)$ is odd. As a function of X, $Y = F(|X|, X)$, where F is obtained from the following function f by string comprehension:

$$f(x, X) = \begin{cases} x & \text{if } x > 0 \text{ and the number of 1 bits in} \\ & \qquad X(0), \ldots, X(x-1) \text{ is odd} \\ |X| + 1 & \text{otherwise} \end{cases}$$

THEOREM 9.7. *Suppose that $\mathcal{L}$ is a class of polynomially bounded functions that includes $\mathbf{FAC}^0$. Then a function is $\mathbf{AC}^0$-reducible to $\mathcal{L}$ iff it can be obtained from $\mathcal{L}$ by finitely many applications of composition and string comprehension.*

PROOF. For the IF direction, it suffices to prove that a function obtained from input functions by either of the operations composition or string comprehension is Σ_0^B-definable from the input functions.

For composition, suppose

$$F(\vec{x}, \vec{X}) = G(h_1(\vec{x}, \vec{X}), \ldots, h_k(\vec{x}, \vec{X}), H_1(\vec{x}, \vec{X}), \ldots, H_m(\vec{x}, \vec{X}))$$

where G and $h_1, \ldots, h_k$, $H_1, \ldots, H_m$ are polynomially bounded. Then F is also polynomially bounded, and its bit graph $F(\vec{x}, \vec{X})(z)$ is represented by the open formula

$$G(h_1(\vec{x}, \vec{X}), \ldots, h_k(\vec{x}, \vec{X}), H_1(\vec{x}, \vec{X}), \ldots, H_m(\vec{x}, \vec{X}))(z)$$

(A similar argument works for a number function f.)

For string comprehension, suppose that $f(x)$ is a polynomially bounded number function. As noted before, the string comprehension $F(y)$ of f is also polynomially bounded, and it has bit graph

$$F(y)(z) \leftrightarrow z < t \wedge \exists x \leq y \; z = f(x)$$

where t is the bounding term for F. Hence F is also Σ_0^B-definable from f.

For the ONLY IF direction, it suffices to show that if $\mathcal{L} \supseteq \mathbf{FAC}^0$ and F (or f) is Σ_0^B-definable from $\mathcal{L}$, then F (resp. f) can be obtained from $\mathcal{L}$ by composition and string comprehension.

Claim. If $\mathcal{L} \supseteq \mathbf{FAC}^0$ and $\varphi(\vec{z}, \vec{X})$ is a $\Sigma_0^B(\mathcal{L})$ formula, then the characteristic function c_φ defined by

$$c_\varphi(\vec{z}, \vec{Z}) = \begin{cases} 1 & \text{if } \varphi(\vec{z}, \vec{Z}) \\ 0 & \text{otherwise} \end{cases}$$

can be obtained from $\mathcal{L}$ by composition.

The Claim is holds because $c_\psi(\vec{x}, \vec{X})$ is in $\mathbf{FAC}^0$ for every $\Sigma_0^B(\mathcal{L}_A^2)$-formula ψ, and (by structural induction on φ) it is clear that for every $\Sigma_0^B(\mathcal{L})$-formula $\varphi(\vec{z}, \vec{Z})$ there is a $\Sigma_0^B(\mathcal{L}_A^2)$-formula $\psi(\vec{x}, \vec{X})$ such that

$$\varphi(\vec{z}, \vec{Z}) \leftrightarrow \psi(\vec{s}, \vec{T})$$

for some $\mathcal{L}$-terms $\vec{s}$ and $\vec{T}$. Hence

$$c_\varphi(\vec{z}, \vec{Z}) = c_\psi(\vec{s}, \vec{T})$$

Now suppose that F is Σ_0^B-definable from $\mathcal{L}$, so

$$F(\vec{z}, \vec{X})(x) \leftrightarrow x < t \wedge \varphi(x, \vec{z}, \vec{X})$$

where $t = t(\vec{z}, \vec{X})$ is an $\mathcal{L}_A^2$ term and φ is a $\Sigma_0^B(\mathcal{L})$ formula.

Define the number function f by cases as follows:

$$f(x, \vec{z}, \vec{X}) = \begin{cases} x & \text{if } \varphi(x, \vec{z}, \vec{X}) \\ t & \text{if } \neg\varphi(x, \vec{z}, \vec{X}) \end{cases}$$

Then by the Claim, f can be obtained from $\mathcal{L}$ by composition as follows. Define the $\mathbf{FAC}^0$ function g by

$$g(x, y, z, w) = x \cdot y + z \cdot w$$

Thus

$$f(x, \vec{z}, \vec{X}) = g(x, c_\varphi, t, c_{\neg\varphi})$$

Now

$$F(\vec{z}, \vec{X}) = Cut(t, G(t, \vec{z}, \vec{X}))$$

where $G(y, \vec{z}, \vec{X})$ is the string comprehension of $f(x, \vec{z}, \vec{X})$, and Cut (see (96) on page 133) is the $\mathbf{FAC}^0$ function defined by

$$Cut(x, X)(z) \leftrightarrow z < x \wedge X(z)$$

It remains to show that if a number function f is Σ_0^B-definable from $\mathcal{L}$ then f can be obtained from $\mathcal{L}$ by composition and string comprehension. Suppose f satisfies

$$y = f(\vec{z}, \vec{X}) \leftrightarrow y < t \wedge \varphi(y, \vec{z}, \vec{X})$$

where $t = t(\vec{z}, \vec{X})$ is a $\mathcal{L}_A^2$ term and φ is a $\Sigma_0^B(\mathcal{L})$ formula. Use the Claim to define $c_\varphi(y, \vec{z}, \vec{X})$ by composition from $\mathcal{L}$, and define g by

$$g(x, \vec{z}, \vec{X}) = x \cdot c_\varphi(x, \vec{z}, \vec{X})$$

Then

$$f(\vec{z}, \vec{X}) = |G(t, \vec{z}, \vec{X})| \dot{-} 1$$

where $G(y, \vec{z}, \vec{X})$ is the string comprehension of $g(x, \vec{z}, \vec{X})$. $\dashv$

9B. Theories for subclasses of P

In this section, we show how to develop finitely axiomatizable theories for a number of uniform subclasses of **P** in the style of **VP** (Section 8A). Recall that **VP** is obtained from the base theory $\mathbf{V}^0$ by augmenting the axiom MCV which states the existence of the value for F_{MCV}, a function which is $\mathbf{AC}^0$-many-one complete for **P**. Here we obtain a theory **VC** for each class **C** which is the $\mathbf{AC}^0$ closure of a polytime function $F_{\mathbf{C}}$ so that the provably total functions of **VC** are precisely the functions in **FC** and the Δ_1^B-definable relations in **VC** are precisely the relations in **C**. Thus the function F_{MCV} plays the role of $F_{\mathbf{C}}$ when $\mathbf{C} = \mathbf{P}$.

First in Section 9B.1 we will define **VC** and state the definability theorems for **VC**. Then in Section 9B.2 we will follow the discussion in Section 8A and introduce the universal theory $\widehat{\mathbf{VC}}$ in the same style as $\widehat{\mathbf{VP}}$. The vocabulary $\mathcal{L}_{\widehat{\mathbf{VC}}}$ of $\widehat{\mathbf{VC}}$ is $\mathcal{L}_{\mathbf{FAC}^0}$ together with the new function $F_{\mathbf{C}}$. We will show that $\widehat{\mathbf{VC}}$ is a conservative extension of **VC**. We will also prove that the terms in $\mathcal{L}_{\widehat{\mathbf{VC}}}$ represent precisely functions in **FC** and hence the relations in **C** are represented by open formula of $\mathcal{L}_{\widehat{\mathbf{VC}}}$. Consequently we derive our definability theorems for both $\widehat{\mathbf{VC}}$ and **VC**.

In Section 9B.3 we introduce a universal theory $\overline{\mathbf{VC}}$. The language $\mathcal{L}_{\mathbf{FC}}$ of $\overline{\mathbf{VC}}$ contains all string functions of **FC**. (Note that by Exercise 9.2 the number functions in **FC** are represented by $\mathcal{L}_{\mathbf{FC}}$-terms of the form $|G|$, for string functions G in $\mathcal{L}_{\mathbf{FC}}$.) We will show that $\overline{\mathbf{VC}}$ is a conservative extension of both $\widehat{\mathbf{VC}}$ and **VC**, and therefore it also characterizes **C**.

In Section 9B.4 we will discuss a general way of applying our results above to subclasses of **P** mentioned at the beginning of the chapter.

9B.1. The theories VC. In the following discussion the intended function $F_{\mathbf{C}}$ will be simply denoted by F. So suppose that F is a polytime function with a Σ_0^B graph:

$$Y = F(X) \leftrightarrow (|Y| \le t \wedge \delta_F(X,Y)) \tag{217}$$

for some $\mathcal{L}_A^2$ term t and $\Sigma_0^B(\mathcal{L}_A^2)$ formula δ_F. Suppose further that $\mathbf{V}^0$ proves the uniqueness of the value of F:

$$\mathbf{V}^0 \vdash \forall Y_1 \forall Y_2 \big((|Y_1| \le t \wedge |Y_2| \le t \wedge \delta_F(X,Y_1) \wedge \delta_F(X,Y_2)) \supset Y_1 = Y_2\big)$$

Let **C** be the class of two-sorted relations which are $\mathbf{AC}^0$-reducible to F. By Lemma 9.5, the class **FC** (Definition 5.15) can be equivalently defined as the $\mathbf{FAC}^0$ closure of F (Definition 9.3).

Our functions $F_{\mathbf{C}}$ introduced later in this chapter often have more than one argument, but they can be easily defined using an one argument function as above. For example, we can easily encode the arguments (a, G, E) of F_{MCV} into a single string argument X and let F be the resulting function:

$$F(X) = F_{MCV}(a, G, E) \qquad \text{whenever } X \text{ encodes } (a, G, E)$$

Then we have $\mathbf{C} = \mathbf{P}$.

Definition 9.8 ($\mathbf{VC}$). *The theory* $\mathbf{VC}$ *has vocabulary* $\mathcal{L}_A^2 \cup \{Row\}$ *and is axiomatized by* $\mathbf{V}^0(Row)$ *and the following axiom*

$$\forall b \forall X \exists Y \forall i < b\, \delta_F(X^{[i]}, Y^{[i]}) \tag{218}$$

Recall the notion of aggregate functions (Definition 8.9). Notice that (218) states the existence of the value for the aggregate function $F^\star$ of F. Even though δ_{MCV} (156) is only the graph of F_{MCV} (as opposed to F^*_{MCV}), the fact that F^*_{MCV} is Σ_1^B-definable in $\mathbf{VP}$ (Lemma 8.10) shows that $\mathbf{VP}$ is equivalent to a theory $\mathbf{VC}$ defined as above. In Section 9B.4 we explain how to design theories for other classes mentioned in the preface. In each case, we will be able to use the (simpler) defining axiom for F instead of the axioms of the form (218). This is because we can prove that $F^\star$ are definable in our theories (although the proofs are different for each theory).

The next lemma is straightforward:

Lemma 9.9. *The functions* F *and* $F^\star$ *are* Σ_0^B*-definable in* $\mathbf{VC}$*, and*

$$\mathbf{VC}(F, F^\star) \vdash \forall b \forall X \forall i < b\, F^\star(b, X)^{[i]} = F(X^{[i]})$$

Our first goal is to prove the following theorem (recall from Corollary 5.29 that a function is provably total in $\mathbf{VC}$ iff it is Σ_1^B-definable in $\mathbf{VC}$):

Theorem 9.10. *A function is provably total in* $\mathbf{VC}$ *iff it is in* $\mathbf{FC}$.

Corollary 9.11. *A relation is in* $\mathbf{C}$ *iff it is* Δ_1^B*-definable in* $\mathbf{VC}$ *iff it is* Δ_1^1*-definable in* $\mathbf{VC}$.

Proof. From Theorems 9.10 and 5.60. $\dashv$

We prove Theorem 9.10 by introducing the universal conservative extension $\widehat{\mathbf{VC}}$ of $\mathbf{VC}$, an analog of $\widehat{\mathbf{VP}}$. The proof is given on page 265.

9B.2. The theory $\widehat{\mathbf{VC}}$. Here we define the universal theory $\widehat{\mathbf{VC}}$ and show that it is a conservative extension of $\mathbf{VC}$. We start by obtaining a quantifier-free defining axiom for F. For this we need a quantifier-free formula that is equivalent to $\delta_F(X, Y)$. So let $\delta'_F(X, Y)$ be the quantifier-free formula over $\mathcal{L}_{\mathbf{FAC}^0}$ which $\overline{\mathbf{V}}^0$ proves equivalent to $\delta_F(X, Y)$ (by Lemma 5.70). Formally, we will not change the defining axiom for F. Therefore let F' be the function with the same value as F but has the following quantifier-free defining axiom:

$$Y = F'(X) \;\leftrightarrow\; (|Y| \le t \wedge \delta'_F(X, Y)) \tag{219}$$

DEFINITION 9.12 ($\widehat{\mathbf{VC}}$). *$\widehat{\mathbf{VC}}$ is the universal theory over the vocabulary $\mathcal{L}_{\widehat{\mathbf{VC}}} = \mathcal{L}_{\mathbf{FAC}^0} \cup \{F'\}$, and is axiomatized by the axioms of $\overline{\mathbf{V}}^0$ and (219) for F'.*

The next theorem is proved in the same way as Theorem 8.13 using Lemma 9.9 above.

THEOREM 9.13. *The theory $\widehat{\mathbf{VC}}$ is a universal conservative extension of $\mathbf{VC}$.*

The next corollary follows from Theorem 8.15 and is proved in the same way as Corollary 8.16:

COROLLARY 9.14. *The theory $\widehat{\mathbf{VC}}$ proves the axiom schemes:*

$\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$-**COMP**, $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$-**IND**, *and* $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$-**MIN**

The following theorem generalizes Theorem 8.12.

THEOREM 9.15. (a) *A function is in $\mathbf{FC}$ if and only if it is represented by a term in $\mathcal{L}_{\widehat{\mathbf{VC}}}$.*

(b) *A relation is in $\mathbf{C}$ if and only if it is represented by an open formula of $\mathcal{L}_{\widehat{\mathbf{VC}}}$ iff it is represented by a $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$ formula.*

PROOF. It is straightforward to prove (b) from (a). So below we will only prove (a). Here $\mathbf{FC}$ is the $\mathbf{FAC}^0$-closure of F. First we prove by induction based on Definition 9.1 that the functions in $\mathbf{FC}$ are represented by $\mathcal{L}_{\widehat{\mathbf{VC}}}$ terms. The base case is obvious: F is represented by the term $F(\vec{x}, \vec{X})$. For the induction step, consider for example the case of a string function. Suppose that $G(\vec{x}, \vec{X})$ is Σ_0^B-definable from $\mathcal{L} = \{F_1 = F, F_2, \ldots, F_n\}$, and that each F_i is represented by a term T_i in $\mathcal{L}_{\widehat{\mathbf{VC}}}$. By definition, there is a $\Sigma_0^B(\mathcal{L})$ formula $\varphi(z, \vec{x}, \vec{X})$ that represents the bit graph of G, i.e.,

$$G(\vec{x}, \vec{X})(z) \leftrightarrow z \leq t \wedge \varphi(z, \vec{x}, \vec{X})$$

for some $\mathcal{L}_A^2$-term t.

Let $\varphi'(z, \vec{x}, \vec{X})$ be the $\mathcal{L}_{\widehat{\mathbf{VC}}}$-formula obtained from $\varphi(z, \vec{x}, \vec{X})$ by simultaneously substituting $T_i(\vec{s}, \vec{S})$ for all occurrences of $F_i(\vec{s}, \vec{S})$. Let $F(\vec{s}_1, \vec{S}_1)$, ..., $F(\vec{s}_m, \vec{S}_m)$ be all maximal occurrences of F in φ'. Thus

$$\varphi'(\vec{x}, \vec{X}) \equiv \varphi''(\vec{x}, \vec{X}, F(\vec{s}_1, \vec{S}_1), \ldots, F(\vec{s}_m, \vec{S}_m))$$

where $\varphi''(\vec{x}, \vec{X}, Y_1, \ldots, Y_m)$ is a $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$-formula. Then G is represented by the $\mathcal{L}_{\widehat{\mathbf{VC}}}$-term $H(\vec{x}, \vec{X}, F(\vec{s}_1, \vec{S}_1), \ldots, F(\vec{s}_m, \vec{S}_m))$, where H is the $\mathbf{AC}^0$ function with bit graph

$$H(\vec{x}, \vec{X}, Y_1, \ldots, Y_m)(z) \leftrightarrow z \leq t \wedge \varphi''(z, \vec{x}, \vec{X}, Y_1, \ldots, Y_m)$$

For the other direction, we prove by induction on the nesting depth of an $\mathcal{L}_{\widehat{\mathbf{VC}}}$-term that it represents a function in $\mathbf{FC}$. The base case (the nesting depth is 0) is obvious, so consider the induction step. Let

$T(\vec{x}, \vec{X})$ be an $\mathcal{L}_{\widehat{\mathbf{VC}}}$ string term of nesting depth $d \geq 1$. (The case of a number term is similar.) Then

$$T(\vec{x}, \vec{X}) = H(s_1(\vec{x}, \vec{X}), \ldots, s_n(\vec{x}, \vec{X}), T_1(\vec{x}, \vec{X}), \ldots, T_m(\vec{x}, \vec{X}))$$

for $\mathcal{L}_{\widehat{\mathbf{VC}}}$-terms s_i, T_j of nesting depth at most $d-1$, and $H = F$ or H is an $\mathbf{AC}^0$ function. By the induction hypothesis, s_i and T_j represent $\mathbf{C}$ functions f_i and G_j, respectively (for $1 \leq i \leq n, 1 \leq j \leq m$). For $1 \leq i \leq n$ let the string functions F_i in $\mathbf{FC}$ be such that $f_i = |F_i|$ (see Exercise 9.2). Then T represents the function K which is Σ_0^B-definable from $H, F_1, \ldots, F_n, G_1, \ldots, G_m$ as follows:

$$K(\vec{x}, \vec{X})(z) \leftrightarrow$$
$$z \leq t \wedge H(|F_1(\vec{x}, \vec{X})|, \ldots, |F_n(\vec{x}, \vec{X})|, G_1(\vec{x}, \vec{X}), \ldots, G_m(\vec{x}, \vec{X}))(z)$$

for some appropriate $\mathcal{L}_A^2$-term t. This shows that T represents a function in $\mathbf{FC}$. ⊣

COROLLARY 9.16. (a) *A function is $\Sigma_1^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$-definable in $\widehat{\mathbf{VC}}$ iff it is in* $\mathbf{FC}$.

(b) *A relation is* $\mathbf{C}$ *iff it is $\Delta_1^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$-definable in $\widehat{\mathbf{VC}}$.*

PROOF. (a) The fact that every function in $\mathbf{FC}$ is Σ_1^B-definable in $\widehat{\mathbf{VC}}$ follows from Lemmas 9.9 and 9.17 (below) and Theorem 9.15. The other direction follows from this theorem and the Herbrand Theorem (see the proof of Corollary 8.14).

(b) Follows from (a) and Theorem 5.60. ⊣

LEMMA 9.17. *Let $\mathcal{T}_1 \subseteq \mathcal{T}_2$ be theories whose vocabularies include $\mathcal{L}_A^2$. Suppose that every function in the vocabulary $\mathcal{L}_2$ of $\mathcal{T}_2$ is Σ_1^B-definable in $\mathcal{T}_1$. Then every function represented by a term in $\mathcal{L}_2$ is Σ_1^B-definable in $\mathcal{T}_1$.*

PROOF. The result follows by structural induction on terms, using Exercise 5.30. ⊣

The next result is important for replacing the $\Sigma_1^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$ (and $\Pi_1^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$ formulas from Corollary 9.16 above by just Σ_1^B (i.e., $\Sigma_1^B(\mathcal{L}_A^2)$) (and Π_1^B) formulas. (Later we will prove similar theorem for number functions f, $f^\star$.)

THEOREM 9.18 (First Elimination Theorem). *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L}$ which extends $\mathbf{V}^0$ (Row) and proves $\Sigma_0^B(\mathcal{L})$-$\mathbf{COMP}$. Suppose that F and $F^\star$ are Σ_1^B-definable in $\mathcal{T}$ (Definition 5.26) and $\mathcal{T}(F, F^\star)$ proves* (165)*:*

$$\forall i < b,\ F^\star(b, \vec{Z}, \vec{X})^{[i]} = F((Z_1)^i, \ldots, (Z_k)^i, X_1^{[i]}, \ldots, X_n^{[i]})$$

Suppose also that every $\Sigma_0^B(\mathcal{L})$ formula is equivalent in $\mathcal{T}$ to a $\Sigma_1^B(\mathcal{L}_A^2)$ formula. Then every $\Sigma_0^B(\mathcal{L} \cup \{F\})$ formula is equivalent in $\mathcal{T}(F)$ to a $\Sigma_1^B(\mathcal{L}_A^2)$ formula.

PROOF. Let

$$\varphi^+ \equiv \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi(\vec{z})$$

be a $\Sigma_0^B(\mathcal{L}, F)$ formula, where $\mathrm{Q}_1, \dots, \mathrm{Q}_n \in \{\exists, \forall\}$ and ψ is a quantifier-free formula. We show that there is a $\Sigma_1^B(\mathcal{L}_A^2)$ formula φ so that

$$\mathcal{T}(F) \vdash \varphi^+ \leftrightarrow \varphi$$

As in the base case in the proof of Theorem 8.15, the idea here is to replace every occurrence of a term $F(\vec{s}, \vec{T})$ in ψ by a new string variable W which has the intended value of $F(\vec{s}, \vec{T})$. We need to state the existence of such strings, and this contributes to the string quantifiers in the resulting Σ_1^B formula.

So suppose that $F(\vec{s}_1, \vec{T}_1), \dots, F(\vec{s}_k, \vec{T}_k)$ are all occurrences of F in ψ. Note that the terms $\vec{s}_i, \vec{T}_i$ may contain $\vec{z}$ as well as nested occurrences of F. Assume further that these F-terms are ordered by depth so that $\vec{s}_1, \vec{T}_1$ do not contain F, and for $1 < i \le k$, any occurrence of F in $\vec{s}_i, \vec{T}_i$ must be of the form $F(\vec{s}_j, \vec{T}_j)$, for some $j < i$.

Let $W_1, ..., W_k$ be new string variables. Let $\overrightarrow{s_1'} = \vec{s}_1$, $\overrightarrow{T_1'} = \vec{T}_1$, and for $2 \le i \le k$, $\overrightarrow{s_i'}$ and $\overrightarrow{T_i'}$ be obtained from $\vec{s}_i$ and $\vec{T}_i$ respectively by replacing every maximal occurrence of any $F(\vec{s}_j, \vec{T}_j)$, for $j < i$, by $W_j^{[\vec{z}]}$. Thus F does not occur in any $\overrightarrow{s_i'}$ and $\overrightarrow{T_i'}$, but for $i \ge 2$, $\overrightarrow{s_i'}$ and $\overrightarrow{T_i'}$ may contain $W_1, \dots, W_{i-1}$.

Let $\psi'(\vec{z}, W_1, \dots, W_k)$ be obtained from $\psi(\vec{z})$ by replacing each maximal occurrence of $F(\vec{s}_i, \vec{T}_i)$ by $W_i^{[\vec{z}]}$, for $1 \le i \le k$. Obviously,

$$\mathcal{T}(F) \vdash \big[\exists W_1 \dots \exists W_k \big((\forall z_1 \le r_1 \dots \forall z_n \le r_n \bigwedge W_i^{[\vec{z}]} = F(\overrightarrow{s_i'}, \overrightarrow{T_i'})) \wedge$$
$$\mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi'(\vec{z}, \vec{W})\big)\big] \supset \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi(\vec{z})$$

Notice that $W_i = F^\star(\langle \vec{r} \rangle, \vec{U}_i, \vec{V}_i)$ satisfy

$$\forall z_1 \le r_1 \dots \forall z_n \le r_n \bigwedge W_i^{[\vec{z}]} = F(\overrightarrow{s_i'}, \overrightarrow{T_i'})$$

where $U_{i,j}$ and $V_{i,\ell}$ are unique strings such that

$$|U_{i,j}| \le t \wedge \forall z_1 \le r_1 \dots \forall z_k \le r_k (U_{i,j})^{\vec{z}} = s'_{i,j} \tag{220}$$

$$|V_{i,\ell}| \le t \wedge \forall z_1 \le r_1 \dots \forall z_k \le r_k V_{i,\ell}^{[\vec{z}]} = T'_{i,\ell} \tag{221}$$

and t is an $\mathcal{L}_A^2$-term such that $t \ge \langle \vec{r}, max\{|\overrightarrow{T_i'}|, \overrightarrow{s_i'}\}\rangle$ for all $1 \le i \le k$.

Denote the conjunction of (220) and (221) for all i, j, ℓ by $\theta(\vec{U}, \vec{V})$. Then

$$\mathcal{T}(F, F^\star) \vdash \big[\exists \vec{U} \exists V \exists \vec{W} \big(\theta(\vec{U}, \vec{V}) \wedge \bigwedge W_i = F^\star(\langle \vec{r} \rangle, \vec{U}_i, \vec{V}_i) \wedge$$
$$\mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi'(\vec{z}, \vec{W})\big)\big] \supset \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi(\vec{z})$$

On the other hand, since $F^\star$ is definable in $\mathcal{T}$ and since $\mathcal{T} \vdash \Sigma_0^B(\mathcal{L})$-**COMP**, we have

$$\mathcal{T}(F, F^\star) \vdash \exists\vec{U}\exists\vec{V}\exists\vec{W}\big(\theta(\vec{U},\vec{V}) \wedge \bigwedge W_i = F^\star(\langle\vec{r}\rangle, \vec{U}_i, \vec{V}_i)\big) \tag{222}$$

Therefore

$$\begin{aligned}\mathcal{T}(F, F^\star) \vdash{}& \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi(\vec{z}) \supset \\ &\big[\exists\vec{U}\exists\vec{V}\exists\vec{W}\big(\theta(\vec{U},\vec{V}) \wedge \bigwedge W_i = F^\star(\langle\vec{r}\rangle, \vec{U}_i, \vec{V}_i) \wedge \\ &\qquad\qquad \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi'(\vec{z}, \vec{W})\big)\big]\end{aligned}$$

As a result,

$$\begin{aligned}\mathcal{T}(F, F^\star) \vdash{}& \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi(\vec{z}) \leftrightarrow \\ &\big[\exists\vec{U}\exists\vec{V}\exists\vec{W}\big(\theta(\vec{U},\vec{V}) \wedge \bigwedge W_i = F^\star(\langle\vec{r}\rangle, \vec{U}_i, \vec{V}_i) \wedge \\ &\qquad\qquad \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi'(\vec{z}, \vec{W})\big)\big]\end{aligned}$$

Finally, the strings in (222) are bounded by some $\mathcal{L}_A^2$-terms and are provably unique in $\mathcal{T}(F^\star)$. Therefore (222) is equivalent in $\mathcal{T}(F^\star)$ to a $\Sigma_1^B(\mathcal{L}_A^2)$ formula. Also, by the hypothesis,

$$\mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi'(\vec{z}, \vec{W})$$

is equivalent in $\mathcal{T}$ to a $\Sigma_1^B(\mathcal{L}_A^2)$ formula. As a result,

$$\begin{aligned}&\exists\vec{U}\exists\vec{V}\exists\vec{W}\big(\theta(\vec{U},\vec{V}) \wedge \bigwedge W_i = F^\star(\langle\vec{r}\rangle, \vec{U}_i, \vec{V}_i) \wedge \\ &\qquad\qquad \mathrm{Q}_1 z_1 \le r_1 \dots \mathrm{Q}_n z_n \le r_n \psi'(\vec{z}, \vec{W})\big)\end{aligned}$$

is equivalent in $\mathcal{T}(F, F^\star)$ to a Σ_1^B formula. The conclusion follows from the fact that $\mathcal{T}(F, F^\star)$ is conservative over $\mathcal{T}(F)$. ⊣

COROLLARY 9.19 ($\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$ Elimination Lemma). *For each $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{VC}}})$ formula φ^+ there is a $\Sigma_1^B(\mathcal{L}_A^2)$ formula φ such that $\widehat{\mathbf{VC}} \vdash \varphi^+ \leftrightarrow \varphi$.*

PROOF. We apply Theorem 9.18 for $\mathcal{T} = \mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$ and $\mathcal{L} = \mathcal{L}_{\mathbf{FAC}^0}$. The hypothesis that every $\Sigma_0^B(\mathcal{L}_{\mathbf{FAC}^0})$ formula is equivalent in $\overline{\mathbf{V}}^0$ to a Σ_1^B formula is from Lemma 5.74, the facts that F and $F^\star$ are both Σ_1^B-definable in $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$, and that $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$ proves (165) are established in Lemma 9.9. ⊣

The next corollary follows from Corollaries 9.16 and 9.19.

COROLLARY 9.20. (a) *A function is in* **FC** *iff it is $\Sigma_1^B(\mathcal{L}_A^2)$-definable in $\widehat{\mathbf{VC}}$.*
(b) *A relation is in* **C** *iff it is $\Delta_1^B(\mathcal{L}_A^2)$-definable in $\widehat{\mathbf{VC}}$.*

Now we are able to prove Theorem 9.10.

PROOF OF THEOREM 9.10. The proof is straightforward using Theorem 9.13 and Corollary 9.20. ⊣

9B.3. The theory $\overline{\mathbf{VC}}$. Here we introduce $\overline{\mathbf{VC}}$, another universal conservative extension of $\mathbf{VC}$ and $\widehat{\mathbf{VC}}$. Its language $\mathcal{L}_{\mathbf{FC}}$ contains symbols for all string functions in $\mathbf{FC}$. The defining axioms for functions in $\mathcal{L}_{\mathbf{FC}}$ are based on their $\mathbf{AC}^0$-reductions to the function $F_{\mathbf{C}}$. Recall the function F' and its quantifier-free defining axiom (219).

DEFINITION 9.21 ($\overline{\mathbf{VC}}$). *$\mathcal{L}_{\mathbf{FC}}$ is the smallest set containing $\mathcal{L}_{\mathbf{FAC}^0} \cup \{F'\}$ and satisfying the following condition: for each open formula $\varphi(z, \vec{x}, \vec{X})$ over $\mathcal{L}_{\mathbf{FC}}$ and $\mathcal{L}^2_A$-term $t = t(\vec{x}, \vec{X})$, there is a string function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FC}}$ with defining axiom* (85)

$$F_{\varphi(z),t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X}) \tag{223}$$

$\overline{\mathbf{VC}}$ is the universal theory over $\mathcal{L}_{\mathbf{FC}}$ whose axioms consist of the axioms of $\overline{\mathbf{V}}^0$, (219) for F', and the above defining axioms for the functions $F_{\varphi(z),t}$.

Note that Lemma 8.19 and Theorem 8.20 apply to $\overline{\mathbf{VC}}$. The next theorem follows from Theorem 8.15.

THEOREM 9.22. *$\overline{\mathbf{VC}}$ is a conservative extension of $\widehat{\mathbf{VC}}$ and $\mathbf{VC}$.*

PROOF. It suffices to show that $\overline{\mathbf{VC}}$ is a conservative extension of $\widehat{\mathbf{VC}}$, because $\widehat{\mathbf{VC}}$ is a conservative extension of $\mathbf{VC}$.

First, $\overline{\mathbf{VC}}$ is an extension of $\widehat{\mathbf{VC}}$ because all axioms of $\widehat{\mathbf{VC}}$ are axioms of $\overline{\mathbf{VC}}$. Thus $\overline{\mathbf{VC}}$ is the union

$$\overline{\mathbf{VC}} = \bigcup_{i \geq 0} \mathcal{T}_i$$

where $\mathcal{T}_0 = \widehat{\mathbf{VC}}$ and for $i \geq 1$ each $\mathcal{T}_i$ is obtained from $\mathcal{T}_{i-1}$ by adding a new function F_i of the form $F_{\varphi(z),t}$ with defining axiom (223), where φ is a quantifier-free formula in the language of $\mathcal{T}_{i-1}$. We will show that F_i is definable in $\mathcal{T}_{i-1}$ and that $\mathcal{T}_{i-1}(F_i)$ proves (223). This implies that $\mathcal{T}_i$ is a conservative extension of $\mathcal{T}_{i-1}$, and hence $\overline{\mathbf{VC}}$ is a conservative extension of $\widehat{\mathbf{VC}}$.

Let $\mathcal{L}_i$ denote the vocabulary of $\mathcal{T}_i$. We prove by induction on $i \geq 0$ that $\mathcal{T}_i$ proves $\Sigma^B_0(\mathcal{L}_i)$-$\mathbf{COMP}$. Then the fact that F_{i+1} is definable (in fact, $\Sigma^B_0(\mathcal{L})$-definable) in $\mathcal{T}_i$ follows from the lemma below, whose proof is straightforward:

LEMMA 9.23. *Let $\mathcal{T}$ be an extension of $\mathbf{V}^0(Row)$ with vocabulary $\mathcal{L}$ such that $\mathcal{T}$ proves $\Sigma^B_0(\mathcal{L})$-$\mathbf{COMP}$. Let $F_{\varphi(z),t}$ be a function Σ^B_0-definable from $\mathcal{L}$ with defining axiom* (223) *where φ is any $\Sigma^B_0(\mathcal{L})$ formula. Then both $F_{\varphi(z),t}$ and $F^\star_{\varphi,t}$ are $\Sigma^B_0(\mathcal{L})$-definable in $\mathcal{T}$, and $\mathcal{T}(F_{\varphi(z),t}, F^\star_{\varphi,t})$ proves* (165) *for $F_{\varphi(z),t}$ and $F^\star_{\varphi,t}$:*

$$\forall i < b,\ F^\star_{\varphi,t}(b, \vec{Z}, \vec{X})^{[i]} = F_{\varphi(z),t}((Z_1)^i, \ldots, (Z_k)^i, X_1^{[i]}, \ldots, X_n^{[i]})$$

Both base case and induction step are easy applications of Theorem 8.15 using Lemma 9.23 above. ⊣

Lemma 9.24. *The theory* $\overline{\mathbf{VC}}$ *proves the axiom schemes* $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$-**COMP**, $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$-**IND** *and* $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$-**MIN**.

Proof. By Corollary 5.8 it suffices to show that $\overline{\mathbf{VC}}$ proves the $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$-**COMP** axioms. This follows from the proof of Theorem 9.22 above, but it also has simple proof as follows.

Let $\varphi(z,\vec{x},\vec{X})$ be a $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$ formula. By Lemma 8.19 there is a quantifier-free $\mathcal{L}_{\mathbf{FC}}$-formula $\varphi^+(z,\vec{x},\vec{X})$ so that

$$\overline{\mathbf{VC}} \vdash \varphi^+(z,\vec{x},\vec{X}) \leftrightarrow \varphi(z,\vec{x},\vec{X})$$

Let $Y = F_{\varphi^+,y}(\vec{x},\vec{X})$, where $F_{\varphi^+,y}$ is the string function of $\mathcal{L}_{\mathbf{FC}}$ with defining axiom

$$F_{\varphi^+,y}(\vec{x},\vec{X})(z) \leftrightarrow z < y \wedge \varphi^+(z,\vec{x},\vec{X})$$

Then

$$\overline{\mathbf{VC}} \vdash |Y| \le y \wedge \forall z < y\big(Y(z) \leftrightarrow \varphi(z,\vec{x},\vec{X})\big)$$

Hence $\overline{\mathbf{VC}}$ proves the comprehension axiom for φ. ⊣

Theorem 9.25. (a) *A string function is in* **FC** *if and only if it is represented by a string function symbol in* $\mathcal{L}_{\mathbf{FC}}$.

(b) *A relation is in* **C** *iff it is represented by an open formula of* $\mathcal{L}_{\mathbf{FC}}$, *iff it is represented by a* $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$ *formula.*

Proof. Part (b) follows from (a), so below we will prove (a). First, we prove by induction using Definition 9.1 that every string function in **FC** is represented by a string function in $\mathcal{L}_{\mathbf{FC}}$. The base case is simple because F and every function in $\mathcal{L}_{\mathbf{FAC}^0}$ are in $\mathcal{L}_{\mathbf{FC}}$. For the induction step, suppose that $G(\vec{x},\vec{X})$ is Σ_0^B-definable from $\mathcal{L} = \{F_1 = F', F_2, \dots, F_n\}$, and that each $F_i \in \mathcal{L}_{\mathbf{FC}}$, for $1 \le i \le n$. By definition, there is a $\Sigma_0^B(\mathcal{L})$ formula φ and an $\mathcal{L}_A^2$-term t such that

$$G(\vec{x},\vec{X})(z) \leftrightarrow z \le t \wedge \varphi(z,\vec{x},\vec{X})$$

By Lemma 8.19 there is a quantifier-free $\mathcal{L}_{\mathbf{FC}}$-formula $\varphi^+(z,\vec{x},\vec{X})$ that is equivalent (in **VC**) to $\varphi(z,\vec{x},\vec{X})$. Hence

$$G(\vec{x},\vec{X})(z) \leftrightarrow z \le t \wedge \varphi^+(z,\vec{x},\vec{X})$$

So G is equal to the function $F_{\varphi^+,t}$ in $\mathcal{L}_{\mathbf{FC}}$.

For the other direction, we prove by induction (using Definition 9.21) that every string function in $\mathcal{L}_{\mathbf{FC}}$ represents a string function in **FC**. For the base case, the functions in $\mathcal{L}_{\mathbf{FAC}^0} \cup \{F'\}$ obviously represent functions in **FC**. For the induction step, let $F_{\varphi(z),t}$ be a function in $\mathcal{L}_{\mathbf{FC}}$, where all functions $F_1, F_2, \dots, F_n$ in φ represent functions in **FC**. Then $F_{\varphi(z),t}$ is $\mathbf{AC}^0$-reducible to $F_1, F_2, \dots, F_n$, hence $F_{\varphi(z),t}$ also represents a function in **FC**. ⊣

The next result is a corollary of Theorem 9.18.

COROLLARY 9.26 ($\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$ Elimination Lemma). *Every* $\Sigma_0^B(\mathcal{L}_{\mathbf{FC}})$ *formula* φ^+ *is equivalent in* $\overline{\mathbf{VC}}$ *to a* $\Sigma_1^B(\mathcal{L}_A^2)$ *formula* φ.

PROOF. Let the theories $\mathcal{T}_i$ and their vocabularies $\mathcal{L}_i$ be as in the proof of Theorem 9.22. We prove by induction on i that for each $\Sigma_0^B(\mathcal{L}_i)$ formula φ^+ there is a $\Sigma_1^B(\mathcal{L}_A^2)$ formula φ such that $\mathcal{T}_i$ proves equivalent to φ^+.

The base case is Lemma 9.19. For the induction step, suppose that the statement is true for some $i \geq 0$. The statement for $i+1$ is proved by applying Theorem 9.18 for $\mathcal{T} = \mathcal{T}_i$ and $\mathcal{L} = \mathcal{L}_i$. The hypothesis of Theorem 9.18 is satisfied by Lemma 9.23 and the fact (from the proof of Theorem 9.22) that $\mathcal{T}_i$ proves $\Sigma_0^B(\mathcal{L}_i)$-**COMP**. $\dashv$

The characterization of $\mathbf{C}$ by $\overline{\mathbf{VC}}$ follows from the above results (and Herbrand Theorem and Theorem 5.60).

COROLLARY 9.27. (a) *A function is in* $\mathbf{FC}$ *iff it is* $\Sigma_1^B(\mathcal{L}_{\mathbf{FC}})$*-definable in* $\overline{\mathbf{VC}}$ *iff it is* $\Sigma_1^B(\mathcal{L}_A^2)$*-definable in* $\overline{\mathbf{VC}}$.

(b) *A relation is in* $\mathbf{C}$ *iff it is* $\Delta_1^B(\mathcal{L}_{\mathbf{FC}})$*-definable in* $\overline{\mathbf{VC}}$ *iff it is* $\Delta_1^B(\mathcal{L}_A^2)$*-definable in* $\overline{\mathbf{VC}}$.

Note that Theorem 9.10 also follows from Theorem 9.22 (that $\overline{\mathbf{VC}}$ is a universal conservative extension of $\mathbf{VC}$), Theorem 9.25 and Corollary 9.26.

9B.4. Obtaining theories for the classes of interest. The results so far in this chapter show how to obtain a theory $\mathbf{VC}$ for each class $\mathbf{C}$ mentioned in the preface. In fact, for each class $\mathbf{C}$ of interest, there is a polytime Turing machine M such that the function

$$F(X) = \text{“the computation of } \mathsf{M} \text{ on input } X\text{”}$$

is $\mathbf{AC}^0$ complete for $\mathbf{C}$. For example, for $\mathbf{C} = \mathbf{P}$ we can take the machine that computes $F_{MCV}(a, G, E)$ by computing inductively the bits of Y that satisfies (156) on page 192.

The $\Sigma_0^B(\mathcal{L}_A^2)$ defining axiom (217) for F can be obtained using the following $\mathbf{AC}^0$ functions (which can be eliminated by Lemma 5.74):

- $\mathit{Init}_{\mathsf{M}}(X)$ is the initial configuration of M given input X,
- $\mathit{Next}_{\mathsf{M}}(U)$ is the next configuration of the configuration U, and
- $\mathit{Cut}(t, Z)$ is the set of all elements of Z that are less than t with defining axiom (96) (page 133):

$$\mathit{Cut}(t, Z) = \{z : z \in Z \wedge z < t\}$$

Let t be an $\mathcal{L}_A^2$ term that bounds the running time of M. We have

$$F(X) = Y \leftrightarrow \big(|Y| \leq \langle t, t\rangle \wedge Y^{[0]} = \mathit{Cut}(t, \mathit{Init}_{\mathsf{M}}(X)) \wedge \\ \forall x < t,\, Y^{[x+1]} = \mathit{Cut}(t, \mathit{Next}_{\mathsf{M}}(Y^{[x]}))\big)$$

By eliminating $Init_{\mathsf{M}}$, $Next_{\mathsf{M}}$ and Cut, the above formula has the required form (217) and it is easy to prove in $\mathbf{V}^0$ the uniqueness for Y (by proving by induction on $x \le t$ that the rows $Y^{[x]}$ are unique).

Although the axiom (218) states the existence of the value for the function $F^\star$, for each class $\mathbf{C}$ that we consider we are able to simplify (218) so that it only states the existence of the value for F. Thus we will need to prove the analogue of Lemma 8.10, i.e., that $F^\star$ is definable using the simplified axiom and $\mathbf{V}^0$. It turns out that the proof is unique for each class that we consider.

In the remaining of this chapter we will develop instances of $\mathbf{VC}$ as discussed here without referring to any specific machine M; they are implicit in the additional defining axioms of the instances that we introduce.

9C. Theories for $\mathbf{TC}^0$

The class $\mathbf{TC}^0$ (see definition in Section 9C.1 below) is the smallest class known to contain problems such as sorting, integer multiplication and division (when the input integer arguments are presented in binary). Here we define $\mathbf{VTC}^0$, $\widehat{\mathbf{VTC}^0}$ and $\overline{\mathbf{VTC}}^0$ (Section 9C.2) in the style of the theories $\mathbf{VC}$, $\widehat{\mathbf{VC}}$ and $\overline{\mathbf{VC}}$ in Section 9B. In Section 9C.3 we define the bounded number recursion (BNR). Then in Section 9C.4 we use number summation, a special case of BNR, to characterize $\mathbf{TC}^0$ and develop $\mathbf{VTC}^0\mathbf{V}$ in the style of $\mathbf{VPV}$ (see Section 8B). This is another universal conservative extension of $\mathbf{VTC}^0$. We formalize a proof of the Pigeonhole Principle in $\mathbf{VTC}^0$ in Section 9C.5. Finally we define the string multiplication function $X \times Y$ and prove its properties in $\mathbf{VTC}^0$ in Section 9C.6.

In Chapter 10 we will prove the Propositional Translation Theorem for $\mathbf{VTC}^0$.

9C.1. The Class $\mathbf{TC}^0$. The class nonuniform $\mathbf{TC}^0$ (or $\mathbf{TC}^0/poly$) consists of languages that are accepted by a family of polynomial-size constant-depth circuits whose gates can be Boolean gates or the *majority* gates. A majority gate has unbounded fan-in and which outputs 1 if and only if the number of 1 inputs is more than the number of 0 inputs. We are interested in $\mathbf{FO}$-uniform $\mathbf{TC}^0$ (or just $\mathbf{TC}^0$) where the family can be described by an $\mathbf{FO}$-formula (Section 4A).

Instead of the majority gates, $\mathbf{TC}^0$ can be equivalently defined using *counting* gates or *threshold* gates. A counting gate C_k (for $k \in \mathbb{N}$) has unbounded fan-in, and $\mathsf{C}_k(x_1, x_2, \dots, x_n)$ is true if and only if there are exactly k inputs x_i that are true. Similarly, for $k \in \mathbb{N}$, the threshold gate Th_k has unbounded fan-in, and $\mathsf{Th}_k(x_1, x_2, \dots, x_n)$ is true if and only if there are at least k inputs that are true.

There are several equivalent characterizations of $\mathbf{TC}^0$ in descriptive complexity theory (see also Section 4A for the descriptive characterization of $\mathbf{AC}^0$). They are obtained by augmenting the first-order logic **FO** with quantifiers that correspond to the majority, counting or threshold gates described above. For example, let $\mathcal{L}_{FO(M)}$ denote the set of formulas over the vocabulary $\mathcal{L}_{FO}$ (41):

$$[0, max; X, BIT, \leq, =]$$

where a new quantifier M is allowed. The meaning of this quantifier is as follows: for a finite structure $\mathcal{M}$ and a $\mathcal{L}_{FO(M)}$ formula $Mx\varphi(x)$,

$$\mathcal{M} \vdash Mx\varphi(x)$$

iff $\mathcal{M} \vdash \varphi(a)$ for at least half of the elements a in the universe of $\mathcal{M}$. Let

$$\mathbf{FO}(M) = \{L \mid L = L(\varphi) \text{ for some } \mathcal{L}_{FO(M)}\text{-sentence } \varphi\}$$

and define $\mathbf{FO}(\mathit{COUNT})$, $\mathbf{FO}(\mathit{THRESHOLD})$ similarly. Then it can be shown that

$$\mathbf{TC}^0 = \mathbf{FO}(M) = \mathbf{FO}(\mathit{COUNT}) = \mathbf{FO}(\mathit{THRESHOLD})$$

$\mathbf{TC}^0$ can also be defined using other computation models, such as the so-called Threshold Turing machines, but we will not go into detail here. The proposition below uses the notion of $\mathbf{AC}^0$-reducibility defined in Section 9A and is based on the fact that $\mathbf{TC}^0 = \mathbf{FO}(\mathit{COUNT})$, or in other words, *numones* is $\mathbf{AC}^0$-complete for $\mathbf{TC}^0$. (Recall the function $\mathit{numones}(y, X)$ defined on page 143: $\mathit{numones}(y, X)$ is the number of elements of X that are $< y$.)

PROPOSITION 9.28. *$\mathbf{TC}^0$ is the $\mathbf{AC}^0$ closure of numones. $\mathbf{FTC}^0$ is the $\mathbf{FAC}^0$ closure of numones.*

Below we will introduce the theories $\mathbf{VTC}^0$, $\widehat{\mathbf{VTC}}^0$ and $\overline{\mathbf{VTC}}^0$. The above proposition will be used to justify the association between these theories and $\mathbf{TC}^0$.

9C.2. The theories $\mathbf{VTC}^0$, $\widehat{\mathbf{VTC}}^0$ and $\overline{\mathbf{VTC}}^0$. The theory $\mathbf{VTC}^0$ is similar to **VP** (Definition 8.1) in the sense that it is axiomatized by $\mathbf{V}^0$ and a defining axiom for the function *numones* (which is $\mathbf{AC}^0$ complete for $\mathbf{TC}^0$). The following defining axiom for *numones* is given in (107) on page 143:

$$\mathit{numones}(y, X) = z \leftrightarrow z \leq y \wedge \exists Z \leq 1 + \langle y, y\rangle,\ (Z)^0 = 0 \wedge (Z)^y = z \wedge$$
$$\forall u < y,\ (X(u) \supset (Z)^{u+1} = (Z)^u + 1) \wedge (\neg X(u) \supset (Z)^{u+1} = (Z)^u)$$

(Recall that $(Z)^u$ denotes $\mathit{seq}(u, Z)$, the u-th element of the bounded sequence of numbers coded by Z, see Definition 5.56.)

Let $\delta_{NUM}(y, X, Z)$ be the $\Sigma_0^B(\mathcal{L}_A^2)$ formula obtained from (224) below by eliminating *seq* as described in Lemmas 5.40 and 5.74:

$$(224) \quad (Z)^0 = 0 \;\wedge$$
$$\forall u < y\big((X(u) \supset (Z)^{u+1} = (Z)^u + 1) \wedge (\neg X(u) \supset (Z)^{u+1} = (Z)^u)\big)$$

Informally, we can think of Z as a "counting sequence" for X:

$$(Z)^u = z \leftrightarrow numones(u, X) = z \qquad \text{for } u \le y$$

Definition 9.29 ($\mathbf{VTC}^0$). *Let NUMONES denote*

$$\forall X \forall y \exists Z \delta_{NUM}(y, X, Z)$$

The theory $\mathbf{VTC}^0$ has vocabulary $\mathcal{L}_A^2$ and is axiomatized by the axioms of $\mathbf{V}^0$ and NUMONES.

In $\mathbf{V}^0$, *NUMONES* is equivalent to the same axiom with $\exists Z$ replaced by the bounded quantifier $\exists Z \le 1 + \langle y, y\rangle$. Hence, $\mathbf{VTC}^0$ is a polynomial-bounded theory. Also, $\mathbf{VTC}^0$ is finitely axiomatizable because $\mathbf{V}^0$ is.

To develop $\widehat{\mathbf{VTC}^0}$, we will use the "string version" of *numones*, denoted by *Numones*, that has the defining axiom:

$$Numones(y, X) = Z \leftrightarrow |Z| \le 1 + \langle y, y\rangle \wedge \delta_{NUM}(y, X, Z)$$

Proposition 9.28 is true if we replace *numones* by *Numones*.

Recall the notion of aggregate functions in Definition 8.9. The next lemma shows that $\mathbf{VTC}^0$ is indeed an instance of the family $\mathbf{VC}$ (because it shows that the existence of the value of $Numones^\star$ is provable in $\mathbf{VTC}^0$).

Lemma 9.30. *The functions numones, Numones, and $Numones^\star$ are Σ_1^B-definable (and hence also Σ_1^1-definable) in $\mathbf{VTC}^0$, and the theory $\mathbf{VTC}^0(Row, Numones, Numones^\star)$ proves*

$$(225) \qquad \forall i < b, Numones^\star(b, Y, X)^{[i]} = Numones((Y)^i, X^{[i]})$$

Proof. The fact that *numones* and *Numones* are provably total in $\mathbf{VTC}^0$ is obvious. We will show that $Numones^\star$ is Σ_1^B-definable in $\mathbf{VTC}^0$. The fact that $\mathbf{VTC}^0(Row, Numones, Numones^\star)$ proves (225) will be clear from the proof below.

For convenience, we use the functions *Row* and *seq* in the defining axiom for $Numones^\star$ described below; it is straightforward to eliminate *Row* and *seq* from the axiom (Lemmas 5.52 and 5.74).

Intuitively we need to show that $\mathbf{VTC}^0(Row, seq)$ proves the existence of Z such that for all $i < b$, $Z^{[i]}$ is the "counting sequence" for $X^{[i]}$:

$$\mathbf{VTC}^0(Row, seq) \vdash \exists Z \forall i < b\; \delta_{NUM}((Y)^i, X^{[i]}, Z^{[i]})$$

The idea is to (i) concatenate the first $(Y)^i$ bits of the rows $X^{[i]}$, for $i < b$, to form a "big" string X', (ii) obtain the counting sequence Z'

for X', and (iii) extract the desired array of counting sequences $Z^{[i]}$ from Z'.

We will use $|Y|$ as an upper bound for $(Y)^i$, for $i < b$. Thus, let X' be defined by

$$X'(i|Y| + x) \leftrightarrow x < (Y)^i \wedge X^{[i]}(x), \quad \text{for } i < b.$$

In other words, for $i < b$, the bit string

$$X'(i|Y|)\, X'(i|Y| + 1)\, \ldots\, X'(i|Y| + (Y)^i - 1)$$

is a copy of

$$X^{[i]}(0)\, X^{[i]}(1)\, \ldots\, X^{[i]}((Y)^i - 1)$$

and $X'(i|Y| + (Y)^i), \ldots, X'((i+1)|Y| - 1)$ are all 0. Therefore, for $u \leq (Y)^i$,

$$\mathit{numones}(u, X^{[i]}) = \mathit{numones}(i|Y| + u, X') - \mathit{numones}(i|Y|, X')$$

Let Z' be such that $\delta_{NUM}(b|Y|, X', Z')$ holds, i.e., Z' is the "counting sequence" for X'. Then

$$\mathit{numones}(u, X^{[i]}) = z \leftrightarrow (Z')^{i|Y|+u} \mathbin{\dot{-}} (Z')^{i|Y|} = z$$

Thus,

$$\begin{aligned}&\mathit{Numones}^\star(b, Y, X) = Z \leftrightarrow \\ &\qquad\qquad \forall i < b \forall u \leq (Y)^i \big((Z^{[i]})^u = (Z')^{i|Y|+u} \mathbin{\dot{-}} (Z')^{i|Y|}\big)\end{aligned}$$

It follows easily that $\mathit{Numones}^\star(b, Y, X)$ is provably total in $\mathbf{VTC}^0$. $\dashv$

Exercise 9.31. *Similar to the aggregate of a string function, we can define the aggregate of a number function as follows. Suppose that $f(x_1, \ldots, x_k, X_1, \ldots, X_n)$ is a polynomially bounded number function, i.e., for some $\mathcal{L}^2_A$ term t,*

$$f(\vec{x}, \vec{X}) \leq t(\vec{x}, |\vec{X}|)$$

Then $f^\star(b, \vec{Z}, \vec{X})$ is the polynomially bounded string *function that satisfies*

$$|f^\star(b, \vec{Z}, \vec{X})| \leq \langle b, 1 + t\rangle$$

and

(226)

$$f^\star(b, \vec{Z}, \vec{X})(w) \leftrightarrow \exists u < b,\ w = \langle u, f((Z_1)^u, \ldots, (Z_k)^u, X_1^{[u]}, \ldots, X_n^{[u]})\rangle$$

Show that $\mathit{numones}^\star$ is provably total in $\mathbf{VTC}^0$.

To define $\widehat{\mathbf{VTC}^0}$ we need a quantifier-free defining axiom for *Numones*. (Formally we will not change the defining axiom for *Numones* but will introduce a function *Numones*$'$ that has the same value as *Numones* and has a quantifier-free defining axiom.) So let $\delta'_{NUM}(y, X, Z)$ be the quantifier-free $\mathcal{L}_{\mathbf{FAC}^0}$-formula

$$(Z)^0 = 0 \land$$
$$u < y \supset \big((X(u) \supset (Z)^{u+1} = (Z)^u + 1) \land (\neg X(u) \supset (Z)^{u+1} = (Z)^u)\big)$$

Notice that $\overline{\mathbf{V}}^0 \vdash \delta_{NUM}(y, X, Z) \leftrightarrow \delta'_{NUM}(y, X, Z)$. Let $\mathit{Numones}'(y, X)$ be defined by

$$\mathit{Numones}'(y, X) = Z \leftrightarrow |Z| \le 1 + \langle y, y\rangle \land \delta'_{NUM}(y, X, Z) \tag{227}$$

Thus $\mathit{Numones}'(y, X) = \mathit{Numones}(y, X)$, but they have different defining axioms.

Definition 9.32 ($\widehat{\mathbf{VTC}^0}$). *$\widehat{\mathbf{VTC}^0}$ is the universal theory over the vocabulary $\mathcal{L}_{\widehat{\mathbf{VTC}^0}} = \mathcal{L}_{\mathbf{FAC}^0} \cup \{\mathit{Numones}'\}$ and is axiomatized by the axioms of $\overline{\mathbf{V}}^0$ and* (227).

We define $\overline{\mathbf{VTC}}^0$ using the number function *numones′* instead of the string function *Numones′*. Here *numones′* has the same value as *numones* but it has the following quantifier-free defining axioms:

$$\mathit{numones}'(0, X) = 0 \tag{228}$$

$$X(z) \supset \mathit{numones}'(z + 1, X) = \mathit{numones}'(z, X) + 1 \tag{229}$$

$$\neg X(z) \supset \mathit{numones}'(z + 1, X) = \mathit{numones}'(z, X). \tag{230}$$

Definition 9.33. *$\mathcal{L}_{\mathbf{FTC}^0}$ is the smallest set that contains $\mathcal{L}_{\mathbf{FAC}^0} \cup \{\mathit{numones}'\}$ such that for every quantifier-free $\mathcal{L}_{\mathbf{FTC}^0}$-formula $\varphi(z, \vec{x}, \vec{X})$ and every $\mathcal{L}^2_A$-term $t = t(\vec{x}, \vec{X})$, there is a string function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FTC}^0}$ with defining axiom* (85)*:*

$$F_{\varphi(z),t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \land \varphi(z, \vec{x}, \vec{X}) \tag{231}$$

$\overline{\mathbf{VTC}}^0$ is the theory over $\mathcal{L}_{\mathbf{FTC}^0}$ and is axiomatized by the axioms of $\overline{\mathbf{V}}^0$ together with (228), (229) *and* (230) *for numones′, and* (231) *for each function $F_{\varphi(z),t}$.*

It is easy to see that $\mathit{Numones} = F_{\varphi(z),t}$ for some $F_{\varphi(z),t} \in \mathcal{L}_{\mathbf{FTC}^0}$. On the other hand, it is also easy to see that $\mathit{numones} = |T|$ for some term $T \in \mathcal{L}_{\widehat{\mathbf{VTC}^0}}$. Therefore the results in Section 9B apply for $\widehat{\mathbf{VTC}^0}$ and $\overline{\mathbf{VTC}}^0$. We summarize the Definability Theorems for these theories as follows:

Theorem 9.34. *Here either $\mathcal{L}$ is $\mathcal{L}_{\widehat{\mathbf{VTC}^0}}$ and $\mathcal{T}$ is $\widehat{\mathbf{VTC}^0}$, or $\mathcal{L}$ is $\mathcal{L}_{\mathbf{FTC}^0}$ and $\mathcal{T}$ is $\overline{\mathbf{VTC}}^0$.*

(a) *A function is in $\mathbf{FTC}^0$ iff it is represented by a term in $\mathcal{L}_{\widehat{\mathbf{VTC}^0}}$. A string function is in $\mathbf{FTC}^0$ iff it is in $\mathcal{L}_{\mathbf{FTC}^0}$. A relation is in $\mathbf{TC}^0$ iff it is represented by an open (or a Σ^B_0) formula in $\mathcal{L}$.*

(b) *For every* $\Sigma_0^B(\mathcal{L})$ *formula* φ^+ *there is a* Σ_1^B*-formula* φ *such that* $\mathcal{T} \vdash \varphi^+ \leftrightarrow \varphi$.
(c) $\mathcal{T}$ *proves* $\Sigma_0^B(\mathcal{L})$-**COMP**.
(d) *Both* $\widehat{\mathbf{VTC}}^0$ *and* $\overline{\mathbf{VTC}}^0$ *are universal conservative extensions of* $\mathbf{VTC}^0$.
(e) *A function is in* $\mathbf{FTC}^0$ *iff it is* Σ_1^B*-definable in* $\mathbf{VTC}^0$ *iff it is* Σ_1^B*-definable in* $\mathcal{T}$.
(f) *A relation is in* $\mathbf{TC}^0$ *iff it is* Δ_1^B*-definable in* $\mathbf{VTC}^0$ *iff it is* Δ_1^B*-definable in* $\mathcal{T}$.

COROLLARY 9.35. $\mathbf{VTC}^0$ *is a proper extension of* $\mathbf{V}^0$. *In fact,* $\mathbf{VTC}^0$ *is not* Σ_0^B*-conservative over* $\mathbf{V}^0$.

PROOF. The first sentence follows from the second, which is true because $\mathbf{VTC}^0$ proves the Pigeonhole Principle (Section 9C.5 below), while this principle is not provable in $\mathbf{V}^0$ (Corollary 7.21).

Another way of proving the the first sentence is to use Theorem 9.34 (e) above. Recall that the number function *parity*(X), which is the parity of the total number of elements in X (Section 5E.1), is not in $\mathbf{FAC}^0$. Hence $\mathbf{V}^0$ does not prove the defining axiom for *parity*. On the other hand, *parity* is in $\mathbf{FTC}^0$, since it can be easily computed using *numones*:

$$\mathit{parity}(X) = \mathit{numones}(|X|, X) \mod 2$$

So $\mathbf{VTC}^0$ proves the defining axiom for *parity*. ⊣

9C.3. Number Recursion and Number Summation. The number recursion operation produces a new number function from existing number functions. This operation is similar to *limited recursion* (Definition 6.15) but the latter defines new *string functions* from existing string functions. It is useful in characterizing **FL** and a number of its subclasses (Later we will use this to develop the theory $\mathbf{VTC}^0\mathbf{V}$, an analogue of **VPV** (Section 8B). See Sections 9D.3, 9D.6, 9E.4, 9F.4 and 9C.4 below). The number summation operation is a special instance of number recursion and is useful in characterizing $\mathbf{FTC}^0$.

DEFINITION 9.36 (Number Recursion). *A number function* $f(y, \vec{x}, \vec{X})$ *is obtained by* number recursion *from* $g(\vec{x}, \vec{X})$ *and* $h(y, z, \vec{x}, \vec{X})$ *if*

$$f(0, \vec{x}, \vec{X}) = g(\vec{x}, \vec{X}) \tag{232}$$

$$f(y+1, \vec{x}, \vec{X}) = h(y, f(y, \vec{x}, \vec{X}), \vec{x}, \vec{X}) \tag{233}$$

If further $f(y, \vec{x}, \vec{X}) < t(y, \vec{x}, \vec{X})$, *then we also say that* f *is obtained by* t*-bounded number recursion (*t*-BNR) from* g *and* h. *In particular, if* f *is polynomially bounded then we say that* f *is obtained from* g *and* h *by* polynomial-bounded number recursion *(pBNR).*

Definition 9.37 (Number Summation). *For a number function* $f(y, \vec{x}, \vec{X})$, *define the number function* $\mathsf{sum}_f(y, \vec{x}, \vec{X})$ *by*

$$\mathsf{sum}_f(y, \vec{x}, \vec{X}) = \sum_{z=0}^{y} f(z, \vec{x}, \vec{X})$$

The function sum_f *is said to be defined from* f by number summation, *or just* summation.

Theorem 9.38. *A function is in* $\mathbf{FTC}^0$ *iff it is obtained from* $\mathbf{FAC}^0$ *functions by finitely many application of composition, string comprehension, and number summation iff it is obtained from* $\mathbf{FAC}^0$ *by* $\mathbf{AC}^0$ *reduction and number summation.*

Proof. By Theorem 9.7 it suffices to prove that a function is in $\mathbf{FTC}^0$ iff it is obtained from $\mathbf{FAC}^0$ by $\mathbf{AC}^0$ reduction and number summation.

For the ONLY IF direction, by Proposition 9.28 we need to show only that *numones* can be obtained by number summation from $\mathbf{AC}^0$ function. This fact is straightforward:

$$\mathit{numones}(y, X) = \sum_{z=0}^{y} f_X(z, X)$$

where f_X is the $\mathbf{AC}^0$ function defined by

$$f_X(z, X) = w \leftrightarrow \big((X(z) \supset w = 1) \wedge (\neg X(z) \supset w = 0)\big) \tag{234}$$

We prove the other direction by induction on the number of applications of the number summation operation. The base case (number summation is not used) is obvious. For the induction step, it suffices to show that sum_f is $\mathbf{AC}^0$ reducible to f and *numones*. Thus define a string W that contains the right number of bits:

$$W(xa + v) \leftrightarrow x \le y, v < f(x)$$

for some $a > \mathit{max}(\{f(x) \ : \ x < y\})$. Then it is easy to verify that $\mathsf{sum}_f(y) = \mathit{numones}((y + 1)a, W)$. $\dashv$

9C.4. The theory $\mathbf{VTC}^0\mathbf{V}$. We define here the theory $\mathbf{VTC}^0\mathbf{V}$, another universal conservative extension of $\mathbf{VTC}^0$. The language of $\mathbf{VTC}^0\mathbf{V}$ contains a symbol for each functions in $\mathbf{FTC}^0$, but here, except for the $\mathbf{FAC}^0$ functions, they are defined using the number summation scheme (based on Theorem 9.38).

Definition 9.39 ($\mathcal{L}_{\mathbf{VTC}^0\mathbf{V}}$). *The language* $\mathcal{L}_{\mathbf{VTC}^0\mathbf{V}}$ *is the smallest set that contains* $\mathcal{L}_{\mathbf{FAC}^0}$ *such that*

1) *for every number function* $f(y, \vec{x}, \vec{X})$ *in* $\mathcal{L}_{\mathbf{VTC}^0\mathbf{V}}$ *the function* $\mathsf{sum}_f(y, \vec{x}, \vec{X})$ *is also in* $\mathcal{L}_{\mathbf{VTC}^0\mathbf{V}}$ *with defining axioms*

(235)

$$\mathsf{sum}_f(0, \vec{x}, \vec{X}) = 0 \wedge \mathsf{sum}_f(y + 1, \vec{x}, \vec{X}) = \mathsf{sum}_f(y, \vec{x}, \vec{X}) + f(y, \vec{x}, \vec{X})$$

2) *for every $\mathcal{L}_A^2$-term t and quantifier-free $\mathcal{L}_{\mathbf{VTC^0V}}$-formula φ the function $F_{\varphi(z),t}$ is in $\mathcal{L}_{\mathbf{VTC^0V}}$ with defining axiom* (85)*:*

$$F_{\varphi(z),t}(\vec{x},\vec{X})(z) \leftrightarrow z < t(\vec{x},\vec{X}) \wedge \varphi(z,\vec{x},\vec{X}) \tag{236}$$

The next results follow from Theorem 9.38.

COROLLARY 9.40. (a) *A function is in* $\mathbf{FTC}^0$ *iff it is represented by a term in* $\mathcal{L}_{\mathbf{VTC^0V}}$.

(b) *A relation is in* $\mathbf{TC}^0$ *iff it is represented by an open (or a Σ_0^B) formula in* $\mathcal{L}_{\mathbf{VTC^0V}}$.

DEFINITION 9.41. *The theory* $\mathbf{VTC^0V}$ *has vocabulary* $\mathcal{L}_{\mathbf{VTC^0V}}$ *and axioms those of* $\overline{\mathbf{V}}^0$ *and* (235) *and* (236) *for the functions* sum_f *and* $F_{\varphi(z),t}$*, respectively.*

The following Lemma is proved in the same way as Lemma 5.71.

LEMMA 9.42. $\mathbf{VTC^0V}$ *proves*
$\Sigma_0^B(\mathcal{L}_{\mathbf{VTC^0V}})$-$\mathbf{COMP}$, $\Sigma_0^B(\mathcal{L}_{\mathbf{VTC^0V}})$-$\mathbf{IND}$ *and* $\Sigma_0^B(\mathcal{L}_{\mathbf{VTC^0V}})$-$\mathbf{MIN}$.

THEOREM 9.43. $\mathbf{VTC^0V}$ *is a universal conservative extension of* $\mathbf{VTC}^0$.

PROOF. First, by definition, $\mathbf{VTC^0V}$ extends $\overline{\mathbf{V}}^0$. As noted in the proof of Theorem 9.38 (the ONLY IF direction), *numones*$' = \mathsf{sum}_{f_X}$ where f_X is defined in (234):

$$f_X(z,X) = w \leftrightarrow \big((X(z) \supset w = 1) \wedge (\neg X(z) \supset w = 0)\big)$$

It is easy to see that $\mathbf{VTC^0V}$ proves the defining axioms (228), (229), (230) for *numones*$'$. It follows that $\mathbf{VTC^0V}$ extends $\mathbf{VTC}^0$.

Now we show that $\mathbf{VTC^0V}$ is conservative over $\mathbf{VTC}^0$. Since $\mathbf{VTC^0V}$ extends $\mathbf{VTC}^0$, we have

$$\mathbf{VTC^0V} = \bigcup_{i \geq 0} \mathcal{T}_i$$

where $\mathcal{T}_0 = \mathbf{VTC}^0$ and each $\mathcal{T}_{i+1}$ is obtained from $\mathcal{T}_i$ by adding the defining axiom for a new function sum_f or $F_{\varphi(z),t}$. We show that $\mathcal{T}_{i+1}$ is conservative over $\mathcal{T}_i$ by showing that the new function of $\mathcal{T}_{i+1}$ is definable in $\mathcal{T}_i$.

Let $\mathcal{L}_i$ denote the vocabulary of $\mathcal{T}_i$. Consider the case where the new function in $\mathcal{T}_{i+1}$ has the form $F_{\varphi(z),t}$ for some quantifier-free $\mathcal{T}_i$-formula φ and $\mathcal{L}_A^2$-term t. It is easy to see that $F_{\varphi(z),t}$ is definable in $\mathcal{T}_i$ if

$$\mathcal{T}_i \vdash \Sigma_0^B(\mathcal{L}_i)\text{-}\mathbf{COMP} \tag{237}$$

Similarly, suppose that the new function in $\mathcal{T}_{i+1}$ has the form sum_f for some number function $f \in \mathcal{L}_i$. Following the IF direction of the proof of Theorem 9.38, the fact that sum_f is definable in $\mathcal{T}_i$ also follows from (237). In fact, using (237) above it can be shown $\mathsf{sum}_f^\star$ is Σ_1^B-definable in $\mathcal{T}_i$. This is left as an exercise. Recall the notion of aggregate function for a number function in Exercise 9.31.

EXERCISE 9.44. *Suppose that* (237) *holds. Show that both* sum_f *and* $\mathsf{sum}^\star_f$ *are definable in* $\mathcal{T}_i$.

It remains to prove (237). The proof is by induction on i. The base case is Theorem 9.34 (c). The induction step follows from Theorem 8.15 (using Lemma 9.23) and the following corollary (using Exercise 9.44). ⊣

The next result is a corollary of Theorem 8.15.

COROLLARY 9.45. *Let* $\mathcal{T}$ *be a theory with vocabulary* $\mathcal{L}$ *which extends* $\mathbf{V}^0$*(Row) and proves* $\Sigma^B_0(\mathcal{L})$-$\mathbf{COMP}$. *Suppose that* f *and* $f^\star$ *are definable in* $\mathcal{T}$ *(Definition 5.26) and* $\mathcal{T}(f, f^\star)$ *proves (226). Then* $\mathcal{T}(f)$ *proves* $\Sigma^B_0(\mathcal{L} \cup \{f\})$-$\mathbf{COMP}$.

PROOF. Let F be the string function that contains at most one element and $|F| = f$:

$$(f = 0 \supset |F| = 0) \wedge (f > 0 \supset (|F| = f \wedge \forall z < f(F(z) \leftrightarrow z + 1 = f)))$$

Then both F and $F^\star$ are definable in $\mathcal{T}$. So the corollary follows easily from Theorem 8.15. ⊣

The next corollary follows from Theorem 9.18 in the same way:

COROLLARY 9.46 (Second Elimination Theorem). *Let* $\mathcal{T}$ *be a theory with vocabulary* $\mathcal{L}$ *which extends* $\mathbf{V}^0$*(Row) and proves* $\Sigma^B_0(\mathcal{L})$-$\mathbf{COMP}$. *Suppose that* f *and* $f^\star$ *are* Σ^B_1*-definable in* $\mathcal{T}$ *(Definition 5.26) and* $\mathcal{T}(f, f^\star)$ *proves* (226). *Suppose also that every* $\Sigma^B_0(\mathcal{L})$ *formula is equivalent in* $\mathcal{T}$ *to a* $\Sigma^B_1(\mathcal{L}^2_A)$ *formula. Then every* $\Sigma^B_0(\mathcal{L} \cup \{f\})$ *formula is equivalent in* $\mathcal{T}(f)$ *to a* $\Sigma^B_1(\mathcal{L}^2_A)$ *formula.*

From the above corollary and Theorem 9.18 we have:

COROLLARY 9.47. *For each* $\Sigma^B_0(\mathcal{L}_{\mathbf{VTC^0V}})$ *formula* φ^+ *there is a* Σ^B_1 *formula* φ *so that* $\mathbf{VTC}^0\mathbf{V} \vdash \varphi^+ \leftrightarrow \varphi$.

The definability theorems for $\mathbf{VTC}^0\mathbf{V}$ are as follows.

COROLLARY 9.48. (a) *A function is in* $\mathbf{FTC}^0$ *iff it is* Σ^B_1*-definable in* $\mathbf{VTC}^0\mathbf{V}$.
(b) *A relation is in* $\mathbf{TC}^0$ *iff it is* Δ^B_1*-definable in* $\mathbf{VTC}^0\mathbf{V}$.

PROOF. The corollary either from Theorem 9.43, or directly from Theorem 9.38 and Lemma 9.42 using the Herbrand Theorem. ⊣

9C.5. Proving the Pigeonhole Principle in $\mathbf{VTC}^0$. We present a proof of the Pigeonhole Principle (Section 7A.2) in $\mathbf{VTC}^0$. As mentioned in the proof of Corollary 7.21 this implies that $\mathbf{VTC}^0$ is a proper extension of $\mathbf{V}^0$. In the next chapter we will show that each Σ^B_0 theorem of $\mathbf{VTC}^0$ translates into a family of tautologies having polysize bounded depth **PTK** proofs. It will follow that the family **PHP** (Definition 7.12) has polysize bounded depth **PTK** proofs. This separates bounded depth **PK** from bounded depth **PTK**. On the other hand, we will show (Section 9E.3) that $\mathbf{VNC}^1$ extends $\mathbf{VTC}^0$. Therefore PHP is also provable in $\mathbf{VNC}^1$. The Propositional Translation Theorem for $\mathbf{VNC}^1$ then allows us to derive a theorem of Buss that **PHP** has polysize **Frege** proofs.

The formula $\mathbf{PHP}(a, X)$ is defined in Example 7.18 as follows:

(238)
$$\forall x \leq a \exists y < a X(x,y) \supset \exists x \leq a \exists z \leq a \exists y < a (x \neq z \wedge X(x,y) \wedge X(z,y))$$

THEOREM 9.49. $\mathbf{VTC}^0 \vdash \mathbf{PHP}(a, X)$.

PROOF. Since $\mathbf{VTC}^0(numones)$ is conservative over $\mathbf{VTC}^0$, it suffices to show that

$$\mathbf{VTC}^0(numones) \vdash \mathbf{PHP}(a, X)$$

We prove by contradiction, so assume that

$$\forall x \leq a \exists y < a X(x,y) \tag{239}$$

and

$$\forall x \leq a \forall z \leq a \forall y < a((x \neq z \wedge X(x,y)) \supset \neg X(z,y)) \tag{240}$$

Let P be the set of pigeons:

$$P = \{0, 1, 2, \ldots, a\}$$

Let $\varphi(x, y)$ be the following formula which asserts that y is the first hole that pigeon x occupies:

$$\varphi(x,y) \equiv x \leq a \wedge y < a \wedge X(x,y) \wedge \forall v < y \neg X(x,v)$$

Then by (239) and (240) φ defines an injective function from P into the set of holes $\{0, 1, 2, \ldots, a-1\}$, i.e., $\mathbf{VTC}^0$ proves

$$\forall x \leq a \exists ! y < a \varphi(x,y) \wedge \forall x \leq a \forall z \leq a \forall y < a((x \neq z \wedge \varphi(x,y)) \supset \neg\varphi(z,y))$$

Let H be the image of P (defined using Σ_0^B-**COMP**):

$$|H| \leq a \wedge \forall y < a(H(y) \leftrightarrow \exists x \leq a \varphi(x,y))$$

Then it is easy to see that φ defines a bijection between P and H (i.e., φ satisfies the premise of (241) below for $b = a+1$). Lemma 9.50 below shows that P and H have the same cardinality:

$$\mathbf{VTC}^0(numones) \vdash numones(a+1, P) = numones(a+1, H)$$

However, it is easy to show that $numones(a+1, P) = a+1$ and $numones(a+1, H) \leq a$, a contradiction. $\dashv$

For the following lemma, informally we show that if there is a bijection between two sets P and H that is described by a Σ_0^B formula $\varphi(x, y)$, then provably in $\mathbf{VTC}^0(numones)$ the sets have the same cardinality.

LEMMA 9.50. *For any $\Sigma_0^B(numones)$ formula $\varphi(x, y)$, the following is a theorem of $\mathbf{VTC}^0(numones)$:*

$$\begin{aligned}(241) \quad \big(&\forall x < b(P(x) \supset \exists! y < b(\varphi(x,y) \wedge H(y)) \\ &\wedge \forall y < b(H(y) \supset \exists! x < a(\varphi(x,y) \wedge P(x))\big) \\ &\supset numones(b, P) = numones(b, H)\end{aligned}$$

PROOF. Let Z be the array whose rows $Z^{[i]}$ are the images of the initial segments $Cut(i, P)$ of P under the bijection, i.e.,

$$\forall i < b \forall y < b\big(Z^{[i]}(y) \leftrightarrow \exists x < i(\varphi(x,y) \wedge P(x))\big)$$

Since φ is $\Sigma_0^B(numones)$ and $\mathbf{VTC}^0(numones) \vdash \Sigma_0^B(numones)$-**COMP** (by Theorem 9.34 (c)), $\mathbf{VTC}^0(numones)$ proves the existence of such Z.

Now we prove by induction on $i < b$ that

$$(242) \qquad numones(i, P) = numones(b, Z^{[i]})$$

It will follow that $numones(b, P) = numones(b, Z^{[b]})$, and since $Z^{[b]} = Cut(b, H)$ we have $numones(b, P) = numones(b, Cut(b, H))$, so

$$numones(b, P) = numones(b, H)$$

The base case ($i = 0$) is obvious. Consider the induction step, assume that (242) is true for some $i \geq 0$. We show that it is also true for $i + 1$. There are two cases: either $i \in P$ or $i \notin P$.

First, suppose that $i \in P$, then $numones(i+1, P) = numones(i, P) + 1$. Let $j \in Z^{[i+1]}$ be such that $\varphi(i, j)$ holds. Then $j \notin Z^{[i]}$, and it can be shown by induction on y that

$$numones(y, Z^{[i+1]}) = \begin{cases} numones(y, Z^{[i]}) & \text{if } y \leq j \\ numones(y, Z^{[i]}) + 1 & \text{if } y \geq j + 1 \end{cases}$$

Hence $numones(b, Z^{[i+1]}) = numones(b, Z^{[i]}) + 1$, and we are done.

The other case is similar. ⊣

9C.6. Defining String Multiplication in $\mathbf{VTC}^0$. Recall that $bin(X)$ is the integer value associated with a string X (46) (page 82):

$$bin(X) = \sum_{i \in X} X(i) 2^i$$

The string multiplication function, $X \times_2 Y$ (or simply $X \times Y$) is defined so that

$$bin(X \times Y) = bin(X) \times bin(Y)$$

Exercise 6.11 shows that this function is Σ_1^B-definable in $\mathbf{V}^1$. Here we will show that it is actually Σ_1^B-definable in $\mathbf{VTC}^0$ by formalizing in $\mathbf{VTC}^0$ a $\mathbf{TC}^0$ algorithm that computes $X \times Y$. Furthermore, $\mathbf{VTC}^0$ proves usual properties of this function, such as commutativity, distributivity over $X + Y$, etc.

Notice that the "school" algorithm described in Exercise 6.11 is a polytime algorithm. The main component of this algorithm is the polytime process that computes the sum of all rows of the table $X \otimes Y$. The $\mathbf{TC}^0$ algorithm for $X \times Y$ is obtained by replacing this polytime process by a uniform family of $\mathbf{TC}^0$ circuits. First, we outline this $\mathbf{TC}^0$ algorithm and formalize it in $\mathbf{VTC}^0$ by showing that the function *Sum* defined below is Σ_1^B-definable in $\mathbf{VTC}^0$.

For the formalizations, recall the string functions $\varnothing, S(X), X+Y$ given in Example 5.42, $Cut(x, X)$ on page 133, and the number function $\lceil \log(x+1) \rceil$ in Exercise 3.55.

9C.6.1. *Adding n Strings.* Suppose that we are to add n integers written as n binary strings, each of length m. The idea is to write these binary strings as rows in a table of n rows and m columns, then divide the columns of the table in to blocks of ℓ columns each (for some parameter ℓ to be determined later) so that the sums of the rows in each blocks can be easily computed in $\mathbf{TC}^0$, and that the desired result can be computed from these sums by a $\mathbf{TC}^0$ circuit.

More precisely, let $\ell = \lceil \log(n+1) \rceil$, then in each block B_i, each row can be seen as a number with value $\leq 2^\ell - 1$. Therefore the sum of the rows in B_i is at most

$$n(2^\ell - 1) < 2^{2\ell}$$

and hence has a binary representation of length at most 2ℓ. It is important that this sum can be defined as the number of 1-bits in a long string easily obtained from B_i.

Now let b_i be the sum of the rows in the block B_i, then the required sum is

$$\sum_i 2^{i\ell} b_i \tag{243}$$

Write each b_i as a binary string of length exactly 2ℓ (add preceding 0's if necessary). Then

$$b_0 + 2^{2\ell} b_2 + 2^{4\ell} b_4 + \dots \tag{244}$$

is simply the concatenation of the strings $b_0, b_2, b_4, \dots$, and similarly for

$$2^\ell b_1 + 2^{3\ell} b_3 + 2^{5\ell} b_5 + \dots \tag{245}$$

As a result, (243) can be computed in $\mathbf{AC}^0$ by adding the above two sums.

9C.6.2. *Formalization.* For the formalization, we will use the function *numones* and some $\mathbf{AC}^0$ functions. It will be clear that the functions defined here belong to $\mathcal{L}_{\mathbf{FTC}^0}$.

Suppose that the n input strings are given as the rows $Z^{[0]}, \dots, Z^{[n-1]}$ in an array Z. We will define in $\mathbf{VTC}^0$ the function $Sum(n, m, Z)$ that satisfies

(246) $\quad Sum(0, m, Z) = \varnothing$

(247) $\quad Sum(n+1, m, Z) = Sum(n, m, Z) + Cut(m, Z^{[n]})$

where $Cut(x, X)$ is the first x bits of X (96):

$$Cut(x, X)(z) \leftrightarrow z < x \wedge X(z)$$

We define the columns of Z as strings using the function *Transpose* defined as follows:

(248) $$Transpose(n, m, X) = Y \leftrightarrow \big(|Y| \le \langle m, n\rangle \wedge \forall z < \langle m, n\rangle \big(Y(z) \leftrightarrow \exists i < m \exists j < n (z = \langle i, j\rangle \wedge X(j, i))\big)\big)$$

Thus, let $V = Transpose(n, m, Z)$, then the sum of the bits in column i of Z is

$$c_i = numones(n, V^{[i]})$$

Let

$$\ell = \lceil \log(n+1) \rceil, \qquad k = \lceil m/2\ell \rceil$$

Note that ℓ is an $\mathbf{AC}^0$ function of n (Exercise 3.55). We want the sequence B: $(B)^0 = b_0, (B)^1 = b_1, \ldots, (B)^{2k} = b_{2k}$ (see (243)) so that

$$(B)^i = \sum_{j=0}^{\ell-1} 2^j c_{i\ell+j}$$

We show how to define each $(B)^i$ by a Σ_1^B formula; it will follow from Exercise 9.31 that B is also Σ_1^B-definable in $\mathbf{VTC}^0$.

To define $(B)^i$, it suffices to define a string U that contains exactly $(B)^i$ 1-bits. Then

$$(B)^i = numones(|U|, U)$$

Notice that $c_i \le n$ for $0 \le i < m$. The string U consists of

$$1 + 2^1 + 2^2 + \cdots + 2^{\ell-1} = 2^\ell - 1$$

substrings, each has n bits so that for $j < \ell$, 2^j substrings contains exactly $c_{i\ell+j}$ 1-bits. Thus U can be defined as follows:

$$|U| \le 2^\ell n \wedge \forall j < \ell \forall u < 2^j \forall v < n \ \big(U((2^j-1)n + un + v) \leftrightarrow v < c_{i\ell+j}\big)$$

Now the sum (244) is formally defined as a string L with bit definition:

$$|L| \le 2k\ell \wedge \forall x < 2k\ell \ \big(L(x) \leftrightarrow \exists i < k \exists y < 2\ell (x = 2i\ell + y \wedge BIT(y, (B)^{2i}))\big)$$

(where BIT is the Δ_0 formula defined in Section 3C.3). Similarly, the sum (245), denoted by H, is defined as follows:

$$|H| \le 2k\ell \wedge$$
$$\forall x < 2k\ell \ \big(H(x) \leftrightarrow \exists i < k \exists y < 2\ell (x = (2i+1)\ell + y \wedge BIT(y, (B)^{2i+1}))\big)$$

Finally,

$$Sum(n, m, Z) = L + H$$

LEMMA 9.51. *The theory* $\overline{\mathbf{VTC}}^0$ *proves* (246) *and* (247).

PROOF. Argue in $\overline{\mathbf{VTC}}^0$: suppose that $n = 0$ then $\ell = 0$. So it is easy to see that $Sum(0, m, Z) = \varnothing$.

We prove (247) by induction on m. The base case ($m = 0$) is obvious. For the induction step, we need the exercise below. Here $Shift(x, y)$ is the string obtained by shifting all bits of the binary representation of y by x positions to the left:

$$Shift(x, y) = U \leftrightarrow \big(|U| \le x + \lceil \log(y + 1)\rceil \wedge \\ \forall z < x + \lceil \log(y + 1)\rceil (U(z) \leftrightarrow \exists i < \lceil \log(y + 1)\rceil BIT(i, y))\big)$$

EXAMPLE 9.52 (Provable in $\overline{\mathbf{V}}^0$).

$$Shift(x, y + z) = Shift(x, y) + Shift(x, z)$$

EXERCISE 9.53. *Show that it is provable in* $\overline{\mathbf{VTC}}^0$ *that*

$$Sum(n, m + 1, Z) = Sum(n, m, Z) + Shift(m, c_m)$$

where c_m *is the sum of the first* n *bits in column* m *of* Z*:*

$$c_m = numones(n, V^{[m]}) \qquad \text{where } V = Transpose(n, m + 1, Z)$$

Argue in $\overline{\mathbf{VTC}}^0$, the induction step follows from Exercise 9.53 as follows. Suppose that we need to prove (247) for $m + 1$. Let c'_m be the sum of the first $(n + 1)$ bits in column m of Z:

$$c'_m = numones(n + 1, Transpose(n + 1, m, Z)^{[m]})$$

Then

$$c'_m = c_m + Z^{[n]}(m) \tag{249}$$

By Exercise 9.53 we need to prove

$$Sum(n + 1, m, Z) + Shift(m, c'_m) = \\ Sum(n, m, Z) + Shift(m, c_m) + Cut(m + 1, Z^{[n]})$$

By the induction hypothesis,

$$Sum(n + 1, m, Z) = Sum(n, m, Z) + Cut(m, Z^{[n]})$$

Also,

$$Cut(m + 1, Z^{[n]}) = Cut(m, Z^{[n]}) + Shift(m, Z^{[n]})$$

So we need to show that

$$Shift(m, c'_m) = Shift(m, c_m) + Shift(m, Z^{[n]}(m))$$

This follows from Example 9.52 and (249). ⊣

9C.6.3. *Defining* $X \times Y$. To define $X \times Y$ we use the table $X \otimes Y$ given in Exercise 6.11 and can be equivalently defined as follows:

$$X \otimes Y = Z \leftrightarrow \big(|Z| \leq \langle |Y|, |X| + |Y| \rangle \wedge \\ \forall i < |Y| ((\neg Y(i) \supset Z^{[i]} = \varnothing) \wedge (Y(i) \supset Z^{[i]} = \mathit{Shift}(i, X)))\big)$$

where $\mathit{Shift}(x, Y)$ is the string obtained from Y by shifting all bits x positions to the left

$$\mathit{Shift}(x, Y) = Z \leftrightarrow \\ |Z| \leq x + |Y| \wedge \forall z < x + |Y| (Z(z) \leftrightarrow \exists u < |Y|, Y(u) \wedge z = x + u)$$

(Notice that $\mathit{Shift}(x, y)$ and $\mathit{Shift}(x, Y)$ have different arity, so even though they have the same name, their meaning will be clear from context.)

We define

$$X \times Y = \mathit{Sum}(|Y|, |X| + |Y|, X \otimes Y)$$

Lemma 9.54. $\overline{\mathbf{VTC}}^0 \vdash X \times Y = Y \times X$.

Proof. By definition, we need to show that

$$\mathit{Sum}(|Y|, |X| + |Y|, X \otimes Y) = \mathit{Sum}(|X|, |X| + |Y|, Y \otimes X)$$

Therefore it suffices to show that the columns of $X \otimes Y$ and $Y \otimes X$ have the same number of 1-bits:

$$numones(|Y|, V^{[i]}) = numones(|X|, W^{[i]}) \tag{250}$$

for $i < |X| + |Y|$ and

$$V = \mathit{Transpose}(|X|, |X| + |Y|, X \otimes Y)$$
$$W = \mathit{Transpose}(|Y|, |X| + |Y|, Y \otimes X)$$

Notice that there is a bijection between $V^{[i]}$ and $W^{[i]}$ defined by

$$V^{[i]}(z) \leftrightarrow W^{[i]}(i \dot{-} z) \qquad \text{for } z \leq i$$

because

$$V^{[i]}(z) \leftrightarrow Y(z) \wedge X(i \dot{-} z) \qquad \text{and} \qquad W^{[i]}(z) \leftrightarrow X(z) \wedge Y(i \dot{-} z)$$

So the conclusion follows from Lemma 9.50. ⊣

Lemma 9.55. $\overline{\mathbf{VTC}}^0 \vdash X \times (Y + Z) = X \times Y + X \times Z$.

Proof. We will prove by induction on $i \leq |X|$ that

$$\mathit{Cut}(i, X) \times (Y + Z) = \mathit{Cut}(i, X) \times Y + \mathit{Cut}(i, X) \times Z \tag{251}$$

The lemma follows by letting $i = |X|$.

For the base case, $i = 0$, we have $\mathit{Cut}(0, X) = \varnothing$. So this case follows from Exercise 9.57 (a) below and Lemma 9.54.

For the induction step, suppose that (251) holds for some $i \geq 0$. We prove it for $i + 1$. There are two cases: either $i \in X$ or $i \notin X$. In the second case $Cut(i+1, X) = Cut(i, X)$ so the conclusion if obvious. Thus we consider the case where $i \in X$. We have

$$Cut(i+1, X) = Cut(i, X) + \{i\}$$

We need the following results:

Exercise 9.56. *Show that the following are theorems of* $\overline{\mathbf{VTC}}^0$*:*
(a) $|X| \leq i \supset (X + \{i\}) \times Y = X \times Y + \{i\} \times Y$.
(b) $\{i\} \times X = \{x + i \ : \ x \in X\}$.
(c) $\{i\} \times (Y + Z) = \{i\} \times Y + \{i\} \times Z$.

Now $|Cut(i, X)| \leq i$. Using Exercises 9.56 and 5.44 and the induction hypothesis we have

$$\begin{aligned}
&Cut(i+1, X) \times (Y + Z) \\
&\qquad = (Cut(i, X) + \{i\}) \times (Y + Z) \\
&\qquad = Cut(i, X) \times (Y + Z) + \{i\} \times (Y + Z) \\
&\qquad = (Cut(i, X) \times Y + Cut(i, X) \times Z) + (\{i\} \times Y + \{i\} \times Z) \\
&\qquad = (Cut(i, X) \times Y + \{i\} \times Y) + (Cut(i, X) \times Z + \{i\} \times Z) \\
&\qquad = (Cut(i, X) + \{i\}) \times Y + (Cut(i, X) + \{i\}) \times Z \\
&\qquad = Cut(i+1, X) \times Y + Cut(i+1, X) \times Z
\end{aligned}$$

So (251) holds for $i + 1$. ⊣

Exercise 9.57. *Show that the following are theorems of* $\overline{\mathbf{VTC}}^0$*:*
(a) $X \times \varnothing = \varnothing$.
(b) $X \times S(Y) = (X \times Y) + X$.

Exercise 9.58. *Show that*

$$\overline{\mathbf{VTC}}^0 \vdash (X \times Y) \times Z = X \times (Y \times Z)$$

9D. Theories for $\mathbf{AC}^0(m)$ and ACC

In this section we will develop the theories associated with the classes $\mathbf{AC}^0(m)$ and their union **ACC**. These classes lie between $\mathbf{AC}^0$ and $\mathbf{TC}^0$. First, in Section 9D.1 we will define the classes. Then in Section 9D.2 we define the theories $\mathbf{V}^0(2)$, $\widehat{\mathbf{V}^0(2)}$ and $\overline{\mathbf{V}^0(2)}$ for $\mathbf{AC}^0(2)$. Functions in $\mathbf{FAC}^0(2)$ can be characterized by a bounded number recursion scheme (see Section 9C.3) and in Section 9D.3 we will use this to develop $\mathbf{VAC}^0(2)\mathbf{V}$, another universal conservative extension of $\mathbf{V}^0(2)$. A discrete version of the Jordan Curve Theorem can be proved in $\mathbf{V}^0(2)$ and we will present the formalization in Section 9D.4. Then in Section 9D.5 we will define theories for other classes $\mathbf{AC}^0(m)$. Finally, the class $\mathbf{FAC}^0(6)$ also has a recursion characterization using the BNR scheme, and we will use this to develop $\mathbf{VAC}^0(6)\mathbf{V}$ in Section 9D.6.

9D.1. The Classes $\mathbf{AC}^0(m)$ and ACC. For each $m \in \mathbb{N}$, $m \geq 2$, the class nonuniform/uniform $\mathbf{AC}^0(m)$ are defined just as nonuniform/uniform $\mathbf{TC}^0$ but using the *modulo m* gates instead of the majority gates. A modulo m gate has unbounded fan-in and outputs 1 if and only if the total number of 1 inputs is exactly 1 modulo m. Also,

$$\mathbf{ACC} = \bigcup_{i \geq 2} \mathbf{AC}^0(m)$$

Obviously, $\mathbf{AC}^0 \subseteq \mathbf{AC}^0(m)$. Furthermore, the relation $PARITY$ (Sections 4A and 5E.1):

$$PARITY(X) \text{ iff } X \text{ contains an odd number of elements}$$

is in $\mathbf{AC}^0(2)$. Since $PARITY \notin \mathbf{AC}^0$, it follows that $\mathbf{AC}^0 \subsetneq \mathbf{AC}^0(2)$.

It is easy to show that for $2 \leq m_1 < m_2 \in \mathbb{N}$, $m_1 \mid m_2$,

$$\mathbf{AC}^0(m_1) \subseteq \mathbf{AC}^0(m_2)$$

On the other hand, let $MODULO_p(X)$ be the relation

$$MODULO_p(X) \text{ iff the number of elements of } X \text{ is} = 1 \ mod(p)$$

Then

$$MODULO_p(X) \in \mathbf{AC}^0(p)$$

and it has been shown that for any two distinct prime numbers p, q,

$$MODULO_p \notin \mathbf{AC}^0(q)$$

As a result,

$$\mathbf{AC}^0(p) \not\subseteq \mathbf{AC}^0(q)$$

Also, the modulo m gates can be easily simulated by the threshold gates. It follows also that

$$\mathbf{AC}^0 \subsetneq \mathbf{AC}^0(p) \subsetneq \mathbf{ACC} \subseteq \mathbf{TC}^0$$

(the last inclusion is because the counting gates can simulate the *modulo m* gate, for any m). On the contrary, it is an open problem whether $\mathbf{AC}^0(m) \subsetneq \mathbf{ACC}$ for composite $m \in \mathbb{N}$. In fact, it is not known whether $\mathbf{AC}^0(6) \subsetneq \mathbf{NP}$.

In descriptive complexity, uniform $\mathbf{AC}^0(m)$ (or just $\mathbf{AC}^0(m)$) can be characterized using the $mod(m)$ quantifier [7]. Here we use the fact that the following function is (Turing) $\mathbf{AC}^0$ complete for $\mathbf{AC}^0(m)$:

(252) $$mod_m(x, Y) = numones(x, Y) \mod m$$

The "string version" of this function, called $Mod_m(x, Y)$, is the sequence of the values of $mod_m(z, Y)$ for $z < x$:

(253)
$$Mod_m(x, Y) = Z \leftrightarrow \big(|Z| \leq 1 + \langle x, m\rangle \wedge \forall z < x((Z)^z = mod_m(z, Z))\big)$$

PROPOSITION 9.59. *A relation is in* $\mathbf{AC}^0(m)$ *iff it is* $\mathbf{AC}^0$*-reducible to* mod_m *iff it is* $\mathbf{AC}^0$*-reducible to* Mod_m*. A function is in* $\mathbf{FAC}^0(m)$ *iff it is* $\mathbf{AC}^0$*-reducible to* mod_m *iff it is* $\mathbf{AC}^0$*-reducible to* Mod_m*.*

9D.2. The theories $\mathbf{V}^0(2)$, $\widehat{\mathbf{V}^0(2)}$ and $\overline{\mathbf{V}^0(2)}$. Consider the formula $\delta_{parity}(x, Y, Z)$ that asserts that for $1 \le z \le x$, $Z(z)$ holds iff there is an odd number of bits in $Y(0)Y(1)\dots Y(z \dot- 1)$:

$$\delta_{parity}(x, Y, Z) \equiv \neg Z(0) \land \forall z < x(Z(z+1) \leftrightarrow (Z(z) \oplus Y(z)))$$

We will use this formula to define the function Mod_2 and the theory $\mathbf{V}^0(2)$.

DEFINITION 9.60 ($\mathbf{V}^0(2)$). *The theory* $\mathbf{V}^0(2)$ *has vocabulary* $\mathcal{L}_A^2$ *and axioms those of* $\mathbf{V}^0$ *and*

$$\exists Z \le x + 1 \ \delta_{parity}(x, Y, Z)$$

EXERCISE 9.61. *Show that* $\mathbf{V}^0(2)$ *proves*

$$\exists Z \forall i < b\big(\neg Z^{[i]}(0) \land \forall z < (X)^i (Z^{[i]}(z+1) \leftrightarrow (Z^{[i]}(z) \oplus Y^{[i]}(z)))\big)$$

The function Mod_2 is also called *Parity* and satisfies

$$Parity(x, Y) = Z \leftrightarrow \big(|Z| \le x + 1 \land \delta_{parity}(x, Y, Z)\big)$$

In order to develop the universal theory $\widehat{\mathbf{V}^0(2)}$ in the style of $\mathbf{VP}$, we need a quantifier-free defining axiom for *Parity*. Here we use:

$$\begin{aligned}(254)\quad & Parity(x, Y) = Z \leftrightarrow \big(|Z| \le x + 1 \land \\ & \quad \neg Z^{[i]}(0) \land \big(z < (X)^i \supset (Z^{[i]}(z+1) \leftrightarrow (Z^{[i]}(z) \oplus Y^{[i]}(z)))\big)\big)\end{aligned}$$

It follows from the above exercise that $Parity^\star$ is Σ_1^B-definable in $\mathbf{V}^0(2)$, $\mathbf{V}^0(2)$ proves (254) and $\mathbf{V}^0(2)(Row, Parity, Parity^\star)$ proves

$$\forall i < b(Parity^\star(b, X, Y)^{[i]} = Parity((X)^i, Y^{[i]}))$$

The universal theory $\widehat{\mathbf{V}^0(2)}$ is defined to be $\overline{\mathbf{V}}^0(Parity)$ with the defining axiom (254) for *Parity*. Its vocabulary $\mathcal{L}_{\mathbf{FAC}^0} \cup \{Parity\}$ is denoted by $\mathcal{L}_{\widehat{\mathbf{V}^0(2)}}$. The theory $\overline{\mathbf{V}^0(2)}$ is defined as follows. Its language, $\mathcal{L}_{\mathbf{FAC}^0(2)}$ is the smallest set the contains $\mathcal{L}_{\widehat{\mathbf{V}^0(2)}}$ such that for every $\mathcal{L}_A^2$-term t and $\mathcal{L}_{\mathbf{FAC}^0(2)}$-formula φ, there is a string function $F_{\varphi(z),t}$ with defining axiom (85). Then $\overline{\mathbf{V}^0(2)}$ is axiomatized by the axioms of $\overline{\mathbf{V}}^0$ together with (254) for *Parity* and (85) for each function $F_{\varphi(z),t}$.

The results of Section 9B applied to the theories just defined as for the case of the theories $\mathbf{VTC}^0$, $\widehat{\mathbf{VTC}}^0$ and $\overline{\mathbf{VTC}}^0$. Here we have the Definability Theorems for the theories in this section as corollaries:

COROLLARY 9.62. *Here either* $\mathcal{L}$ *is* $\mathcal{L}_{\widehat{\mathbf{V}^0(2)}}$ *and* $\mathcal{T}$ *is* $\widehat{\mathbf{V}^0(2)}$, *or* $\mathcal{L}$ *is* $\mathcal{L}_{\mathbf{FAC}^0(2)}$ *and* $\mathcal{T}$ *is* $\overline{\mathbf{V}^0(2)}$.

(a) *A function is in* $\mathbf{FAC}^0(2)$ *iff it is represented by a term in* $\mathcal{L}_{\widehat{\mathbf{V}^0(2)}}$ *(and for string functions) iff it is represented by a symbol in* $\mathcal{L}_{\mathbf{FAC}^0(2)}$. *A relation is in* $\mathbf{AC}^0(2)$ *iff it is represented by an open (or* Σ_0^B*) formula in* $\mathcal{L}$.

(b) $\overline{\mathbf{V}^0(2)}$ *is a universal conservative extension of* $\widehat{\mathbf{V}^0(2)}$, *which is in turn a universal conservative extension of* $\mathbf{V}^0(2)$.

(c) *Every* $\Sigma_0^B(\mathcal{L})$ *formula is equivalent in* $\mathcal{T}$ *to a* $\Sigma_1^B(\mathcal{L}_A^2)$ *formula.*

(d) $\mathcal{T}$ *proves* $\Sigma_0^B(\mathcal{L})$-**COMP**

(e) *A function is in* $\mathbf{FAC}^0(2)$ *iff it is* Σ_1^B*-definable in* $\mathbf{V}^0(2)$ *iff it is* Σ_1^B*-definable in* $\mathcal{T}$.

(f) *A relation is in* $\mathbf{AC}^0(2)$ *iff it is* Δ_1^B*-definable in* $\mathbf{V}^0(2)$ *iff it is* Δ_1^B*-definable in* $\mathcal{T}$.

The bijective pigeonhole principle **BPHP** states that there is no bijection between $(a+1)$ "pigeons" and a "holes". Formally, **BPHP** is the Σ_0^B formula

$$\mathbf{BPHP}(a,X) \equiv (\forall x \le a \exists z < a\, X(x,z)) \supset (\neg INJ(a,X) \vee \neg SUR(a,X))$$

where

$$INJ(a,X) \equiv \forall x \le a \forall y \le a \forall z < a((X(x,z) \wedge X(y,z)) \supset x = y)$$
$$SUR(a,X) \equiv \forall z < a \exists x \le a X(x,z)$$

Ajtai proves that the family of tautologies BPHP does not have polynomial size **bPK** proofs. It follows that **BPHP** is not a theorem of $\mathbf{V}^0$. On the other hand, it is relatively easy to show that **BPHP** is provable in $\mathbf{V}^0(2)$, as in the next exercise.

EXERCISE 9.63. *Show that* $\mathbf{V}^0(2)$ *proves* $\mathbf{BPHP}(a,X)$. *(Hint: it suffice to show that* $\widehat{\mathbf{V}^0(2)}$ *proves* $\mathbf{BPHP}(a,X)$. *Reason by way of contradiction in* $\widehat{\mathbf{V}^0(2)}$, *define an array* Z *such that for* $x < a$, *the row* $Z^{[x]}$ *is the set of all pigeons that occupy holes* $0,1,\ldots,x$. *Prove by induction on* x *that*

$$x < a \supset (Parity(a, Z^{[x]})(a) \leftrightarrow x \mod 2 = 0)$$

See also Lemma 9.50.)

COROLLARY 9.64. $\mathbf{V}^0 \subsetneq \mathbf{V}^0(2)$.

PROOF. As for Corollary 9.35, there are two way of proving this corollary. One is to use the lemma above and the fact (following from Ajtai's Theorem) that **BPHP** is not provable in $\mathbf{V}^0$. The other is to use the fact that *Parity* is definable in $\mathbf{V}^0(2)$ but not in $\mathbf{V}^0$ (because it is not in $\mathbf{FAC}^0$). $\dashv$

Another interesting theorem of $\mathbf{V}^0(2)$ is a discrete version of the Jordan Curve Theorem which will be discussed in Section 9D.4.

9D.3. The theory $\mathbf{VAC}^0(2)\mathbf{V}$. The universal theory $\mathbf{VAC}^0(2)\mathbf{V}$ is defined in the same way as $\mathbf{VTC}^0\mathbf{V}$ and $\mathbf{VPV}$. Its vocabulary symbols for every function in $\mathbf{FAC}^0(2)$. Their defining axioms are based on the recursion theoretic characterization of $\mathbf{FAC}^0(2)$ using a bounded number recursion (BNR) scheme as shown in Theorem 9.65 below. (Recall also the BNR from Section 9C.3.)

Theorem 9.65. *$\mathbf{FAC}^0(2)$ is equal to the closure of $\mathbf{FAC}^0$ under $\mathbf{AC}^0$ reduction and 2-BNR and also equal the closure of $\mathbf{FAC}^0$ under composition, string comprehension and 2-BNR.*

Proof. By Theorem 9.7 it suffices to show that a function is in $\mathbf{FAC}^0(2)$ iff it can be obtained from $\mathbf{FAC}^0$ functions by finitely many applications of $\mathbf{AC}^0$ reduction and 2-BNR.

The ONLY IF direction follows easily from the fact that the function $mod_2(x, Y)$ (252) is obtained from the function f_X ((234) on page 275) by 2-BNR.

For the IF direction, we prove by induction on the number of the applications of 2-BNR. The base case (no application of 2-BNR) is obvious. For the induction step, suppose that $f(y, \vec{x}, \vec{X})$ is obtained from $\mathbf{FAC}^0(2)$ functions g, h by 2-BNR as in Definition 9.36:

$$f(0, \vec{x}, \vec{X}) = g(\vec{x}, \vec{X})$$
$$f(y+1, \vec{x}, \vec{X}) = h(y, f(y, \vec{x}, \vec{X}), \vec{x}, \vec{X})$$

and for all $y, \vec{x}, \vec{X}$, $f(y, \vec{x}, \vec{X}) < 2$.

For $y \geq 1$, let (we drop mention of $\vec{x}, \vec{X}$)

$$z = max(\{0\} \cup \{u < y \ : \ h(u,0) = h(u,1)\})$$
$$n = mod_2(y, \{u \ : \ z < u < y \wedge h(u,0) \neq 0\})$$
$$v = \begin{cases} g & \text{if } z = 0 \\ h(z,0) & \text{otherwise} \end{cases}$$

Then $f(y) = 0$ iff either (i) $v = 0$ and $n = 0$, or (ii) $v = 1$ and $n = 1$. In other words, f can be obtained from g, h and mod_2 by $\mathbf{AC}^0$ reduction. ⊣

Definition 9.66. *The language $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$ is the smallest set that includes $\mathcal{L}_{\mathbf{FAC}^0}$ such that for every $\mathcal{L}_A^2$-term t and open $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$-formula φ the function $F_{\varphi(z),t}$ with defining axiom* (85) *is in $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$, and for every number functions $g(\vec{x}, \vec{X}), h(y, z, \vec{x}, \vec{X}) \in \mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$, there is a number function $f_{g,h}$ in $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$ with defining axioms* *(omitting $\vec{x}, \vec{X}$):*

$$(255)\quad (g < 2 \supset f_{g,h}(0) = g) \wedge (g \geq 2 \supset f_{g,h}(0) = 0) \wedge (h(y, f_{g,h}(y)) < 2 \supset f_{g,h}(y+1) = h(y, f_{g,h}(y))) \wedge (h(y, f_{g,h}(y) \geq 2 \supset f_{g,h}(y+1) = 0)$$

Thus, it follows from Theorem 9.65 that semantically the functions in $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$ represent precisely the functions in $\mathbf{FAC}^0(2)$. We have:

COROLLARY 9.67. (a) *A function is in $\mathbf{FAC}^0(2)$ iff it is represented by a function in $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$.*

1) *A relation is in $\mathbf{AC}^0(2)$ iff it is represented by an open formula in $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$ iff it is represented by a $\Sigma_0^B(\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}})$ formula.*

DEFINITION 9.68. *The theory $\mathbf{VAC}^0(2)\mathbf{V}$ has vocabulary $\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}}$ and is axiomatized by the axioms of $\overline{\mathbf{V}}^0$ together with* (85) *for the functions $F_{\varphi(z),t}$ and* (255) *for the functions $f_{g,h}$.*

THEOREM 9.69. (a) *The theory $\mathbf{VAC}^0(2)\mathbf{V}$ is a universal conservative extension of $\mathbf{V}^0(2)$.*

(b) *For every $\Sigma_0^B(\mathcal{L}_{\mathbf{VAC}^0(2)\mathbf{V}})$ formula φ^+ there is a $\Sigma_1^B(\mathcal{L}_A^2)$ formula φ that is equivalent in $\mathbf{VAC}^0(2)\mathbf{V}$ to φ^+.*

PROOF SKETCH. Part (a) of the following theorem can be proved by formalizing the proof of Theorem 9.65 (see also Theorem 9.43). Part (b) follows from Theorem 9.18 and Corollary 9.46 (see also Corollary 9.47). ⊣

The characterization of $\mathbf{AC}^0(2)$ by $\mathbf{VAC}^0(2)\mathbf{V}$ can be proved as in Section 9C.4 (for the class $\mathbf{TC}^0$ and the theory $\mathbf{VTC}^0\mathbf{V}$).

COROLLARY 9.70. (a) *A function is in $\mathbf{FAC}^0(2)$ iff it is Σ_1^B-definable in $\mathbf{VAC}^0(2)\mathbf{V}$.*

(b) *A relation is in $\mathbf{AC}^0(2)$ iff it is Δ_1^B-definable in $\mathbf{VAC}^0(2)\mathbf{V}$.*

9D.4. The Jordan Curve Theorem and Related Principles. The Jordan Curve Theorem (JCT) asserts that any simple, closed curve divides the plane into exactly two connected components. Here we consider the setting where the curve lies on a grid graph and consists of only horizontal or vertical edges. The notions of grid vertex and edge can be defined using the pairing function. To state the theorem, one way is to represent the curve as a *sequence* of edges that form a simple cycle. To show that there are exactly two connected components we can show that (i) any path (represented by a sequence of edges) that connect two points on different sides of the curve must intersect the curve, and (ii) any two points on the same side of the curve can be connected by a path without intersecting the curve.

Suppose that instead of representing the curve as a sequence of edges we have a *set* of edges such that every grid vertex has degree either 0 or 2. So there may be multiple simple closed curves, and we can only show that there are *at least* two connected components. We will refer to this as the *set* setting of JCT, as opposed to the above *sequence* setting. In this section we will show that the set JCT is a theorem of $\mathbf{V}^0(2)$. The sequence setting is a theorem of $\mathbf{V}^0$, see [65] for details.

We start by defining the notions of grid vertices (or just vertices, or points) and edges, and certain sets of edges which include closed curves, or connect grid points. All of these notions are definable by Σ_0^B-formulas, and their basic properties can be proved in $\mathbf{V}^0$.

We assume a parameter n which bounds the x and y coordinates of points on the curve in question. Thus a point p is a pair (x, y) which is encoded by the pairing function $\langle x, y\rangle$ (see (69) on page 109), where $0 \le x, y \le n$. The x and y coordinates of a point p are denoted by $x(p)$ and $y(p)$ respectively. Thus if $p = \langle i, j\rangle$ then $x(p) = i$ and $y(p) = j$. An (undirected) *edge* is a pair (p_1, p_2) (represented by $\langle p_1, p_2\rangle$) of adjacent points; i.e. either $|x(p_2) - x(p_1)| = 1$ and $y(p_2) = y(p_1)$, or $x(p_2) = x(p_1)$ and $|y(p_2) - y(p_2)| = 1$. For a horizontal edge e, we also write $y(e)$ for the (common) y-coordinate of its endpoints.

Let E be a set of edges (represented by a set of numbers representing those edges). The *E-degree* of a point p is the number of edges in E that are incident to p.

DEFINITION 9.71. *A* curve *is a nonempty set E of edges such that the E-degree of every grid point is either 0 or 2. A set E of edges is said to* connect *two points p_1 and p_2 if the E-degrees of p_1 and p_2 are both 1 and the E-degrees of all other grid points are either 0 or 2. Two sets E_1 and E_2 of edges are said to* intersect *if there is a grid point whose E_i-degree is ≥ 1 for $i = 1, 2$.*

As noted before, a curve in the above sense is actually a collection of one or more disjoint closed curves. Also if E connects p_1 and p_2 then E consists of a path connecting p_1 and p_2 together with zero or more disjoint closed curves.

We also need to define the notion of two points being on different sides of a curve. We are able to consider only points which are "close" to the curve. It suffices to consider the case in which one point is above and one point is below an horizontal edge in E. (Note that the case in which one point is to the left and one point is to the right of a vertical edge in E can be reduced to this case by rotating the $(n+1) \times (n+1)$ array of all grid points by 90 degrees.)

DEFINITION 9.72. *Two points p_1, p_2 are said to be* on different sides *of E if*

(i) $x(p_1) = x(p_2) \wedge |y(p_1) - y(p_2)| = 2$,
(ii) *the E-degree of $p_i = 0$ for $i = 1, 2$, and*
(iii) *the E-degree of p is 2, where p is the point with $x(p) = x(p_1)$ and $y(p) = \frac{1}{2}(y(p_1) + y(p_2))$. (See Figure 5.)*

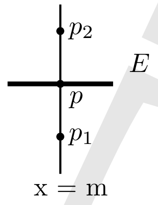

FIGURE 5. p_1, p_2 are on different sides of E.

Now we show that any set of edges that forms at least one simple curve must divide the plane into at least two connected components.

THEOREM 9.73. *The theory $\mathbf{V}^0(2)$ proves the following: Suppose that B is a set of edges forming a curve, p_1 and p_2 are two points on different sides of B, and that R is a set of edges that connects p_1 and p_2. Then B and R intersect.*

PROOF. Since $\widehat{\mathbf{V}^0(2)}$ is conservative over $\mathbf{V}^0(2)$, it suffices to give a $\widehat{\mathbf{V}^0(2)}$ proof of the theorem. By Theorem 9.62 we can use $\Sigma_0^B(\mathit{Parity})$-**COMP** and hence also $\Sigma_0^B(\mathit{Parity})$-**IND** and $\Sigma_0^B(\mathit{Parity})$-**MIN** (Theorem 5.8).

In the following discussion we also refer to the edges in B as "blue" edges, and the edges in R as "red" edges.

We argue in $\mathbf{V}^0(2)$, and prove the theorem by contradiction. Suppose to the contrary that B and R satisfy the hypotheses of the theorem, but do not intersect.

Notation. A horizontal edge is said to be *on column k* (for $k \leq n-1$) if its endpoints have x-coordinates k and $k+1$.

Let $m = x(p_1) = x(p_2)$. W.l.o.g., assume that $2 \leq m \leq n-2$. Also, we may assume that the red path comes to both p_1 and p_2 from the left, i.e., the two red edges that are incident to p_1 and p_2 are both horizontal and on column $m-1$ (see Figure 6). (Note that if the red path does not come to both points from the left, we could fix this by effectively doubling the density of the points by doubling n to $2n$, replacing each edge in B or R by a double edge, and then extending each end of the new path by three (small) edges forming a "C" shape to end at points a distance 1 from the blue curve, approaching from the left.)

We say that edge e_1 *lies below* edge e_2 if e_1 and e_2 are horizontal and in the same column and $y(e_1) < y(e_2)$. For each horizontal red edge r we consider the parity of the number of horizontal blue edges b that lie below r. We say that an edge is "odd" if it is red and there are an odd number of blue edges below it. Recall that $\mathit{PARITY}(X)$ holds iff X contains an odd number of elements:

$$\mathit{PARITY}(X) \leftrightarrow \mathit{Parity}(|X|, X)(|X| \dot{-} 1)$$

Formally we have:

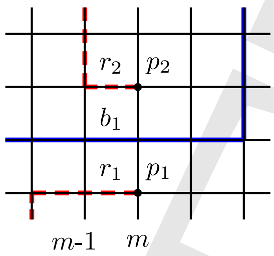

Figure 6. The red (dashed) path must cross the blue (un-dashed) curve.

Notation. For each edge r let Z_r denote the set of all horizontal blue edges that lie below r. An edge r is said to be an *odd edge* if it is red and horizontal and $PARITY(Z_r)$.

For example, it is easy to show in $\mathbf{V}^0(2)$ that exactly one of r_1, r_2 in Figure 6 is an odd edge.

For each $k \le n-1$, define using $\Sigma_0^B(Parity)$-**COMP** the set

$$X_k = \{r : r \text{ is an odd edge in column } k\}$$

Lemma 9.74. *It is provable in $\mathbf{V}^0(2)$ that*

(a) $PARITY(X_{m-1}) \leftrightarrow \neg PARITY(X_m)$.
(b) *For $0 \le k \le n-2$, $k \ne m$, $PARITY(X_k) \leftrightarrow PARITY(X_{k+1})$.*

Using this lemma the proof of the theorem is completed as follows. We may assume that there are no edges in either B or R in columns 0 and $(n-1)$, so $\neg PARITY(X_0) \wedge \neg PARITY(X_{n-1})$. On the other hand, it follows by $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{V}^0(2)}})$-**IND** using Lemma 9.74 (b) that $PARITY(X_0) \leftrightarrow PARITY(X_{m-1})$ and $PARITY(X_m) \leftrightarrow PARITY(X_{n-1})$, which contradicts (a). $\dashv$

It remains to prove Lemma 9.74.

Proof of Lemma 9.74. First we prove (b). For $k \le n-1$ and $0 \le j \le n$, let $e_{k,j}$ be the horizontal edge on column k with y-coordinate j. Fix $k \le n-2$. Define the ordered lists (see Figure 7)

$$L_0 = e_{k,0}, e_{k,1}, \ldots, e_{k,n}; \qquad L_{n+1} = e_{k+1,0}, e_{k+1,1}, \ldots, e_{k+1,n}$$

and for $1 \le j \le n$:

$$L_j = e_{k+1,0}, \ldots, e_{k+1,j-1}, \langle (k+1, j-1), (k+1, j) \rangle, e_{k,j}, \ldots, e_{k,n}$$

A red edge $r \in L_j$ is said to be *odd in L_j* if there are an odd number of blue edges in L_j preceding r. Formally, for each red edge $r \in L_j$ let W be the set of blue edges in L_j that precede r. Then r is odd in L_j just in case $PARITY(W)$ is true. In particular, X_k and X_{k+1} consist of odd edges in L_0 and L_{n+1}, respectively.

For $0 \le j \le n+1$, let

$$Y_j = \{r : r \text{ is an odd edge in } L_j\}$$

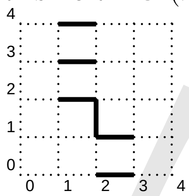

Figure 7. L_2 (for $n = 4, k = 1$).

Thus $Y_0 = X_k$ and $Y_{n+1} = X_{k+1}$.

Claim. If $k \neq m - 1$ then $PARITY(Y_j) \leftrightarrow PARITY(Y_{j+1})$ for $j \leq n$.

This is because the symmetric difference of Y_j and Y_{j+1} has either no red edges, or two red edges with the same parity.

Thus by $\Sigma_0^B(\mathcal{L}_{\widehat{\mathbf{V}^0(2)}})$-**IND** on j we have $PARITY(Y_0) \leftrightarrow PARITY(Y_{n+1})$, and hence $PARITY(X_k) = PARITY(X_{k+1})$.

The proof of (a) is similar. The only change here is that $PARITY(L_j)$ and $PARITY(L_{j+1})$ must differ for exactly one value of j: either $j = y(p_1)$ or $j = y(p_2)$ (because either r_1 is odd in $L_{y(p_1)}$ or r_2 is odd in $L_{y(p_2)}$, but not both). $\dashv$

9D.5. The theories for $\mathbf{AC}^0(m)$ and ACC. Now we present theories associated with $\mathbf{AC}^0(m)$, for $m \geq 3$. They are defined in the same way as the theories $\mathbf{V}^0(2)$, $\widehat{\mathbf{V}^0(2)}$ and $\overline{\mathbf{V}^0(2)}$. Let $\delta_{MOD_m}(x, Y, Z)$ be the $\Sigma_0^B(\mathcal{L}_A^2)$ equivalence (by Lemma 5.74) of the following formula:

$$Z(0,0) \wedge \forall z < x,$$
$$(Y(z) \supset (Z)^{z+1} = ((Z)^z + 1) \mod m)) \wedge (\neg Y(z) \supset (Z)^{z+1} = (Z)^z).$$

Thus $\delta_{MOD_m}(x, Y, Z)$ states that $Z = Mod_m(x, Y)$, the "counting modulo m sequence" for Y (see (253) on page 285). Indeed, the Σ_0^B graph of Mod_m is as follows:

$$Mod_m(x, Y) = Z \leftrightarrow \big(|Z| \leq 1 + \langle x, m\rangle \wedge \delta_{MOD_m}(x, Y, Z)\big)$$

Let

$$MOD_m \equiv \forall x \forall Y \exists Z \delta_{MOD_m}(x, Y, Z)$$

Definition 9.75. *For each $m \geq 3$, the theory $\mathbf{V}^0(m)$ has vocabulary $\mathcal{L}_A^2$ and is axiomatized by $\mathbf{V}^0$ and the axiom MOD_m.*

The next exercise can be proved in the same way as Lemma 9.30.

Exercise 9.76. *For each $m \geq 3$ the function $Mod_m^\star$ is Σ_1^B-definable in $\mathbf{V}^0(m)$, and $\mathbf{V}^0(m)(Row, Mod_m, Mod_m^\star)$ proves*

$$\forall i < b,\ Mod_m^\star(b, X, Y)^{[i]} = Mod_m((X)^i, Y^{[i]})$$

Now we define $\overline{\mathbf{V}^0(m)}$ and $\widehat{\mathbf{V}^0(m)}$. To define $\widehat{\mathbf{V}^0(m)}$ we use the string function $Mod'_m(x, Y)$ defined by

$$Mod'_m(x, Y) = Z \leftrightarrow \big(|Z| \le 1 + \langle x, m\rangle \wedge \delta'_{MOD_m}(x, Y, Z)\big) \tag{256}$$

where $\delta'_{MOD_m}(x, Y, Z)$ is the quantifier-free $\mathcal{L}_{\mathbf{FAC}^0}$-formula that is equivalent to $\delta_{MOD_m}(x, Y, Z)$ over $\overline{\mathbf{V}}^0$ (see Lemma 5.70).

DEFINITION 9.77 ($\widehat{\mathbf{V}^0(m)}$). *For each $m \ge 3$, the theory $\widehat{\mathbf{V}^0(m)}$ has vocabulary $\mathcal{L}_{\widehat{\mathbf{V}^0(m)}} = \mathcal{L}_{\mathbf{FAC}^0} \cup \{Mod'_m\}$ and axioms that of $\overline{\mathbf{V}}^0$ and (256).*

For $\overline{\mathbf{V}^0(m)}$ we start with the function mod'_m which is equal to mod_m but has the following quantifier-free defining axioms (we identify the natural number m with the corresponding numeral $\underline{m}$):

$$mod'_m(0, Y) = 0 \tag{257}$$

(258)
$$(Y(x) \wedge mod'_m(x, Y) + 1 < m) \supset mod'_m(x + 1, Y) = mod'_m(x, Y) + 1$$

$$(Y(x) \wedge mod'_m(x, Y) + 1 = m) \supset mod'_m(x + 1, Y) = 0 \tag{259}$$

$$\neg Y(x) \supset mod'_m(x + 1, Y) = mod'_m(x, X) \tag{260}$$

DEFINITION 9.78. *For each $m \ge 2$, $\mathcal{L}_{\mathbf{FAC}^0(m)}$ is the smallest set that contains $\mathcal{L}_{\mathbf{FAC}^0} \cup \{mod'_m\}$ such that for each quantifier-free formula $\varphi(z, \vec{x}, \vec{X})$ of $\mathcal{L}_{\mathbf{FAC}^0(m)}$ and term $t(\vec{x}, \vec{X})$ of $\mathcal{L}^2_A$, there is a string function $F_{\varphi(z),t}$ with defining axiom (85):*

$$F_{\varphi(z),t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X})$$

The theory $\overline{\mathbf{V}^0(m)}$ has vocabulary $\mathcal{L}_{\mathbf{FAC}^0(m)}$ and is axiomatized by the axioms of $\overline{\mathbf{V}}^0$, (257), (258), (259) and (260) for mod'_m, and (85) for each function $F_{\varphi(z),t}$.

The following Definability Theorems follows from the results in Section 9B:

COROLLARY 9.79. *Here either $\mathcal{L}$ is $\mathcal{L}_{\widehat{\mathbf{V}^0(m)}}$ and $\mathcal{T}$ is $\widehat{\mathbf{V}^0(m)}$, or $\mathcal{L}$ is $\mathcal{L}_{\mathbf{FAC}^0(m)}$ and $\mathcal{T}$ is $\overline{\mathbf{V}^0(m)}$.*

(a) *A function is in $\mathbf{FAC}^0(m)$ iff it is represented by a term in $\mathcal{L}_{\widehat{\mathbf{V}^0(m)}}$ (and for string function) iff it is represented by a symbol in $\mathcal{L}_{\mathbf{FAC}^0(m)}$. A relation is in $\mathbf{AC}^0(m)$ iff it is represented by an open (or a Σ^B_0) formula of $\mathcal{L}$.*
(b) *Every $\Sigma^B_0(\mathcal{L})$ formula is equivalent in $\mathcal{T}$ to a $\Sigma^B_1(\mathcal{L}^2_A)$ formula.*
(c) *$\mathcal{T}$ proves $\Sigma^B_0(\mathcal{L})$-**COMP**.*
(d) *Both $\widehat{\mathbf{V}^0(m)}$ and $\overline{\mathbf{V}^0(m)}$ are universal conservative extensions of $\mathbf{V}^0(m)$.*

(e) *A function is in* $\mathbf{FAC}^0(m)$ *iff it is* Σ_1^B*-definable in* $\mathbf{V}^0(m)$ *iff it is* Σ_1^B*-definable in* $\mathcal{T}$.

(f) *A relation is in* $\mathbf{FAC}^0(m)$ *iff it is* Δ_1^B*-definable in* $\mathbf{V}^0(m)$ *iff it is* Δ_1^B*-definable in* $\mathcal{T}$.

COROLLARY 9.80. *Let* p, q *be two distinct prime numbers.*

(a) $\mathbf{V}^0(p) \not\subseteq \mathbf{V}^0(q)$.

(b) $\mathbf{V}^0 \subsetneq \mathbf{V}^0(p) \subsetneq \mathbf{VTC}^0$.

Theories for $\mathbf{ACC}$ are as follows:

DEFINITION 9.81.

$$\mathbf{VACC} = \bigcup_{m\geq 2} \mathbf{V}^0(m)$$

$$\mathcal{L}_{\mathbf{FACC}} = \bigcup_{m\geq 2} \mathcal{L}_{\mathbf{FAC}^0(2)}, \qquad \overline{\mathbf{VACC}} = \bigcup_{m\geq 2} \overline{\mathbf{V}^0(m)}$$

$$\mathcal{L}_{\widehat{\mathbf{VACC}}} = \bigcup_{m\geq 2} \mathcal{L}_{\widehat{\mathbf{V}^0(m)}}, \qquad \widehat{\mathbf{VACC}} = \bigcup_{m\geq 2} \widehat{\mathbf{V}^0(m)}$$

The next Definability Theorems for the theories associated with $\mathbf{ACC}$ follow from Corollary 9.79.

COROLLARY 9.82. *Here either* $\mathcal{L}$ *is* $\mathcal{L}_{\mathbf{FACC}}$ *and* $\mathcal{T}$ *is* $\widehat{\mathbf{VACC}}$, *or* $\mathcal{L}$ *is* $\mathcal{L}_{\mathbf{FACC}}$ *and* $\mathcal{T}$ *is* $\overline{\mathbf{VACC}}$.

(a) *A function is in* $\mathbf{FACC}$ *iff it is represented by a term in* $\mathcal{L}_{\widehat{\mathbf{VACC}}}$ *(and for string function) iff it is represented by a symbol in* $\mathcal{L}_{\mathbf{FACC}}$. *A relation is in* $\mathbf{ACC}$ *iff it is represented by an open (or a* Σ_0^B*) formula of* $\mathcal{L}$.

(b) *Every* $\Sigma_0^B(\mathcal{L})$ *formula is equivalent in* $\mathcal{T}$ *to a* $\Sigma_1^B(\mathcal{L}_A^2)$ *formula.*

(c) $\mathcal{T}$ *proves* $\Sigma_0^B(\mathcal{L})$-$\mathbf{COMP}$.

(d) *Both* $\widehat{\mathbf{VACC}}$ *and* $\overline{\mathbf{VACC}}$ *are universal conservative extensions of* $\mathbf{VACC}$.

(e) *A function is in* $\mathbf{FACC}$ *iff it is* Σ_1^B*-definable in* $\mathbf{V}^0(m)$ *iff it is* Σ_1^B*-definable in* $\mathcal{T}$.

(f) *A relation is in* $\mathbf{FACC}$ *iff it is* Δ_1^B*-definable in* $\mathbf{V}^0(m)$ *iff it is* Δ_1^B*-definable in* $\mathcal{T}$.

EXERCISE 9.83. (a) *If* $\mathbf{VACC}$ *is finitely axiomatizable, then* $\mathbf{ACC} = \mathbf{AC}^0(m)$, *for some* m.

(b) $\mathbf{VACC} \subseteq \mathbf{VTC}^0$, *and* $\mathbf{V}^0(p) \subsetneq \mathbf{VACC}$ *for any prime* p.

9D.6. The theory $\mathbf{VAC}^0(6)\mathbf{V}$. Here we develop $\mathbf{VAC}^0(6)\mathbf{V}$ in the same way as $\mathbf{VAC}^0(2)\mathbf{V}$, using Theorem 9.84 below. Recall the bounded number recursion (BNR) from Section 9C.3. It can be shown that:

THEOREM 9.84. *A function is in* $\mathbf{FAC}^0(6)$ *iff it is obtained from* $\mathbf{FAC}^0$ *by finitely many applications of* $\mathbf{AC}^0$ *reduction and 3-BNR iff it* *is obtained from* $\mathbf{FAC}^0$ *by finitely many applications of* $\mathbf{AC}^0$ *reduction and 4-BNR.*

Thus, the functions in $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$ defined below represent precisely the functions in $\mathbf{FAC}^0(6)$.

DEFINITION 9.85. *The language* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$ *is the smallest set that includes* $\mathcal{L}_{\mathbf{FAC}^0}$ *such that for every* $\mathcal{L}_A^2$*-term* t *and open* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$*-formula* φ *the function* $F_{\varphi(z),t}$ *with defining axiom* (85) *is in* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$*, and for every number functions* $g(\vec{x},\vec{X}), h(y,z,\vec{x},\vec{X}) \in \mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$*, there is a number function* $f_{g,h}$ *in* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$ *with defining axioms (omitting* $\vec{x},\vec{X}$*):*

$$
\begin{aligned}
(261)\quad & (g < 3 \supset f_{g,h}(0) = g) \wedge (g \geq 3 \supset f_{g,h}(0) = 0) \wedge \\
& \quad (h(y, f_{g,h}(y)) < 3 \supset f_{g,h}(y+1) = h(y, f_{g,h}(y))) \wedge \\
& \qquad\qquad (h(y, f_{g,h}(y) \geq 3 \supset f_{g,h}(y+1) = 0)
\end{aligned}
$$

The next corollary follows from Theorem 9.84. It states that semantically the functions in $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$ represent precisely the functions in $\mathbf{FAC}^0(6)$.

COROLLARY 9.86. (a) *A function is in* $\mathbf{FAC}^0(6)$ *iff it is represented by an* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$*-term.*

1) *A relation is in* $\mathbf{AC}^0(6)$ *iff it is represented by an open formula in* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$ *iff it is represented by a* $\Sigma_0^B(\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}})$ *formula.*

DEFINITION 9.87. *The theory* $\mathbf{VAC}^0(6)\mathbf{V}$ *has vocabulary* $\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}}$ *and is axiomatized by the axioms of* $\overline{\mathbf{V}}^0$ *together with* (85) *for the functions* $F_{\varphi(z),t}$ *and* (261) *for the functions* $f_{g,h}$.

THEOREM 9.88. (a) *The theory* $\mathbf{VAC}^0(6)\mathbf{V}$ *is a universal conservative extension of* $\mathbf{V}^0(6)$.

(b) *Every* $\Sigma_0^B(\mathcal{L}_{\mathbf{VAC}^0(6)\mathbf{V}})$ *formula* φ^+ *is equivalent in* $\mathbf{VAC}^0(6)\mathbf{V}$ *to a* Σ_1^B *formula* φ.

PROOF SKETCH. Part (a) is proved by formalizing both directions of the proof of Theorem 9.84 in appropriate theories ($\mathbf{V}^0(6)$ or $\mathbf{VAC}^0(6)\mathbf{V}$). Part (b) is proved in the same way as Corollary 9.47. $\dashv$

The characterization of $\mathbf{AC}^0(6)$ by $\mathbf{VAC}^0(6)\mathbf{V}$ can be proved as in Section 9C.4 (for the class $\mathbf{TC}^0$ and the theory $\mathbf{VTC}^0\mathbf{V}$).

COROLLARY 9.89. (a) *A function is in* $\mathbf{FAC}^0(6)$ *iff it is* Σ_1^B*-definable in* $\mathbf{VAC}^0(6)\mathbf{V}$.

(b) *A relation is in* $\mathbf{AC}^0(6)$ *iff it is* Δ_1^B*-definable in* $\mathbf{VAC}^0(6)\mathbf{V}$.

9E. Theories for $\mathbf{NC}^1$ and the NC Hierarchy

The classes $\mathbf{NC}^k$ and $\mathbf{AC}^k$ form an interesting hierarchy inside $\mathbf{P}$. We have already encountered a member of this hierarchy, namely $\mathbf{AC}^0$. In this section we develop theories for other classes, and we will focus on the theories for $\mathbf{NC}^1$ which is the next level above $\mathbf{AC}^0$.

First, in Section 9E.1 we define the classes. Then in Section 9E.2 we develop the theories $\mathbf{VNC}^1$, $\widehat{\mathbf{VNC}}^1$ and $\overline{\mathbf{VNC}}^1$ that characterize $\mathbf{NC}^1$ as discussed in Section 9B. Then in Section 9E.3 we show that $\mathbf{VNC}^1$ extends $\mathbf{VTC}^0$. Based on Barrington's Theorem, in Section 9E.4 we use the bounded number recursion (BNR) operation (see Section 9C.3) to develop $\mathbf{VNC}^1\mathbf{V}$, a universal conservative extension of $\mathbf{VNC}^1$. Finally, the theories for other classes in the $\mathbf{NC}$ hierarchy are defined in Section 9E.5.

In Section 10C.1 we will prove the Propositional Translation Theorem for $\mathbf{VNC}^1$.

9E.1. $\mathbf{NC}^1$ and the NC Hierarchy. Recall the definition of $\mathbf{AC}^0$ using uniform families of circuits in Section 4A. In general, for $k \geq 0$, $\mathbf{FO}$-uniform $\mathbf{AC}^k$ (or just $\mathbf{AC}^k$) is the class of problems decidable using an $\mathbf{FO}$-uniform family $\langle C_n \rangle$ of polynomial size Boolean circuits, where each circuit C_n has n input bits and $(\log n)^k$ depth, and the gates in C_n have unbounded fan-in. The class $\mathbf{FO}$-uniform $\mathbf{NC}^k$ (or simply $\mathbf{NC}^k$) is defined in the same way, except for the gates have bounded fan-in.

It is easy to see that for $k \geq 0$:

$$\mathbf{NC}^k \subseteq \mathbf{AC}^k \subseteq \mathbf{NC}^{k+1}$$

Furthermore, $\mathbf{NC}^0 \subsetneq \mathbf{AC}^0$ because the $\mathbf{NC}^0$ circuits cannot compute the conjunction of all inputs, and $\mathbf{AC}^0 \subsetneq \mathbf{NC}^1$ because the function $\mathit{parity}(X)$ is in $\mathbf{NC}^1$ but not in $\mathbf{AC}^0$. In summarize, we have:

$$\mathbf{NC}^0 \subsetneq \mathbf{AC}^0 \subsetneq \mathbf{NC}^1 \subseteq \mathbf{AC}^1 \subseteq \mathbf{NC}^2 \subseteq \dots$$

These classes form the $\mathbf{NC}$ hierarchy:

$$\mathbf{NC} = \bigcup_{k \geq 0} \mathbf{NC}^k = \bigcup_{k \geq 0} \mathbf{AC}^k$$

We are only interested in classes that contain $\mathbf{AC}^0$. Various notions of uniformity can be used to define $\mathbf{AC}^k$ and $\mathbf{NC}^{k+1}$ for $k \geq 1$. (In this book we use $\mathbf{AC}^0$ uniformity as mentioned in Chapter 4.) Also, $\mathbf{NC}$ can be alternatively defined using alternating Turing machines (ATMs). In particular, let $\mathbf{ASPACE\text{-}ALT}(s, r)$ denote the class of languages accepted by alternating Turing machines in space $\mathcal{O}(s)$ with $\mathcal{O}(r)$ alternations. Then for $k \geq 1$,

$$\mathbf{AC}^k = \mathbf{ASPACE\text{-}ALT}(\log(n), (\log(n))^k)$$

(Recall that $\mathbf{AC}^0$ consists of languages computable by alternating Turing machines working in time $\mathcal{O}(\log(n))$ and constant alternations.)

Similarly, let $\mathbf{ASPACE\text{-}TIME}(s,t)$ denote the class of languages accepted by alternating Turing machines in simultaneous space $\mathcal{O}(s)$ and time $\mathcal{O}(t)$. Then for $k \geq 1$,

$$\mathbf{NC}^k = \mathbf{ASPACE\text{-}TIME}(\log(n), (\log(n))^k)$$

For this reason, $\mathbf{NC}^1$ is also called **ALogTime** in literature.

The Boolean Sentence Value Problem (BSVP) is to decide the truth value of a Boolean sentence given its infix representation. In Section 10C.2 we will show that that BSVP is $\mathbf{AC}^0$-many-one complete for $\mathbf{NC}^1$. In fact, the problem remains $\mathbf{AC}^0$-many-one complete for $\mathbf{NC}^1$ even for monotone formulas that have a "balanced" structure when viewed as a binary tree. We use this fact here to define the function *Fval* (*Fval* stands for "formula value") whose $\mathbf{AC}^0$ closure is $\mathbf{NC}^1$.

Consider the following encoding of a balanced monotone Boolean sentence using the heap data structure. We view the sentence as a balanced binary tree with $(2a-1)$ nodes: a leaves numbered

$$a, (a+1), \ldots, (2a-1)$$

and $(a-1)$ inner nodes numbered

$$1, 2, \ldots, (a-1)$$

Each inner node (or gate) is either an $\wedge$- or an $\vee$-gate, and each leaf is labeled with a Boolean value. The two children of an inner node x are $2x$ and $(2x+1)$ (as in the heap data structure). Therefore the sentence can be encoded by (a, G, I), where $G(x)$ specifies the label of node x: for $x < a$,

if $G(x)$ holds then node x is an $\wedge$-gate, otherwise x is an $\vee$-gate,

and I specifies the values at the leaves: for $x < a$,

$I(x)$ is the value labeling leaf $(a+x)$.

We will also refer to the binary tree (b, G) as a tree-like circuit and I its inputs.

The function $\mathit{Fval}(a, G, I)$ describes a polytime procedure that computes the value of the sentence encoded by (a, G, I). This procedure evaluates the values at the nodes of the tree (a, G, I) inductively, starting with the leaves. In the formula $\delta_{MFV}(a, G, I, Y)$ below $Y(x)$ is the value of gate x, for $x < 2a$. (MFV stands for "monotone formula value".)

$$\begin{aligned}(262)\quad \delta_{MFV}(a, G, I, Y) \equiv\ & \forall x < a\big(Y(x+a) \leftrightarrow I(x)\ \wedge \\ & 0 < x \supset (Y(x)\ \leftrightarrow\ ((G(x) \wedge Y(2x) \wedge Y(2x+1)) \vee \\ & (\neg G(x) \wedge (Y(2x) \vee Y(2x+1))))\big)\end{aligned}$$

Figure 8 depicts a computation of (the bits of) Y for $a = 6$. Here the gates $G(1), \ldots, G(5)$ are not shown.

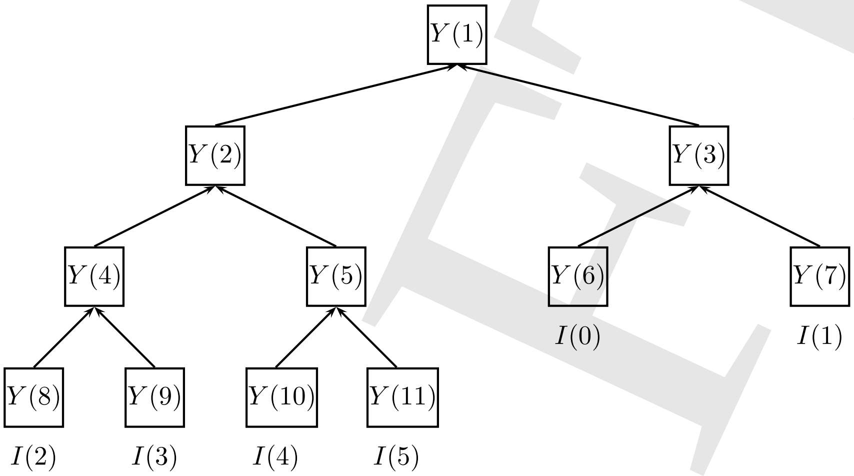

FIGURE 8. Computing Y which satisfies $\delta_{MFV}(a, G, I, Y)$ for $a = 6$

DEFINITION 9.90.

$$R_{MFV}(a, G, I) \leftrightarrow \exists Y \leq 2a(\delta_{MFV}(a, G, I, Y) \wedge Y(1))$$

The following proposition shows that R_{MFV} is $\mathbf{AC}^0$-many-one complete for $\mathbf{NC}^1$. For a proof see [14] and [7, Lemma 6.2].

PROPOSITION 9.91. *For every relation* $R(\vec{x}, \vec{X})$ *in* $\mathbf{NC}^1$ *there are* $\mathbf{AC}^0$ *functions* a_0, G_0, I_0 *such that*

$$R(\vec{x}, \vec{X}) \leftrightarrow R_{MFV}(a_0(\vec{x}, \vec{X}), G_0(\vec{x}, \vec{X}), I_0(\vec{x}, \vec{X}))$$

9E.2. The theories $\mathbf{VNC}^1$, $\widehat{\mathbf{VNC}}^1$ and $\overline{\mathbf{VNC}}^1$. We define the theories $\mathbf{VNC}^1$, $\widehat{\mathbf{VNC}}^1$ and $\overline{\mathbf{VNC}}^1$ as in Section 9B using the formula δ_{MFV} above. In Section 9E.4 we will define $\mathbf{VNC}^1\mathbf{V}$, another universal conservative extension of $\mathbf{VNC}^1$ using number recursion.

DEFINITION 9.92 ($\mathbf{VNC}^1$). *Let*

$$MFV \equiv \forall a \forall G \forall I \exists Y \leq 2a\, \delta_{MFV}(a, G, I, Y) \tag{263}$$

The theory $\mathbf{VNC}^1$ *has vocabulary* $\mathcal{L}_A^2$ *and is axiomatized by* MFV *and the axioms of* $\mathbf{V}^0$.

DEFINITION 9.93. *The function* $Fval(a, G, I)$ *is defined as follows:*

$$Fval(a, G, I) = Y \leftrightarrow |Y| \leq 2a \wedge \delta_{MFV}(a, G, I, Y) \tag{264}$$

PROPOSITION 9.94. *The function* $Fval$ *is many-one* $\mathbf{AC}^0$ *complete for* $\mathbf{NC}^1$.

To define $\widehat{\mathbf{VNC}}^1$ we use the following quantifier-free defining axiom for *Fval*. It is easy to see that this defining axiom is equivalent to (264).

$$\text{(265)}\quad Fval(a,G,I)=Y\leftrightarrow$$
$$(|Y|\le 2a)\wedge\big(x<a\supset(Y(x+a)\leftrightarrow I(x))\big)\ \wedge$$
$$(0<x\wedge x<a)\supset\big(Y(x)\ \leftrightarrow\ \big((G(x)\wedge Y(2x)\wedge Y(2x+1))\vee$$
$$(\neg G(x)\wedge(Y(2x)\vee Y(2x+1)))\big)\big)$$

DEFINITION 9.95. $\widehat{\mathbf{VNC}}^1$ *is the universal theory over the vocabulary* $\mathcal{L}_{\widehat{\mathbf{VNC}}^1}=\mathcal{L}_{\mathbf{FAC}^0}\cup\{Fval\}$ *with axioms those of* $\overline{\mathbf{V}}^0$ *and the defining axiom* (265) *for Fval.*

DEFINITION 9.96. $\mathcal{L}_{\mathbf{FNC}^1}$ *is the smallest set that contains* $\mathcal{L}_{\widehat{\mathbf{VNC}}^1}$ *such that for every* $\mathcal{L}^2_A$*-term* t *and every quantifier-free* $\mathcal{L}_{\mathbf{FNC}^1}$*-formula* φ *there is a function* $F_{\varphi(z),t}$ *in* $\mathcal{L}_{\mathbf{FNC}^1}$ *with defining axiom* (85).

The characterization of $\mathbf{NC}^1$ *by* $\mathbf{VNC}^1$ *and* $\widehat{\mathbf{VNC}}^1$ *can be proved as in Section 8A using Proposition 9.91 above, or follows from the results in Section 9B. Either way we need to show that* $Fval^\star$ *is* Σ^B_1*-definable in* $\mathbf{VNC}^1$*. Proving this fact requires a few technical results and is postponed until Exercise 9.102.*

$\overline{\mathbf{VNC}}^1$ *is the universal theory over* $\mathcal{L}_{\mathbf{FNC}^1}$ *that is axiomatized by the axioms of* $\widehat{\mathbf{VNC}}^1$ *and* (85) *for each function* $F_{\varphi(z),t}$ *of* $\mathcal{L}_{\mathbf{FNC}^1}$*.*

Given the fact that $Fval^\star$ is Σ^B_1-definable in $\mathbf{VNC}^1$ (Exercise 9.102) the following Definability Theorems follow as corollaries from the results in Section 9B.

COROLLARY 9.97. *Here either* $\mathcal{L}$ *is* $\mathcal{L}_{\widehat{\mathbf{VNC}}^1}$ *and* $\mathcal{T}$ *is* $\widehat{\mathbf{VNC}}^1$*, or* $\mathcal{L}$ *is* $\mathcal{L}_{\mathbf{FNC}^1}$ *and* $\mathcal{T}$ *is* $\overline{\mathbf{VNC}}^1$*.*

(a) *A function is in* $\mathbf{FNC}^1$ *iff it is represented by a term in* $\mathcal{L}_{\widehat{\mathbf{VNC}}^1}$ *(and for a string function) iff it is represented by a function in* $\mathcal{L}_{\mathbf{FNC}^1}$*. A relation is in* $\mathbf{NC}^1$ *iff it is represented by an open (or* Σ^B_0*) formula of* $\mathcal{L}$*.*
(b) *Every* $\Sigma^B_0(\mathcal{L})$ *formula is equivalent in* $\mathcal{T}$ *to a* $\Sigma^B_1(\mathcal{L}^2_A)$ *formula.*
(c) $\mathcal{T}$ *proves* $\Sigma^B_0(\mathcal{L})$*-*$\mathbf{COMP}$*.*
(d) $\overline{\mathbf{VNC}}^1$ *is a universal conservative of* $\widehat{\mathbf{VNC}}^1$*, which is in turn a universal conservative extension of* $\mathbf{VNC}^1$*.*
(e) *A function is in* $\mathbf{FNC}^1$ *iff it is* Σ^B_1*-definable in* $\mathbf{VNC}^1$ *iff it is* Σ^B_1*-definable in* $\mathcal{T}$*.*
(f) *A relation is in* $\mathbf{NC}^1$ *iff it is* Δ^B_1*-definable in* $\mathbf{VNC}^1$ *iff it is* Δ^B_1*-definable in* $\mathcal{T}$*.*

The original definition of $\mathbf{VNC}^1$ [33] uses the axiom scheme Σ^B_0-*TreeRec* in stead of the axiom *MFV*.

DEFINITION 9.98 (Σ_0^B-*TreeRec*). *Σ_0^B-TreeRec is the set of axioms of the form*

$$\text{(266)}\quad \exists Y \forall x < a$$
$$(Y(x+a) \leftrightarrow \psi(x)) \wedge \big((0 < x \supset (Y(x) \leftrightarrow \varphi(x)[Y(2x), Y(2x+1)])\big)$$

where $\psi(x)$ is a Σ_0^B formula, $\varphi(x)[p, q]$ is a Σ_0^B formula which contains two Boolean variables p and q, and Y does not occur in ψ and φ.

We will show that our definition of $\mathbf{VNC}^1$ here is equivalent to the original definition. Since *MFV* is an instance of the Σ_0^B-*TreeRec* axiom scheme, we need only to show that Σ_0^B-*TreeRec* is provable in $\mathbf{VNC}^1$. This is proved in Theorem 9.99 below. In Section 9E.3 below we will show that $\mathbf{VNC}^1$ proves several generalizations of Σ_0^B-*TreeRec* (Theorems 9.100 and 9.101).

THEOREM 9.99. *The Σ_0^B-TreeRec axiom scheme is provable in $\mathbf{VNC}^1$.*

PROOF. Given a, ψ and φ, the idea is to construct a (large) treelike circuit (b, G) and inputs I so that from $Fval(b, G, I)$ we can extract Y (using Σ_0^B-$\mathbf{COMP}$) that satisfies (266).

Notice the "gates" $\varphi(x)[p, q]$ in (266) can be any of the sixteen Boolean functions in two variables p, q. We will (uniformly) construct binary treelike $\wedge$-$\vee$ circuits of constant depth that compute $\varphi(x)[p, q]$.

Let

$$\beta_1, \ldots, \beta_8, \beta_9 \equiv \neg\beta_1, \ldots, \beta_{16} \equiv \neg\beta_8$$

be the sixteen Boolean functions in two variables p, q. Each β_i can be computed by a binary treelike and-or circuit of depth 2 with inputs among 0, 1, p, q, $\neg p$, $\neg q$. For $1 \leq i \leq 16$, let X_i be defined by

$$X_i(x) \leftrightarrow (x < a \wedge \varphi(x)[p, q] \leftrightarrow \beta_i(p, q))$$

Then,

$$\varphi(x)[p, q] \leftrightarrow \bigvee_{i=1}^{16} (X_i(x) \wedge \beta_i(p, q))$$

Consequently, $\varphi(x)[p, q]$ can be computed by a binary and-or tree T_x of depth 7 whose inputs are $0, 1, p, \neg p, q, \neg q, X_i(x)$. Similarly, $\neg\varphi(x)[p, q]$ is computed by a binary and-or tree T'_x having the same depth and set of inputs. Our large tree G has one copy of T_1, and in general for each copy of T_x or T'_x, there are multiple copies of $T_{2x}, T_{2x+1}, T'_{2x}, T'_{2x+1}$ that supply the inputs $Y(2x), Y(2x+1), \neg Y(2x), \neg Y(2x+1)$, and other trivial treelike circuits that provide inputs 0, 1, $X_i(x)$ $(1 \leq i \leq 16)$.

Finally, I is defined as follows: $I(x) \leftrightarrow (x < a \wedge \psi(x))$. $\dashv$

9E.3. $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$. It is known that $\mathbf{TC}^0 \subseteq \mathbf{NC}^1$ (although it is unknown whether the inclusion is proper). Here we will show, informally, that $\mathbf{VNC}^1$ proves this inclusion. In particular, we will show that $\mathbf{VNC}^1$ extends $\mathbf{VTC}^0$. Note that for this it suffices to show that $\mathbf{VNC}^1$ proves the axiom *NUMONES*. Our proof is by formalizing in $\mathbf{VNC}^1$ the construction of $\mathbf{NC}^1$ circuits that compute *numones* and prove in $\mathbf{VNC}^1$ the correctness of this construction. Here we formalize the construction by Buss [14].

The next two theorems show that $\mathbf{VNC}^1$ proves some generalizations of Σ_0^B-*TreeRec*. They are useful in formalizing the construction of the counting circuits. They are also useful in proving that the function $Fval^\star$ is Σ_1^B-definable in $\mathbf{VNC}^1$ (see Exercise 9.102), a result that we need for Corollary 9.97 stated earlier.

First, Theorem 9.100 asserts, informally, that we can evaluate in $\mathbf{VNC}^1$ formulas whose underlying trees have an arbitrary constant branching factor (as opposed to binary trees).

Theorem 9.100. *Suppose that $2 \le k \in \mathbb{N}$, $\psi(x)$ is a Σ_0^B formula, and $\varphi(x)[p_0, \dots, p_{k-1}]$ is a Σ_0^B formula that contains also Boolean variables p_i. Then $\mathbf{VNC}^1$ proves*

$$(267)\quad \exists Y,\ \forall x < ka,\ a \le x \supset Y(x) \leftrightarrow \psi(x) \wedge \\ \forall x < a,\ Y(x) \leftrightarrow \varphi(x)[Y(kx), \dots, Y(kx+k-1)]$$

Proof. We prove for the case $k = 4$; similar arguments work for other cases.

Using Theorem 9.99 we will define a', ψ', φ' so that from Y' that satisfies the Σ_0^B-*TreeRec* axiom (266) for a', ψ' and φ' we can obtain Y that satisfies (267) above.

Intuitively, consider Y in (267) as a forest of three trees whose nodes are labeled with $Y(x)$, $x < |Y|$. Then Y has branching factor of 4 (since $k = 4$), and the three trees are rooted at $Y(1)$, $Y(2)$ and $Y(3)$. (See Figure 9.) Note also that each layer in Y corresponds to two layers in the binary tree Y'.

We will define an injective map f so that $Y(x) \leftrightarrow Y'(f(x))$. Since the trees rooted at $Y(1)$, $Y(2)$ and $Y(3)$ are disjoint, f is defined so that these trees are the images of disjoint subtrees in the tree Y'. For example, we can choose the subtrees rooted at $Y'(4)$, $Y'(5)$ and $Y'(6)$. Thus,

$$f(1) = 4, f(2) = 5, f(3) = 6$$

In general, consider the function f defined by:

$$f(4^m + y) = 4^{m+1} + y \qquad \text{for } \ 0 \le y < 3 \cdot 4^m$$

(By the results in Chapter 3, f is provably total in $\mathbf{I\Delta}_0$, and hence also in $\mathbf{V}^0$.)

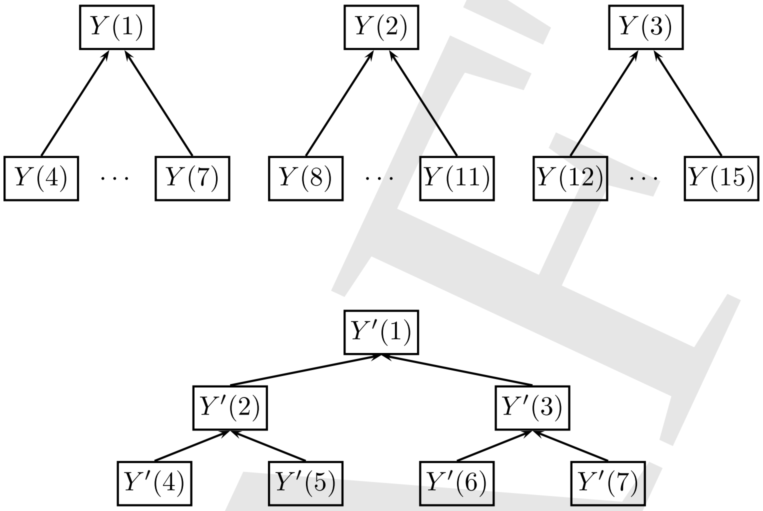

Figure 9. The forest Y in Theorem 9.100 when $k = 4$. Trees rooted at $Y(1)$, $Y(2)$ and $Y(3)$ are simulated by the sub-trees $Y'(4)$, $Y'(5)$ and $Y'(6)$, respectively.

Now we need ψ' such that for $a \le x < ka$

$$\psi'(f(x)) \leftrightarrow \psi(x)$$

So define ψ' as follows: for $y < 3 \cdot 4^m$ and $a \le 4^m + y < ka$,

$$\psi'(4^{m+1} + y) \leftrightarrow \psi(4^m + y)$$

To obtain φ', write $\varphi(x)[p_0, p_1, p_2, p_3]$ in the form

$$\varphi_1(x)[\varphi_2(x)[p_0, p_1], \varphi_3(x)[p_2, p_3]]$$

where φ_i is Σ_0^B with at most 2 Boolean variables, for $1 \le i \le 3$. Define φ' as follows:

$$\begin{aligned}
\varphi'(4^{m+1} + y)[p, q] &\leftrightarrow \varphi_1(4^m + y)[p, q] && \text{for } y < 3 \cdot 4^m \\
\varphi'(2 \cdot 4^{m+1} + 2y)[p, q] &\leftrightarrow \varphi_2(4^m + y)[p, q] && \text{for } y < 3 \cdot 4^m/2 \\
\varphi'(2 \cdot 4^{m+1} + 2y + 1)[p, q] &\leftrightarrow \varphi_3(4^m + y)[p, q] && \text{for } y < 3 \cdot 4^m/2
\end{aligned}$$

Finally, let $a' = f(a)$. Let Y' satisfies (266) for a', ψ' and φ', and let Y be such that

$$Y(x) \leftrightarrow Y'(f(x))$$

It is straightforward to verify that Y satisfies (267). $\dashv$

The next theorem shows that in $\mathbf{VNC}^1$ we can evaluate multiple inter-connected Boolean circuits each has logarithmic depth and constant fan-in.

THEOREM 9.101. *Suppose that* $1 \leq m, \ell \in \mathbb{N}$, *and for* $1 \leq i \leq m$, $\psi_i(x,y)$ *and* $\varphi_i(x,y)[p_1, q_1, \ldots, p_{m\ell}, q_{m\ell}]$ *are* Σ_0^B *formulas where* $\vec{p}, \vec{q}$ *are the Boolean variables. Then* $\mathbf{VNC}^1$ *proves the existence of* $Z_1, \ldots, Z_m$ *such that*

$$\forall z < c \forall x < a \bigwedge_{i=1}^{m} \Big((Z_i^{[z]}(x+a) \leftrightarrow \psi_i(z,x)) \wedge 0 < x \supset \big(Z_i^{[z]}(x) \leftrightarrow$$

$$\varphi_i(z,x)[Z_1^{[z]}(2x), Z_1^{[z]}(2x+1), \ldots, Z_m^{[z+\ell-1]}(2x), Z_m^{[z+\ell-1]}(2x+1)] \big) \Big)$$

PROOF. Using Theorem 9.100 above, the idea is to construct a constant k, a number a' and Σ_0^B formulas $\psi'(c,x)$ and $\varphi'(c,x)[p_0, \ldots, p_{k-1}]$ so that from the set Y that satisfies (267) (for k, a', ψ' and φ') we can obtain $Z_1, \ldots, Z_m$.

Consider for example $m = 2, \ell = 2$. W.l.o.g., assume that $c \geq 1$. The (overlapping) subtrees

$$Z_1^{[0]}, Z_2^{[0]}, \ldots, Z_1^{[c-1]}, Z_2^{[c-1]} \tag{268}$$

have branching factor 8 (i.e., $2m\ell$). So let $k = 8$ (i.e., $k = 2m\ell$). We will construct Y (with branching factor 8) so that the disjoint subtrees rooted at

$$Y(c), \ldots, Y(3c-1) \tag{269}$$

are exactly the subtrees listed in (268).

We will define an 1-1, into map

$$s : \{1,2\} \times \mathbb{N}^2 \to \mathbb{N}$$

so that

$$Z_i^{[z]}(x) \leftrightarrow Y(s(i,z,x))$$

The map s must be defined in such a way that the nodes of the trees listed in (268) match with those whose roots are listed in (269). For example, for the root level we need

$$s(1,0,1) = c,\ s(2,0,1) = c+1,\ s(1,1,1) = c+2,\ s(2,1,1) = c+3,\ \ldots$$

For other levels we need: If $s(i,z,x) = y$, then

$$s(1,z,2x) = 8y,\ s(1,z,2x+1) = 8y+1,\ \ldots,\ s(2,z+1,2x+1) = 8y+7$$

To define s we define partial, onto maps $f, g : \mathbb{N} \to \mathbb{N}$ and $h : \mathbb{N} \to \{1,2\}$ so that

$$s(h(y), g(y), f(y)) = y$$

In other words,

$$Y(y) \leftrightarrow Z_{h(y)}^{[g(y)]}(f(y))$$

For example, for $0 \leq z < 2c$:

$$f(c+z) = 1, \qquad g(c+z) = \lfloor z/2 \rfloor, \qquad h(c+z) = 1 + (z \mod 2)$$

In general, we need to define f, g, h only for values of x of the form $8^r c + z$ for $0 \le z < 2 \cdot 8^r c$. The definitions of f, g, h at $8^r c + z$ are straightforward using the base 8 notation for z, where $0 \le z < 2 \cdot 8^r c$.

Once f, g, h are defined, the formula ψ' and φ' are defined by

$$\psi'(c, x) \leftrightarrow \psi_{h(x)}(g(x), f(x)) \qquad \text{and} \qquad \varphi'(c, x)[\dots] \leftrightarrow \varphi_{h(x)}(g(x), f(x))[\dots]$$

(where $\dots$ is the list of $2m\ell$ Boolean variables). $\dashv$

Exercise 9.102. *Using Theorem 9.101, show that the function $Fval^\star$ is Σ^1_1-definable in $\mathbf{VNC}^1$.*

For the next theorem we use $Sum(a, X)$ for the sum of a rows of X:

$$Sum(a, X) = \begin{cases} \varnothing & \text{if } a = 0 \\ X^{[0]} + X^{[1]} + \dots + X^{[a \dot{-} 1]} & \text{if } a \ge 1 \end{cases}$$

(We introduced the function $Sum(m, n, X)$ in (246) and (247) on page 281. The two functions $Sum(a, X)$ and $Sum(m, n, X)$ have the same name but different arity, so the exact meaning is clear from context.)

Theorem 9.103. *The function $Sum(a, X)$ with the following defining axiom is Σ^B_1-definable in $\overline{\mathbf{VNC}}^1$:*

(270) $Sum(a, X) = Y \leftrightarrow$

$$|Y| \le \langle a, |X| \rangle \wedge Y^{[0]} = \varnothing \wedge \forall x < a (Y^{[x+1]} = Y^{[x]} + X^{[x]})$$

The fact that $\mathbf{VNC}^1 \subseteq \mathbf{VNC}^1$ follows easily:

Corollary 9.104. $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$.

Proof of Theorem 9.103. Informally we need to construct a circuit that adds all rows

$$X^{[0]}, X^{[1]}, \dots, X^{[a-1]}$$

The idea is to use the divide-and-conquer technique. We will construct a balanced binary tree Z that has $(2a - 1)$ nodes (see Figure 10 for an example):

- a leaves $Z^{[a]}, Z^{[a+1]}, \dots, Z^{[2a-1]}$ such that
$$Z^{[a+x]} = X^{[x]} \qquad \text{for } 0 \le x < a$$
- $(a - 1)$ inner nodes $Z^{[1]}, Z^{[2]}, \dots, Z^{[a-1]}$; the two children of node $Z^{[x]}$ are $Z^{[2x]}$ and $Z^{[2x+1]}$, so that
$$Z^{[x]} = Z^{[2x]} + Z^{[2x+1]} \qquad \text{for } 1 \le x < a$$

Lemma 9.105. *Let $DaCAdd(a, X, Z)$ be the formula*

(271) $$\forall x < a,\ Z^{[a+x]} = X^{[x]} \wedge x > 0 \supset Z^{[x]} = Z^{[2x]} + Z^{[2x+1]}$$

($DaCAdd$ stands for "divide-and-conquer addition".) Then

$$\overline{\mathbf{VNC}}^1 \vdash \forall a \forall X \exists Z\, DaCAdd(a, X, Z)$$

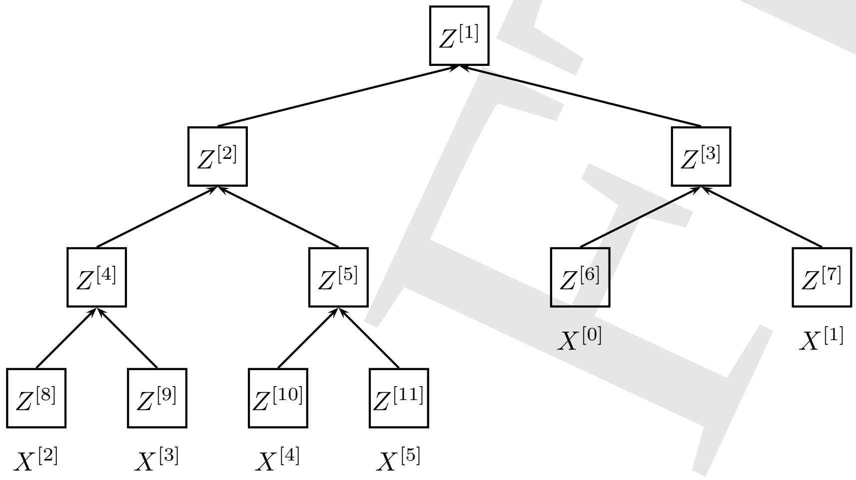

FIGURE 10. The balanced binary tree Z for $DaCAdd(6, X, Z)$.

PROOF. We show how to compute Z by an $\mathbf{NC}^1$ circuit. Note that if for each $x < a$ we simply construct an $\mathbf{AC}^0$ circuit that performs string addition to compute $Z^{[x]}$ from $Z^{[2x]}$ and $Z^{[2x+1]}$ (i.e. $Z^{[x]} = Z^{[2x]} + Z^{[2x+1]}$) and stack them together, the resulting circuit has depth $\mathcal{O}(\log(n))$ (where n is the number of input bits) but unbounded fan-in.

Here we use the fact that

$$X + Y + Z = G(X, Y, Z) + H(X, Y, Z) \tag{272}$$

where $G(X, Y, Z)$ is the string of bit-wise sums, and $H(X, Y, Z)$ is the string of carries:

$$\begin{aligned}
G(X, Y, Z)(z) &\leftrightarrow X(z) \oplus Y(z) \oplus Z(z)\\
H(X, Y, Z)(0) &\leftrightarrow \bot\\
H(X, Y, Z)(z+1) &\leftrightarrow ((X(z) \wedge Y(z)) \vee (X(z) \wedge Z(z)) \vee (Y(z) \wedge Z(z)))
\end{aligned}$$

EXERCISE 9.106. *Show that* $\mathbf{V}^0(G, H)$ *proves the equation* (272).

Thus, for each $Z^{[x]}$ we have a pair of strings $(S^{[x]}, C^{[x]})$ where $S^{[x]}$ is the string of bit-wise sums and $C^{[x]}$ is the string of carries; and for $1 \le x < a$,

$$Z^{[x]} = S^{[x]} + C^{[x]}$$

(For $a \le x < 2a$, we will take $S^{[x]} = S^{[x]}, C^{[x]} = \varnothing$.)

We need for $1 \le x < a$,

$$S^{[x]} + C^{[x]} = S^{[2x]} + C^{[2x]} + S^{[2x+1]} + C^{[2x+1]}$$

So

$$S^{[x]} = G(C^{[2x+1]}, U, V), \qquad C^{[x]} = H(C^{[2x+1]}, U, V)$$

where

$$U = G(S^{[2x]}, C^{[2x]}, S^{[2x+1]}), \qquad V = H(S^{[2x]}, C^{[2x]}, S^{[2x+1]})$$

In other words, let F_1, F_2 be the $\mathbf{AC}^0$ functions:

$$F_1(X, Y, Z, W) = G(W, G(X, Y, Z), H(X, Y, Z))$$
$$F_2(X, Y, Z, W) = H(W, G(X, Y, Z), H(X, Y, Z))$$

Then

$$S^{[x]} = F_1(S^{[2x]}, C^{[2x]}, S^{[2x+1]}, C^{[2x+1]}), \qquad C^{[x]} = F_2(S^{[2x]}, C^{[2x]}, S^{[2x+1]}, C^{[2x+1]})$$

In summary we need to prove in $\mathbf{VNC}^1$ the existence of S and C such that

$$\forall x < a, S^{[x+a]} = I^{[x]} \wedge C^{[x+a]} = \varnothing \wedge 0 < x \supset$$
$$S^{[x]} = F_1(S^{[2x]}, C^{[2x]}, S^{[2x+1]}, C^{[2x+1]}) \wedge C^{[x]} = F_2(S^{[2x]}, C^{[2x]}, S^{[2x+1]}, C^{[2x+1]})$$

Notice that for each z, the bits $S^{[x]}(z), C^{[x]}(z)$ are computed from the bits

$$\{S^{[2x]}(y), S^{[2x+1]}(y), C^{[2x]}(y), C^{[2x+1]}(y) : z \dot{-} 2 \leq y \leq z\}$$

(where we define $S^{[2x]}(y) \equiv \bot$ if $y < 0$, etc.). This is not in the form of the hypothesis of Theorem 9.101, but we can put it in the required form by transposing S and C. Recall the function *Transpose* from (248) (page 281). We will first compute $S_t = \mathit{Transpose}(b, b, S)$ and $C_t = \mathit{Transpose}(b, b, C)$ where $b = |I|$ is a sufficiently large bound.

Thus $S_t^{[z]}(x)$ and $C_t^{[z]}(x)$ are computed from

$$\{S_t^{[y]}(2x), S_t^{[y]}(2x+1), C_t^{[y]}(2x), C_t^{[y]}(2x+1) : z - 2 \leq y \leq z\}$$

by a Σ_0^B formulas. Therefore by Theorem 9.101, $\mathbf{VNC}^1$ proves the existence of S_t and C_t. $\dashv$

Notice that $\mathbf{V}^0$ proves the uniqueness of Z in (271). Define $\mathit{Sum}(a, X)$ as follows. First,

$$\mathit{Sum}(0, X) = \varnothing$$

For $a \geq 1$ we apply the above lemma for a *full* binary tree. Thus let a_1 be the smallest power of 2 that is $\geq a$, and define X_1 such that

(273) $\qquad X_1^{[x]} = X^{[x]}$ for $x < a$ and $X_1^{[x]} = \varnothing$ for $a \leq x < a_1$.

Let Z be the string that satisfies $\mathit{DaCAdd}(a_1, X_1, Z)$ as in Lemma 9.105 (see Figure 11). Define

$$\mathit{Sum}(a, X) = Z^{[1]}$$

It remains to show that $\overline{\mathbf{VNC}}^1(\mathit{Sum})$ proves (270), i.e.,

(274) $\qquad \mathit{Sum}(a+1, X) = \mathit{Sum}(a, X) + X^{[a]}$

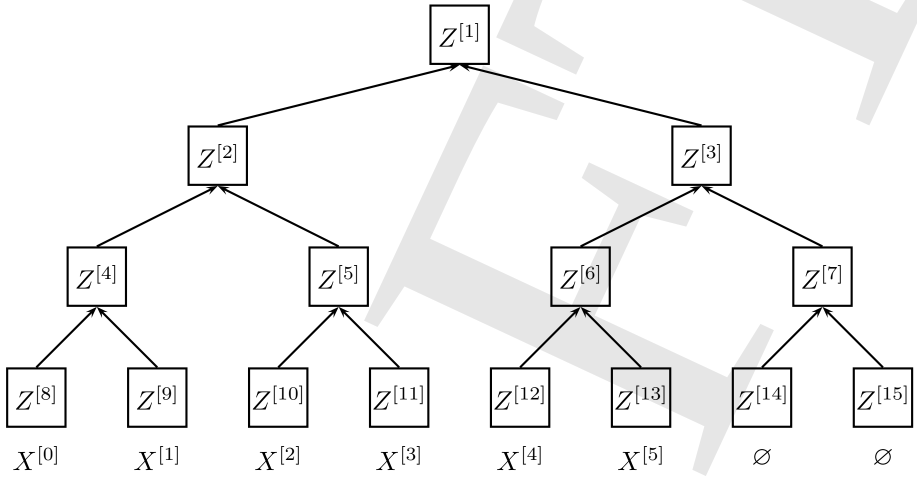

FIGURE 11. Defining $Sum(6, X)$ using a full binary tree.

When $a = 0$ this is straightforward. We consider the case where $a \geq 1$ and a is not a power of 2. Let X_1 be as in (273), and let X_2 be such that

$$X_2^{[x]} = X^{[x]} \text{ for } x \leq a \text{ and } X_2^{[x]} = \varnothing \text{ for } a < x < a_1.$$

Let Z_1 and Z_2 be such that $\varphi(a_1, X_1, Z_1)$ and $\varphi(a_1, X_2, Z_2)$ hold. By definition,

$$Sum(a, X) = Z_1^{[1]} \qquad \text{and} \qquad Sum(a+1, X) = Z_2^{[1]}$$

The trees Z_1 and Z_2 have the same height $h = \lceil \log(a_1) \rceil$. Note that $h + 1$ is the length of the binary representation of $(a_1 + a)$ Also, h is definable in $\mathbf{I}\Delta_0$ (see Section 3C.3). Let

$$d_0 = 1, d_1 = 3, d_2, \ldots, d_h = (a_1 + a)$$

be all initial segments of the binary representation of $(a_1 + a)$. Then

$$Z_2^{[d_0]}, Z_2^{[d_1]}, \ldots, Z_2^{[d_h]}$$

are all nodes in the tree Z_2 on the path from the root to the leaf $Z^{[a_1+a]} = X^{[a]}$.

Recall that the function $|x| = \lceil \log(x + 1) \rceil$ is definable in $\mathbf{I}\Delta_0$ (Section 3C.3). It can be proved by reverse induction on i that

$$(Z_2^{[d_i]} = Z_1^{[d_i]} + X^{[a]}) \;\wedge\; \forall x < a_1(|x| = |d_i| \wedge x < d_i \supset Z_2^{[x]} = Z_1^{[x]})$$

For $i = 0$ we obtain $Z_2^{[1]} = Z_1^{[1]} + X^{[a]}$.

The case where a is a power of 2 is left as an exercise. $\dashv$

EXERCISE 9.107. *Finish the proof of the theorem by showing that* (274) *is true when a is a power of 2.*

9E.4. The theory $\mathbf{VNC}^1\mathbf{V}$. In this section we will define $\mathbf{VNC}^1\mathbf{V}$ using 5-BNR, a bounded number recursion scheme that characterizes $\mathbf{FNC}^1$. This recursion theoretic characterization is based on Barrington's Theorem that asserts that $\mathbf{NC}^1$ is the class of relations computable by width 5 branching programs, or equivalently the word problem for the permutation group S_5 is complete for $\mathbf{NC}^1$.

Here the languages of $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$ consists of symbols for all $\mathbf{FNC}^1$ functions, but their defining axioms are based on 5-BNR rather than by $\mathbf{AC}^0$-reductions to the function *Fval* (that are used to define $\overline{\mathbf{VNC}}^1$). Recall the bounded number recursion (BNR) operation in Section 9C.3.

THEOREM 9.108 (Barrington). *A function is in $\mathbf{FNC}^1$ iff it can be obtained from the empty set of functions by finitely many applications of $\mathbf{AC}^0$ reduction and 5-BNR.*

By Theorem 9.7 it follows also that $\mathbf{FNC}^1$ is the class of functions obtained from $\mathbf{FAC}^0$ by finitely many applications of composition, string comprehension and 5-BNR.

DEFINITION 9.109. *The language $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$ is the smallest set that contains $\mathcal{L}_{\mathbf{FAC}^0}$ such that*

- *for each $\mathcal{L}^2_A$-term t and quantifier-free $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$-formula φ there is a function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$ with defining axiom* (85)*:*

$$F_{\varphi(z),t}(\vec{x},\vec{X})(z) \leftrightarrow z < t(\vec{x},\vec{X}) \wedge \varphi(z,\vec{x},\vec{X}) \tag{275}$$

- *for any number functions $g(\vec{x},\vec{X})$ and $h(y,z,\vec{x},\vec{X})$ in $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$, there is a number function $f_{g,h}(y,\vec{x},\vec{X})$ in $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$ with defining axiom (omitting $\vec{x},\vec{X}$)*

$$\begin{aligned}(g < 5 \supset f_{g,h}(0) = g) \wedge (g \geq 5 \supset f_{g,h}(0) = 0) \wedge \\ (h(y, f_{g,h}(y)) < 5 \supset f_{g,h}(y+1) = h(y, f_{g,h}(y))) \wedge \\ (h(y, f_{g,h}(y) \geq 5 \supset f_{g,h}(y+1) = 0)\end{aligned} \tag{276}$$

The next corollary follows from Theorem 9.108.

COROLLARY 9.110. (a) *A function is in $\mathbf{FNC}^1$ iff it is represented by a term in $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$.*

(b) *A relation is in $\mathbf{NC}^1$ iff it is represented by an open (or a Σ^B_0) formula of $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$.*

DEFINITION 9.111 ($\mathbf{VNC}^1\mathbf{V}$). *The theory $\mathbf{VNC}^1\mathbf{V}$ has vocabulary $\mathcal{L}_{\mathbf{VNC}^1\mathbf{V}}$ and axioms those of $\overline{\mathbf{V}}^0$ together with* (275) *for each function $F_{\varphi(z),t}$ and* (276) *for each function $f_{g,h}$.*

The next exercise can be proved as in Lemma 9.42 and Corollary 9.47.

EXERCISE 9.112. (a) *Show that the theory* $\mathbf{VNC^1V}$ *proves the axiom schemes*

$$\Sigma_0^B(\mathcal{L}_{\mathbf{VNC^1V}})\text{-}\mathbf{COMP}, \Sigma_0^B(\mathcal{L}_{\mathbf{VNC^1V}})\text{-}\mathbf{IND}, \text{ and } \Sigma_0^B(\mathcal{L}_{\mathbf{VNC^1V}})\text{-}\mathbf{MIN}$$

(b) *Show that for every* $\Sigma_0^B(\mathcal{L}_{\mathbf{VNC^1V}})$ *formula* φ^+ *there is a* Σ_1^B *formula* φ *so that* $\mathbf{VNC^1V} \vdash \varphi^+ \leftrightarrow \varphi$.

COROLLARY 9.113. (a) *A function is in* $\mathbf{FNC^1}$ *iff it is* Σ_1^B*-definable in* $\mathbf{VNC^1V}$.

(b) *A relation is in* $\mathbf{NC^1}$ *iff it is* Δ_1^B*-definable in* $\mathbf{VNC^1V}$.

PROOF SKETCH. (a) The proof is from Theorem 9.108 and Exercise 9.112 above, and is proved in the same way as Corollary 9.48.

(b) From (a) and Theorem 5.60. ⊣

THEOREM 9.114. $\mathbf{VNC^1V}$ *is a universal conservative extension of* $\mathbf{VNC^1}$.

We outline the proof below. For details see [63].

PROOF IDEA. To show that $\mathbf{VNC^1V}$ extends $\mathbf{VNC^1}$, the main task is to show that $\mathbf{VNC^1V} \vdash MFV$. The idea is to formalize in $\mathbf{VNC^1V}$ the proof that the Boolean Sentence Value Problem (see page 298) can be computed using width 5 branching programs (the $\Longrightarrow$ direction of Theorem 9.108). To show that $\mathbf{VNC^1V}$ is conservative over $\mathbf{VNC^1}$ essentially we need to show that width 5 branching programs can be simulated by families of $\mathbf{NC^1}$ circuits. The proof can be by induction on the definition of $\mathbf{VNC^1V}$. (See also Section 8B.2 for the proof that $\mathbf{VPV}$ is a universal conservative extension of $\mathbf{VP}$.) ⊣

9E.5. Theories for the NC hierarchy. We develop the theories for $\mathbf{AC}^k$ and $\mathbf{NC}^{k+1}$ using the fact that the Circuit Value Problem is complete for the respective classes under appropriate restriction on the given circuits. Consider encoding a layered, monotone Boolean circuit C with $(d+1)$ layers and n unbounded fan-in ($\wedge$ or $\vee$) gates on each layer. We need to specify the type (either $\wedge$ or $\vee$) of each gate, and the wires between the gates. Suppose that layer 0 contains the inputs which are specified by a string variable I of length $|I| \leq n$. To encode the gates on other layers, there is a string variable G such that for $1 \leq z \leq d$, $G(z,x)$ holds if and only if gate x on layer z is an $\wedge$-gate (otherwise it is an $\vee$-gate). Also, the wires of C are encoded by a 3-dimensional array E: $\langle z, x, y\rangle \in E$ iff the output of gate x on layer z is connected to the input of gate y on layer $z+1$.

The following algorithm computes the outputs of C using $(d+1)$ loops: in loop z it identifies all gates on layer z which output 1. It starts by identifying the input gates with the value 1. Then in each subsequent loop $(z+1)$ the algorithm identifies the following gates on layer $(z+1)$:

- $\vee$-gates that have at least one input which is identified in loop z;

- $\wedge$-gates all of whose inputs are identified in loop z.

The formula $\delta_{LMCV}(n, d, E, G, I, Y)$ below formalizes this algorithm (LMCV stands for "layered monotone circuit value"). The 2-dimensional array Y stores the result of computation: For $1 \leq z \leq d$, row $Y^{[z]}$ contains the gates on layer z that output 1.

$$\begin{aligned}(277)\quad \delta_{LMCV}(n, d, E, G, I, Y) \equiv \forall x < n \forall z < d\big((Y(0, x) \leftrightarrow I(x)) \wedge \\ \big(Y(z+1, x) \leftrightarrow (G(z+1, x) \wedge \forall u < n,\ E(z, u, x) \supset Y(z, u)) \vee \\ (\neg G(z+1, x) \wedge \exists u < n,\ E(z, u, x) \wedge Y(z, u))\big)\big)\end{aligned}$$

For $\mathbf{NC}^k$ we need the following formula which states that the circuit with underlying graph (n, d, E) has fan-in 2:

$$\begin{aligned}\mathit{Fanin2}(n, d, E) \equiv \forall z < d \forall x < n \exists u_1 < n \exists u_2 < n \\ \forall v < n\big(E(z, v, x) \supset (v = u_1 \vee v = u_2)\big)\end{aligned}$$

Recall (Section 3C.3) that the function $|x| = \lceil \log(x+1) \rceil$ is an $\mathbf{AC}^0$ function with a Δ_0 graph. Define the functions Lmcv_k and $\mathit{Lmcv}_{k,2}$ as follows:

$$\mathit{Lmcv}_k(n, E, G, I) = Y \leftrightarrow \big(|Y| \leq \langle n, |n|^k \rangle \wedge \delta_{LMCV}(n, |n|^k, E, G, I, Y)\big)$$

and

$$\begin{aligned}\mathit{Lmcv}_{k,2}(n, E, G, I) = Y \leftrightarrow \big((\neg \mathit{Fanin2}(n, d, E) \wedge Y = \varnothing) \vee \\ (\mathit{Fanin2}(n, d, E) \wedge |Y| \leq \langle n, |n|^k \rangle \wedge \delta_{LMCV}(n, |n|^k, E, G, I, Y))\big)\end{aligned}$$

Theorem 9.115. *For $k \geq 1$, Lmcv_k is $\mathbf{AC}^0$ many-one complete for $\mathbf{AC}^k$. For $k \geq 2$, $\mathit{Lmcv}_{k,2}$ is $\mathbf{AC}^0$ many-one complete for $\mathbf{NC}^k$.*

Proof Sketch. First, it is easy to see that every function in uniform $\mathbf{AC}^k$ (resp. $\mathbf{NC}^k$) is $\mathbf{AC}^0$ many-one reducible to Lmcv_k (resp. $\mathit{Lmcv}_{k,2}$). It remains to show that the Lmcv functions belong to the respective classes.

We show that Lmcv_1 is in $\mathbf{AC}^1$. The argument for Lmcv_k in general is similar. Consider a tuple (n, d, E, G, I) that encodes an unbounded fan-in circuit C of depth $d \leq c\log(n)$ for some $c \in \mathbb{N}$, and I encodes the inputs to C. For each $z \leq c\log(n)$ and $x \leq n$ we construct a constant-depth sub-circuit $K_{z,x}$ that computes the output of gate $\langle z, x \rangle$ (gate numbered x on layer z) in C. The inputs to $K_{z,x}$ are the bits of E (that specify the inputs to gate $\langle z, x \rangle$) and the output of other gates $K_{z-1,y}$. In particular, $K_{z,x}$ computes the following formula (recall that $G(z, x)$

holds iff gate $\langle z, x\rangle$ is an $\wedge$ gate):

$$\big(G(z,x) \wedge \bigwedge_{y<n} (E(z-1,y,x) \supset K(z-1,y))\big) \vee$$
$$\big(\neg G(z,x) \wedge \bigvee_{y<n} (E(z-1,y,x) \wedge K(z-1,y))\big)$$

Our $\mathbf{AC}^1$ circuit that computes $Lmcv_1(n,d,E,G,I)$ is obtained by stack the sub-circuits $K_{z,x}$ together. To make sure that it has depth $\log(m)$ where m is the length of the encoding of (n,d,E,G,I), we require that m is at least n^c whenever $(c-1)\log(n) < d \leq c\log(n)$.

Now we show that $Lmcv_{2,2}$ is in $\mathbf{NC}^2$ (the argument for $Lmcv_{k,2}$ where $k > 2$ is similar). Suppose that (n,d,E,G,I) encodes a circuit C of of fan-in 2 and depth $d \leq c\log(n)$ for some $c \in \mathbb{N}$, and I encodes the inputs to C. We use a $\log\log(n)$-depth unbounded fan-in circuit K that computes whether there is a path in E from a gate $\langle z, y\rangle$ to $\langle z', x\rangle$ for any $z < z' \leq d$, $z' \leq z + \log(n)$ and $x, y < n$

Using circuit K we can evaluate each $\log(n)$-depth sub-circuit of C rooted at gate $\langle z, x\rangle$ by a sub-circuit $K_{z,x}$ of depth $\mathcal{O}(\log(n)$. Our $\mathbf{NC}^2$ circuit computing $Lmcv_{2,2}$ is obtained by stacking the sub-circuits $K_{i\log(n),x}$ together (on top of K), for $i \leq c$. Note that K can be simulated by a bounded fan-in circuit of depth $\mathcal{O}(\log(n))$. Again, we can make sure that the resulting circuit has depth $(\log(m))^2$, where m is the length of the encoding of (n,d,E,G,I), by requiring that $m \geq n^{c'}$ whenever $(c-1)\log(n) < d \leq c\log(n)$ for some c' depending on c. ⊣

Note that we do not know whether $Lmcv_{1,2}$ is in $\mathbf{NC}^1$.

Definition 9.116 ($\mathbf{VAC}^k$ and $\mathbf{VNC}^k$). *For $k \geq 1$, the theory $\mathbf{VAC}^k$ has vocabulary $\mathcal{L}_A^2$ and is axiomatized by $\mathbf{V}^0$ and the axiom*

$$\forall n \forall E \forall G \forall I \exists Y\, \delta_{LMCV}(n, |n|^k, E, G, I, Y)$$

For $k \geq 2$, $\mathbf{VNC}^k$ has vocabulary $\mathcal{L}_A^2$ and is axiomatized by $\mathbf{V}^0$ and the axiom

$$\forall n \forall E \forall G \forall I (\mathit{Fanin2}(n, |n|^k, E) \supset \exists Y\, \delta_{LMCV}(n, |n|^k, E, G, I, Y))$$

It is straightforward to show that the aggregate functions $Lmcv_k^\star$ for $k \geq 1$ (resp. $Lmcv_{k,2}^\star$, for $k \geq 2$) is Σ_1^B-definable in $\mathbf{VAC}^k$ (resp. $\mathbf{VNC}^k$, for $k \geq 2$). Details are left as an exercise.

Exercise 9.117. *Show that for $k \geq 1$ $Lmcv_k^\star$ and $Lmcv_{k+1,2}^\star$ are respectively Σ_1^B-definable in $\mathbf{VAC}^k$ and $\mathbf{VNC}^{k+1}$.*

This can be used to show, as in Section 8A or 9B, the following result:

Corollary 9.118. *For $k \geq 1$:*

(a) *The Σ_1^B-definable functions of $\mathbf{VAC}^k$ (resp. $\mathbf{VNC}^{k+1}$) are precisely the functions in $\mathbf{FAC}^k$ (resp. $\mathbf{FNC}^{k+1}$).*

(b) *The Δ_1^B-definable functions of* $\mathbf{VAC}^k$ *(resp.* $\mathbf{VNC}^{k+1}$*) are precisely the relations in* $\mathbf{AC}^k$ *(resp.* $\mathbf{NC}^{k+1}$*).*

Corollary 9.119. (a) *A function is in* **FNC** *iff it is* Σ_1^B*-definable in* $\mathbf{VAC}^k$ *for some* $k \geq 0$.

(b) *A relation is in* **NC** *iff it is* Δ_1^B*-definable in* $\mathbf{VAC}^k$ *for some* $k \geq 0$.

9F. Theories for NL and L

The class **NL** (resp. **L**) is the class of problems solvable by a nondeterministic (resp. deterministic) Turing machine in space $\mathcal{O}(\log n)$. It is straightforward that $\mathbf{L} \subseteq \mathbf{NL}$ and both are subclasses of **P**. In fact, it can be shown that $\mathbf{NL} \subseteq \mathbf{AC}^1$ (see Exercise 9.129 below). It is also easy to see that **L** is closed under $\mathbf{AC}^0$ reduction, while for **NL** this follows from the important theorem of Immerman and Szelepcsényi which states that **NL** is closed under complement.

The theory **VNL** is developed using the fact that the *st*-Connectivity (*st*-CONN) problem is $\mathbf{AC}^0$-complete for **NL**. Here the problem is to decide, for a given graph G and two designated vertices s and t, whether there is a path from s to t in G.

The Krom formulas are propositional formulas in conjunctive normal form where each clause contains at most two literals. The Krom-SAT problem, which is the problem of deciding whether a given Krom formula is satisfiable, is known to be complete for *co*-**NL** (and hence also for **NL**). It has been used to develop the theory $\mathbf{V}^1$-**KROM** in the same style as $\mathbf{V}^1$-**HORN** (Section 8D). We will show that $\mathbf{V}^1$-**KROM** is equivalent to **VNL**.

Now consider a restricted version of the *st*-CONN problem where every vertex in G has out-degree at most one. This is called the PATH problem and it is $\mathbf{AC}^0$-many-one complete for **L**. We will use this fact to develop the triple **VL**, $\widehat{\mathbf{VL}}$ and $\overline{\mathbf{VL}}$ in the family of theories discussed in Section 9B.

Finally, the bounded number recursion scheme *p*BNR (Section 9C.3) can be used to characterize **FL**. Based on this we will develop a universal theory call **VLV** in the style of **VPV** and $\mathbf{VTC}^0\mathbf{V}$. Here the language of **VLV** contains symbols for every functions in **FL**. Their defining axioms are based on *p*BNR.

This section is organized as follows. First, we define the theory **VNL** and its universal conservative extensions $\widehat{\mathbf{VNL}}$, $\overline{\mathbf{VNL}}$ in Section 9F.1. Then we define $\mathbf{V}^1$-**KROM** and show that it is equivalent to **VNL** in Section 9F.2. In Section 9F.3 we define **VL**, $\widehat{\mathbf{VL}}$ and $\overline{\mathbf{VL}}$. Finally, in Section 9F.4 we define **VLV**.

9F.1. The theories VNL, $\widehat{\mathbf{VNL}}$ and $\overline{\mathbf{VNL}}$. The theories **VNL**, $\widehat{\mathbf{VNL}}$ and $\overline{\mathbf{VNL}}$ are developed based on the fact that the *st*-CONN problem is complete for **NL**. First, we need to formalize this problem. Here we encode a directed graph G by a pair (a, E) as follows:

- a is the number of vertices in G, and the vertices of G are numbered $0, \ldots, (a-1)$, and
- for $x, y < a$, $E(x, y)$ holds if and only if there is a directed edge from x to y in G.

Our designated "source" s is always the vertex 0. Consider the algorithm that solves the *st*-CONN problem by inductively computing all vertices in G that have distance from s at most $0, 1, \ldots, (a-1)$. The formula $\delta_{CONN}(a, E, Y)$ below states that $Y^{[z]}$ is the set of all vertices with distance at most z from 0 (recall that $x \in Y^{[z]} \equiv Y(z, x)$):

(278) $\delta_{CONN}(a, E, Y) \equiv Y(0,0) \wedge \forall x < a(x \neq 0 \supset \neg Y(0, x)) \ \wedge$

$$\forall z < a \forall x < a\big(Y(z+1, x) \leftrightarrow (Y(z, x) \vee \exists y < a(Y(z, y) \wedge E(y, x)))\big)$$

We define the relation R_{CONN} below by assigning the "target" vertex t number 1.

DEFINITION 9.120.

$$R_{CONN}(a, E) \leftrightarrow \exists Y \leq \langle a, a \rangle (\delta_{CONN}(a, E, Y) \wedge Y(a, 1))$$

THEOREM 9.121. *The relation R_{CONN} is in* **NL**, *and for every relation $R(\vec{x}, \vec{X})$ in* **NL** *there are* $\mathbf{AC}^0$ *functions a_0, E_0 such that*

$$R(\vec{x}, \vec{X}) \leftrightarrow R_{CONN}(a_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$$

PROOF SKETCH. The fact that R_{CONN} is in **NL** is straightforward: on input (a, E) the NTM guesses a path from 0 to 1 by enumerating the edges on the path.

Now let $R(\vec{x}, \vec{X})$ be a relation in **NL**, so R is accepted by a nondeterministic Turing machine M that works in logspace. Suppose without loss of generality that M has a unique accepting configuration. The configurations of M (without the input tape content) can be encoded by numbers $< t(\vec{x}, \vec{X})$ (for some number term t bounding the running time of M) such that 0 is the initial configuration and 1 is the only accepting configuration (see also Exercise 6.13). Consider the directed graph G with vertices the numbers $< t(\vec{x}, \vec{X})$ and there is an edge from z_1 to z_2 iff z_2 is a next configuration of z_1. Then M accepts $(\vec{x}, \vec{X})$ iff there is a path from 0 to 1 in G.

The fact that z_1 encodes a next configuration of z_2 can be expressed by a Σ_0^B formula $\varphi(z_1, z_2, \vec{x}, \vec{X})$. In other words, there is an $\mathbf{AC}^0$ string function $E_0(\vec{x}, \vec{X})$ so that for $z_1, z_2 < t(\vec{x}, \vec{X})$,

$$E_0(\vec{z}, \vec{X})(z_1, z_2) \leftrightarrow \varphi(z_1, z_2, \vec{x}, \vec{X})$$

Consequently M accepts $(\vec{x}, \vec{X})$ iff $R_{CONN}(t(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$ holds. $\dashv$

Definition 9.122 (**VNL**). *The theory* **VNL** *has vocabulary* $\mathcal{L}_A^2$ *and is axiomatized by the axioms of* $\mathbf{V}^0$ *together with the axiom CONN, where*

$$CONN \equiv \forall a \forall E \exists Y\, \delta_{CONN}(a, E, Y)$$

The string Y above can be bounded by $\langle a, a\rangle$. So **VNL** is a polynomial-bounded theory.

Let the string function $Conn(z, a, E)$ be the set of all vertices of the graph $G = (a, E)$ that have distance from 0 at most z. It has the following quantifier-free defining axioms (over $\mathcal{L}_{\mathbf{FAC}^0}$):

$$Conn(0, a, E) = \{0\} \tag{279}$$

and

$$\begin{aligned} &Conn(z+1, a, E) = \\ &\qquad Conn(z, a, E) \cup \{x < a \mid Conn(z, a, E) \cap IP(x, a, E) \neq \varnothing\} \end{aligned} \tag{280}$$

Here $\cup$, $\cap$ and IP are $\mathbf{AC}^0$ functions: $X \cup Y$ (resp. $X \cap Y$) is the union (resp. intersection) of the sets X and Y, and $IP(x, a, E)$ is the set of all *immediate predecessors* of x in the graph (a, E). They can be defined as follows:

$$\begin{aligned} X \cup Y = Z &\leftrightarrow \big(|Z| \le |X| + |Y| \wedge z < |Z| \supset Z(z) \leftrightarrow (X(z) \vee Y(z))\big) \\ X \cap Y = Z &\leftrightarrow \big(|Z| \le |X| \wedge z < |Z| \supset Z(z) \leftrightarrow (X(z) \wedge Y(z))\big) \\ IP(x, a, E) = Z &\leftrightarrow \big(|Z| \le a \wedge y < a \supset (Z(y) \leftrightarrow E(y, x))\big) \end{aligned}$$

Proposition 9.123. *The function Conn is* $\mathbf{AC}^0$*-many-one complete for* **FNL**.

The next lemma is straightforward:

Lemma 9.124. *The function Conn is* Σ_1^B*-definable in* **VNL**.

To show that the Σ_1^B-definable functions of **VNL** comprise **FNL** functions we need the following result whose proof is left as an exercise:

Exercise 9.125. *Show that the function* $Conn^\star$ *is* Σ_1^B*-definable in* **VNL**, *and that* $\mathbf{VNL}(Row, Conn, Conn^\star)$ *proves* (165)*:*

$$\forall i < b,\ Conn^\star(b, X, E)^{[i]} = Conn((X)^i, E^{[i]})$$

Following the method of Sections 8A and 9B we will define the theories $\widehat{\mathbf{VNL}}$ and $\overline{\mathbf{VNL}}$.

Definition 9.126 ($\widehat{\mathbf{VNL}}$). *Let* $\mathcal{L}_{\widehat{\mathbf{VNL}}} = \mathcal{L}_{\mathbf{FAC}^0} \cup \{Conn\}$. $\widehat{\mathbf{VNL}}$ *is the theory with vocabulary* $\mathcal{L}_{\widehat{\mathbf{VNL}}}$ *and is axiomatized by the axioms of* $\overline{\mathbf{V}}^0$ *together with* (279), (280).

DEFINITION 9.127 ($\overline{\mathbf{VNL}}$). *The vocabulary $\mathcal{L}_{\mathbf{FNL}}$ is the smallest set that contains $\mathcal{L}_{\widehat{\mathbf{VNL}}}$ such that for every $\mathcal{L}_A^2$-term t and every quantifier-free $\mathcal{L}_{\mathbf{FNL}}$-formula φ the function $F_{\varphi(z),t}$ with defining axiom* (85)*:*

$$F_{\varphi(z),t}(\vec{x},\vec{X})(z) \leftrightarrow z < t(\vec{x},\vec{X}) \wedge \varphi(z,\vec{x},\vec{X}) \tag{281}$$

is in $\mathcal{L}_{\mathbf{FNL}}$.

The theory $\overline{\mathbf{VNL}}$ has vocabulary $\mathcal{L}_{\mathbf{FNL}}$ and is axiomatized by the axioms of $\widehat{\mathbf{VNL}}$ and (281) *for each function $F_{\varphi(z),t}$.*

The Definability Theorems for our theories here follow from our discussion in Section 9B.

COROLLARY 9.128. *Here either $\mathcal{L}$ is $\mathcal{L}_{\widehat{\mathbf{VNL}}}$ and $\mathcal{T}$ is $\widehat{\mathbf{VNL}}$, or $\mathcal{L}$ is $\mathcal{L}_{\mathbf{FNL}}$ and $\mathcal{T}$ is $\overline{\mathbf{VNL}}$.*

(a) *A function is in* **FNL** *iff it is represented by a term in $\mathcal{L}_{\widehat{\mathbf{VNL}}}$ (and for a string function) iff it is represented by a function symbol in $\mathcal{L}_{\mathbf{FNL}}$. A relation is in* **NL** *iff it is represented by an open (or a Σ_0^B) formula of $\mathcal{L}$.*
(b) *For every $\Sigma_0^B(\mathcal{L})$ formula φ^+ there is a $\Sigma_1^B(\mathcal{L}_A^2)$ formula φ so that $\mathcal{T} \vdash \varphi^+ \leftrightarrow \varphi$.*
(c) $\mathcal{T} \vdash \Sigma_0^B(\mathcal{L})$-**COMP**.
(d) $\overline{\mathbf{VNL}}$ *is a universal conservative extension of $\widehat{\mathbf{VNL}}$ which is in turn a universal conservative extension of* **VNL**.
(e) *The Σ_1^B-definable functions of* **VNL** *(or $\mathcal{T}$) are precisely functions in* **FNL**.
(f) *The Δ_1^B-definable relations of* **VNL** *(or $\mathcal{T}$) are precisely relations in* **NL**.

EXERCISE 9.129. *Recall the theory $\mathbf{VAC}^1$ from Section 9E.5. Show that $\mathbf{VNL} \subseteq \mathbf{VAC}^1$.*

9F.2. The theory $\mathbf{V}^1$-KROM. A Krom formula is a propositional formula in conjunctive normal form where each clause contains at most two literals. The Satisfiability Problem for Krom formulas, Krom-SAT, is complete for *co*-**NL** (or equivalently **NL**, by Immerman-Szelepcsényi Theorem). In descriptive complexity theory Grädel's Theorem states that **NL** is the class of finite models of the second-order Krom formulas [38]. These have been used to develop $\mathbf{V}^1$-**KROM**.

First, we will define Σ_1^1-**Krom** formulas which are Σ_1^1 and resemble the propositional Krom formulas. In Theorem 9.134 we show that Σ_1^1-**Krom** formulas represent precisely *co*-**NL** relations.

DEFINITION 9.130 (Σ_1^1-**Krom** Formula). *A $\Sigma_1^B(\mathcal{L}_A^2)$ formula $\psi(\vec{x},\vec{X})$ is called a Σ_1^1-**Krom** formula if it is of the form:*

$$\exists P_1 \dots \exists P_k \forall z_1 \le t_1(\vec{x},\vec{X}) \dots \forall z_m \le t_m(\vec{x},\vec{X}) \varphi(\vec{z},\vec{P},\vec{x},\vec{X}) \tag{282}$$

where t_i are $\mathcal{L}_A^2$-terms and $\varphi(\vec{z},\vec{P},\vec{x},\vec{X})$ is a quantifier-free formula in conjunctive normal form and each clause contains at most two literals *of the form $P_j(s(\vec{z}, \vec{x}, \vec{X}))$ or $\neg P_j(s(\vec{z}, \vec{x}, \vec{X}))$ for some number term s. A clause in φ may contain other quantifier-free subformulas, but no term of the form $|P_j|$ may occur in φ.*

Notice that $\Sigma_0^B \not\subseteq \Sigma_1^1$-**Krom**, although we will show later (Theorem 9.142) that each Σ_0^B formula is equivalent in the theory $\mathbf{V}^0$ to a Σ_1^1-**Krom** formula.

EXAMPLE 9.131 (Transitive Closure in Graphs). *Suppose that a graph G is coded by (a, E) as before (page 314). The formula $ContainTC(a, E, P)$ below states that P contains the transitive closure of G, i.e., if there is a path from x to y in G, then $P(x, y)$ holds:*

$$ContainTC(a, E, P) \equiv \forall x < a \forall y < a \forall z < a,$$
$$(E(x, y) \supset P(x, y)) \wedge (P(x, y) \wedge E(y, z) \supset P(x, z))$$

*The following Σ_1^1-**Krom** formula states that there is* no *path from x_1 to x_2 in G:*

(283) $\varphi_{\neg Reach}(x_1, x_2, a, E) \equiv \exists P(ContainTC(a, E, P) \wedge \neg P(x_1, x_2))$

The set Y that satisfies the comprehension for $\varphi_{\neg Reach}$:

$$|Y| \leq a \wedge \forall y < a(Y(y) \leftrightarrow \varphi_{\neg Reach}(x, y, a, E))$$

is the set of all vertices that are not *reachable from vertex x.*

The formula φ in (282) is a quantifier-free formula. In some cases it is convenient to allow the non-P_i part of φ to be a Σ_0^B formula. The next lemma shows that this is possible.

LEMMA 9.132. *Suppose that ψ is a Σ_1^1 formula*

$$\exists \vec{P} \forall \vec{z} \leq \vec{t} \bigwedge_i \varphi_i(\vec{z}, \vec{P}) \tag{284}$$

where each formula φ_i is a disjunction of the form

$$\ell \vee \ell' \vee \rho_i(\vec{z})$$

*where ℓ, ℓ' are literals of the form $P_j(\vec{s})$ or $\neg P_j(\vec{s})$ (for some number terms $\vec{s}$ not containing any of $\vec{P}$) and ρ_i is a Σ_0^B formula that does not contain any of $\vec{P}$. Then ψ is equivalent in $\mathbf{V}^0$ to a Σ_1^1-**Krom** formula.*

COROLLARY 9.133. *Suppose that ψ is a formula of the form* (284) *where now the formulas ρ_i are quantifier-free over $\mathcal{L}_{\mathbf{FAC}^0}$ (the number terms $\vec{s}$ in $P_i(\vec{s})$ are still $\mathcal{L}_A^2$-term). Then ψ is equivalent in $\mathbf{V}^0$ to a Σ_1^1-**Krom** formula.*

PROOF OF LEMMA 9.132. We prove the lemma by structural induction on the formulas ρ_i. Assume w.o.l.g. that they are in prenex form. The base case (all ρ_i are quantifier-free) is obvious. Consider the the induction step. First suppose that for some i the formula ρ_i has the form

$$\forall u \leq t \rho_i'(u, \vec{z})$$

Let $\varphi'(u,\vec{z},\vec{P})$ be obtained from $\varphi_i(\vec{z},\vec{P})$ by replacing ρ_i by ρ'_i. Then

$$\psi \leftrightarrow \exists \vec{P}\forall \vec{z} \leq \vec{t}\forall u \leq t\big(\varphi'_i(u,\vec{z},\vec{P}) \wedge \bigwedge_{j\neq i} \varphi_j(\vec{z},\vec{P})\big)$$

Now consider the case where $\rho_i(\vec{z},\vec{P}) \equiv \exists u \leq t\rho'_i(u,\vec{z},\vec{P})$. Suppose w.o.l.g. that

$$\varphi_i \equiv (P_1(\vec{s}) \wedge \forall u \leq t\rho'_i(u,\vec{z})) \supset P_2(\vec{r})$$

In the following formula we introduce a new variable Q so that $Q(v)$ encodes the truth value of $P_1(\vec{s}) \wedge \forall u \leq v\rho'_i(u,\vec{z})$:

$$\begin{aligned}\varphi'(u,\vec{z},Q,P_1,P_2) \equiv{}& (P_1(\vec{s}) \wedge \rho'_i(0,\vec{z}) \supset Q(0)) \wedge \\ & (Q(u) \wedge \rho'_i(u,\vec{z}) \supset Q(u+1)) \wedge (Q(t+1) \supset P_2(\vec{r}))\end{aligned}$$

It is easy to see that

$$\psi \leftrightarrow \exists \vec{P}\exists Q\forall \vec{z} \leq \vec{t}\forall u \leq t\big(\varphi'_i(u,\vec{z},Q,P_1,P_2) \wedge \bigwedge_{j\neq i} \varphi_j(\vec{z},\vec{P})\big)$$

⊣

THEOREM 9.134. *A relation is represented by a* Σ^1_1-**Krom** *formula if and only if it is in co*-**NL**.

PROOF. First we prove the ONLY IF direction. Let $R(\vec{x},\vec{X})$ be a relation represented by the Σ^1_1-**Krom** formula (282):

$$\exists P_1 \ldots \exists P_k \forall z_1 \leq t_1(\vec{x},\vec{X}) \ldots \forall z_m \leq t_m(\vec{x},\vec{X})\varphi(\vec{z},\vec{P},\vec{x},\vec{X})$$

For a given input $(\vec{x},\vec{X})$, let v_i be the value of t_i (for $1 \leq i \leq m$). Now for each $(z_1,z_2,\ldots,z_m)$ where

$$0 \leq z_i \leq v_i \qquad (\text{for } 1 \leq i \leq m)$$

we treat the atoms of the form $P_j(s(\vec{z},\vec{x},\vec{X}))$ as propositional variables. Since all terms and other variables in φ can be evaluated, $\varphi(\vec{z},\vec{P},\vec{x},\vec{X})$ can be made into a Krom formula $A_{z_1,\ldots,z_m}$ whose variables are of the form $P_j(s(\vec{z},\vec{x},\vec{X}))$. Semantically,

$$\forall z_1 \leq t_1(\vec{x},\vec{X}) \ldots \forall z_m \leq t_m(\vec{x},\vec{X})\varphi(\vec{z},\vec{P},\vec{x},\vec{X})$$

is equivalent to the Krom formula

$$\bigwedge_{z_1=0}^{v_1} \cdots \bigwedge_{z_m=0}^{v_m} A_{z_1,\ldots,z_m} \tag{285}$$

Therefore $(\vec{x},\vec{X}) \in R$ iff (285) is satisfiable.

Notice that (285) can be obtained from the formula (282) and $(\vec{x},\vec{X})$ in deterministic logspace (in fact, $\mathbf{AC}^0$). So to show that R is in *co*-**NL**, it suffices to give a nondeterministic logspace algorithm that accepts $(\vec{x},\vec{X})$ precisely when (285) is unsatisfiable.

The formula (285) is unsatisfiable iff it contains a set of clauses of the form:

(286)

$$\ell_0 \supset \ell_1, \ell_1 \supset \ell_2,\ \ldots,\ \ell_k \supset \neg\ell_0,\ \neg\ell_0 \supset \ell'_1,\ \ell'_1 \supset \ell'_2,\ \ldots,\ \ell'_n \supset \ell_0$$

for some literals ℓ_i, ℓ'_j. The existence of such a set can be easily guessed and verified in logspace.

Now we prove the IF direction. Suppose that $R(\vec{x}, \vec{X})$ is a *co*-**NL** relation, we show that R can be represented by a Σ^1_1-**Krom** formula. By Proposition 9.121 there are $\mathbf{AC}^0$ functions $a_0(\vec{x}, \vec{X})$ and $E_0(\vec{x}, \vec{X})$ so that

$$R(\vec{x}, \vec{X}) \leftrightarrow \neg R_{CONN}(a_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$$

i.e., $(\vec{x}, \vec{X}) \in R$ iff 1 is not reachable from 0 in the graph $(a_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$. Thus by Example 9.131,

$$R(\vec{x}, \vec{X}) \leftrightarrow \varphi_{\neg Reach}(0, 1, a_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$$

By Corollary 9.133 $\varphi_{\neg Reach}(0, 1, a_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$ is equivalent in $\mathbf{V}^0$ to a Σ^1_1-**Krom** formula. ⊣

Definition 9.135. *The theory* $\mathbf{V}^1$-**KROM** *has vocabulary* $\mathcal{L}^2_A$ *and is axiomatized by* 2-**BASIC** *(Figure 2) and the comprehension axiom scheme for all* Σ^1_1-**Krom** *formulas.*

Although $\Sigma^B_0 \not\subseteq \Sigma^1_1$-**Krom**, we will show that $\mathbf{V}^1$-**KROM** extends $\mathbf{V}^0$:

Lemma 9.136. $\mathbf{V}^0 \subseteq \mathbf{V}^1$-**KROM**.

First we prove:

Lemma 9.137. $\mathbf{V}^1$-**KROM** *proves the multiple comprehension axioms (see Lemma 5.50) for quantifier-free formulas.*

Proof. We have to show that $\mathbf{V}^1$-**KROM** proves

(287)

$$\exists X \le \langle y_1, \ldots, y_k\rangle \forall z_1 < y_1 \ldots \forall z_k < y_k (X(z_1, \ldots, z_k) \leftrightarrow \varphi(z_1, \ldots, z_k))$$

for any quantifier-free formula φ.

A first attempt to prove this lemma might be to show that

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X \le \langle \vec{y}\rangle \forall x < \langle \vec{y}\rangle \big(X(x) \leftrightarrow \exists \vec{z} < \vec{y}(x = \langle \vec{z}\rangle \wedge \varphi(\vec{z}))\big)$$

However, $\exists \vec{z} < \vec{y}(x = \langle \vec{z}\rangle \wedge \varphi(\vec{z}))$ is not a Σ^1_1-**Krom** formula.

Here we prove (287) using Σ^1_1-**Krom-COMP** as follows. Let X satisfy:

$$\exists X \le \langle \vec{y}\rangle \forall x < \langle \vec{y}\rangle \ \big(X(x) \leftrightarrow \exists P \forall \vec{z} < \langle \vec{y}\rangle \big((P(\langle \vec{z}\rangle) \leftrightarrow \varphi(\vec{z})) \wedge P(x)\big)\big)$$

It is straightforward to verify that such X also satisfies (287). ⊣

Proof of Lemma 9.136. We prove the lemma by showing that $\mathbf{V}^1$-**KROM** proves the multiple comprehension axiom for any Σ_0^B formula φ. The proof is by structural induction on φ. Assume without loss of generality that φ is in prenex form.

The base case, where φ is a quantifier-free formula, follows from the lemma above because a quantifier-free formula is also in Σ_1^1-**Krom**.

For the induction step, suppose that we need to prove

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X \leq \langle \vec{a} \rangle \forall \vec{x} < \vec{a},\, X(\vec{x}) \leftrightarrow \varphi(\vec{x}) \tag{288}$$

First consider the case where $\varphi(\vec{x}) \equiv \forall z < a\, \psi(\vec{x}, z)$. By the induction hypothesis for ψ,

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X' \leq \langle \vec{a}, a \rangle \forall \vec{x} < \vec{a} \forall z < a,\, X'(\vec{x}, z) \leftrightarrow \psi(\vec{x}, z)$$

Now we can apply the multiple comprehension axiom for the Σ_1^1-**Krom** formula $\forall z < a\, X'(\vec{x}, z)$:

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X \leq \langle \vec{a} \rangle \forall \vec{x} < \vec{a},\, X(\vec{x}) \leftrightarrow \forall z < a\, X'(\vec{x}, z)$$

Such X satisfies (288).

Finally suppose that $\varphi(\vec{x}) \equiv \exists z < a\, \psi(\vec{x}, z)$. Let $\psi'(\vec{x}, z)$ be the prenex formula equivalent to $\neg\psi(\vec{x}, z)$ obtained by pushing the $\neg$ connective through the block of quantifiers using DeMorgan's laws. By the previous case

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X' \leq \langle \vec{a} \rangle \forall \vec{x} < \vec{a},\, X'(\vec{x}) \leftrightarrow \forall z < a\, \psi'(\vec{x}, z)$$

Let X be such that

$$|X| \leq \langle \vec{a} \rangle \wedge \forall \vec{x} < \vec{a},\, X(\vec{x}) \leftrightarrow \neg X'(\vec{x})$$

Then X satisfies (288). ⊣

Now we prove the main result of this section. The proof ends with Exercise 9.141 on page 323.

Theorem 9.138. $\mathbf{V}^1$-**KROM** = **VNL**.

Proof. First we show that $\mathbf{VNL} \subseteq \mathbf{V}^1$-**KROM**. By Lemma 9.136 above, $\mathbf{V}^1$-**KROM** is an extension of $\mathbf{V}^0$. It remains to show that $\mathbf{V}^1$-**KROM** proves the axiom *CONN* (Definition 9.122).

The fact that $\mathbf{V}^1$-**KROM** extends $\mathbf{V}^0$ also gives us:

Claim. $\mathbf{V}^1$-**KROM** proves the multiple comprehension axiom scheme (Lemma 5.50) for Σ_1^1-**Krom** formulas. For each Σ_1^1-**Krom** formula φ, $\mathbf{V}^1$-**KROM** proves the comprehension for $\neg\varphi$.

Recall that in the formula $\delta_{CONN}(a, E, Y)$ in (278), $Y(z, x)$ holds iff in the graph G coded by (a, E) there is a path from 0 to x of length $\leq z$. The Σ_1^1-**Krom** formula $\varphi_{\neg Dist}(x_1, x_2, z, a, E)$ below states that there is *no* path from x_1 to x_2 in G of length $\leq z$. The string variable P codes a (superset of) the "connectivity to x_1" relation, i.e.,

if there is a path from x_1 to y of length $\leq u$ then $P(u, y)$ holds.

($P(u, y)$ might hold even if there is no x_1y path of length $\leq u$.)

$$\varphi_{\neg Dist}(x_1, x_2, z, a, E) \equiv \exists P \forall u < z \forall x < a \forall y < a$$
$$\neg P(z, x_2) \wedge P(0, x_1) \wedge (P(u, x) \wedge E(x, y) \supset P(u+1, y))$$

By the claim above, $\mathbf{V}^1$**-KROM** proves the existence of Y such that

$$\forall z < a \forall x < a,\ Y(z, x) \leftrightarrow \neg\varphi_{\neg Dist}(0, x, z, a, E)$$

In other words, $Y(z, x)$ holds iff the distance from 0 to x is at most z, i.e., Y satisfies $\delta_{CONN}(a, E, Y)$ (278). The formal argument is left as an exercise.

EXERCISE 9.139. *Show that*

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \delta_{CONN}(a, E, Y)$$

Now we show that $\mathbf{V}^1$**-KROM** $\subseteq$ **VNL**. Let

$$\psi(y, \vec{x}, \vec{X}) \equiv \exists \vec{P} \forall \vec{z} \leq \vec{t} \varphi(y, \vec{z}, \vec{P}, \vec{x}, \vec{X})$$

be a Σ_1^1**-Krom** formula. We need to show that the comprehension axiom for ψ is provable in **VNL**:

(289) $$\mathbf{VNL} \vdash \exists Y \leq b \forall y < b,\ Y(y) \leftrightarrow \exists \vec{P} \forall \vec{z} \leq \vec{t} \varphi(y, \vec{z}, \vec{P}, \vec{x}, \vec{X})$$

The idea is to formalize in **VNL** the ONLY IF direction in the proof of Theorem 9.134. For a fixed set of values for $\vec{x}, \vec{X}$, for each value of $y < b$ we consider the propositional formula (285):

(290) $$\psi_y \equiv \bigwedge_{z_1=0}^{v_1} \cdots \bigwedge_{z_m=0}^{v_m} A_{z_1,\ldots,z_m}$$

As in the proof of Theorem 9.134, $\neg\exists \vec{P} \forall \vec{z} \leq \vec{t} \varphi(y, \vec{z}, \vec{P}, \vec{x}, \vec{X})$ holds iff ψ_y is unsatisfiable iff ψ_y contains a set of clauses of the form (286):

(291)
$$\ell_0 \supset \ell_1, \ell_1 \supset \ell_2,\ \ldots,\ \ell_k \supset \neg\ell_0,\ \neg\ell_0 \supset \ell'_1,\ \ell'_1 \supset \ell'_2,\ \ldots,\ \ell'_n \supset \ell_0$$

Here we need to formalize this argument in **VNL**. So let G_y be the graph with vertices labeled by the literals of ψ_y, and there is an edge from ℓ_1 to ℓ_2 in G_y iff the clause

$$\ell_1 \supset \ell_2$$

is in ψ_y. Note that if the edge (ℓ_1, ℓ_2) is in G_y then so is the edge $(\neg\ell_2, \neg\ell_1)$. Also, the encoding of G_y by a pair $(a(y), E^{[y]})$ can be described by a Σ_0^B formula and we omit the details here. It is important that we can check simultaneously in each G_y whether there is a path from any vertex u to a vertex v (by Exercise 9.125).

The fact (289) follows from the next lemma:

LEMMA 9.140. **VNL** *proves that* $\neg\exists \vec{P} \forall \vec{z} \leq \vec{t} \varphi(y, \vec{z}, \vec{P}, \vec{x}, \vec{X})$ *is equivalent to the statement that* G_y *contains a path from* p *to* $\neg p$ *and a path from* $\neg p$ *to* p *for some propositional variable* p *of* ψ_y.

It remains to prove the lemma. Argue in **VNL**: the ($\Longleftarrow$) direction is straightforward, so consider the ($\Longrightarrow$) direction. We prove the contrapositive. Suppose that G_y does not contain simultaneously a path from p to $\neg p$ and a path from $\neg p$ to p for any propositional variable p, we will define a set of values for the string variables $\vec{P}$ that satisfies

$$\forall \vec{z} \leq \vec{t} \varphi(y, \vec{z}, \vec{P}, \vec{x}, \vec{X})$$

It is easy to define such a set in polytime, however, here we need to define it in **NL**. We will give an **NL** algorithm that assigns values for the propositional variables that satisfies ψ_y. It will be clear that the values for $\vec{P}$ defined accordingly satisfies the requirement. Furthermore, it is straightforward that these arguments can be formalized in **VNL**.

The algorithm works as follows. First, identify all literals ℓ such that there is a path from $\neg\ell$ to ℓ in G_y. These are the literals that are forced to to be true; so assign $\top$ to these literals and all other literals that are reachable from them. Note that by the hypothesis, no variable gets conflict truth value.

Now suppose that

$$p_1, p_2, \ldots, p_n$$

are the remaining variables. Let G'_y be the induced subgraph of G_y on the literals

$$p_1, \neg p_1, p_2, \neg p_2, \ldots, p_n, \neg p_n$$

For each literal ℓ let

$$C(\ell) = \{\ell' \ : \ \text{there is a path from } \ell \text{ to } \ell' \text{ or from } \ell' \text{ to } \ell \text{ in } G'_y\}$$

Note that for any literal ℓ', at most one of $\ell', \neg\ell'$ is in $C(\ell)$. Also,

$$\ell \in C(\ell') \qquad \text{iff} \qquad \ell' \in C(\ell)$$

Let (see Figure 12 below)

$$C^+(\ell) = \{\ell\} \cup \{\ell' \ : \ \text{there is a path from } \ell \text{ to } \ell' \text{ in } G'_y\}$$
$$C^-(\ell) = C(\ell) - C^+(\ell)$$

Notice that if $\ell_1 \notin C(\ell_2)$ (hence $\ell_2 \notin C(\ell_1)$), then

$$C^+(\ell_1) \cap C^-(\ell_2) = C^-(\ell_1) \cap C^+(\ell_2) = \varnothing \tag{292}$$

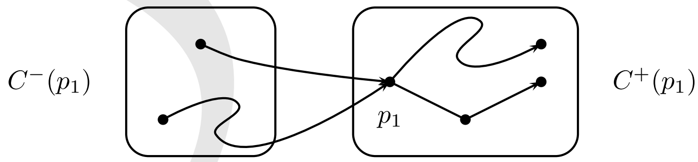

FIGURE 12. $C(p_1)$ and $C^-(p_1), C^+(p_1)$

The idea is to select indices $i_1 \le i_2 \le \cdots \le i_n \le n$ (with repetition) such that

(293) for every variable p, exactly one of $\{p, \neg p\}$ is in $C = \bigcup_j C(p_{i_j})$

Then we assign $\top$ to every literal in

$$C^+ = \bigcup_j C^+(p_{i_j})$$

and $\bot$ to every literal in

$$C^- = \bigcup_j C^-(p_{i_j})$$

The condition (293) ensures that every variable get some truth value.

Notice that

$$\ell \text{ is in } C(\ell') \text{ iff } \neg\ell \text{ is in } C(\neg\ell')$$

The indices $i_1, i_2, \ldots, i_m$ are defined (in parallel) as follows:

$$i_j = \min\{t \ : \ t \ge j \text{ and } p_t, \neg p_t \notin \bigcup_{r \le j} \big(C(p_r) \cup C(\neg p_r)\big)\}$$

Observe that if $i_j < i_k$, then $p_{i_k} \notin C(p_{i_j}) \cup C(\neg p_{i_j})$, so the observation (292) guarantees that our truth assignment is well-defined. Thus, for any i, the truth value of p_i is determined as follows:

- find the smallest j such that p_i or $\neg p_i$ is in $C(p_{i_j})$
- assign p_i the value $\top$ just in case either $p_i \in C^+(p_{i_j})$ or $\neg p_i \in C^-(p_{i_j})$.

To complete the proof of Theorem 9.138 we need to show that the truth assignment above is correct. This is left as an exercise. ⊣

Exercise 9.141. *Complete the argument above, i.e.,*

(a) *show that $p_{i_1}, p_{i_2}, \ldots, p_{i_n}$ satisfy the condition* (293)*, and*
(b) *show that the truth assignment described above satisfies ψ_y.*

The following result is interesting but is independent from the rest of the section:

Theorem 9.142. *For each Σ_0^B formula φ, there is a Σ_1^1-**Krom** formula φ' so that $\mathbf{V}^0 \vdash \varphi \leftrightarrow \varphi'$.*

Proof. Without loss of generality suppose that φ is the formula

$$\exists x_1 < a \forall y_1 < a \exists x_2 < a \forall y_2 < a \ldots \exists x_k < a \forall y_k < a \psi(x_1, y_1, \ldots, x_k, y_k)$$

The truth value of φ depends on the existence of the witnessing values for the x_i. So our Σ_1^1-**Krom** formula φ' will use an existentially quantified string variable S to describe a search algorithm for x_i. Here S encodes a $(2k-1)$-dimension array.

Consider the $\vee$-$\wedge$ tree which results from φ by expanding the bounded quantifiers to finite disjunctions and conjunctions (see Figure 13). Our search is a depth-first search for "true" nodes on the tree. If a node is an $\vee$-node, then each of its children is tried successively until a true one is found, in which case the search ends for that node. If the node is an $\wedge$-node, then all of its children are tested in parallel using universal quantifier.

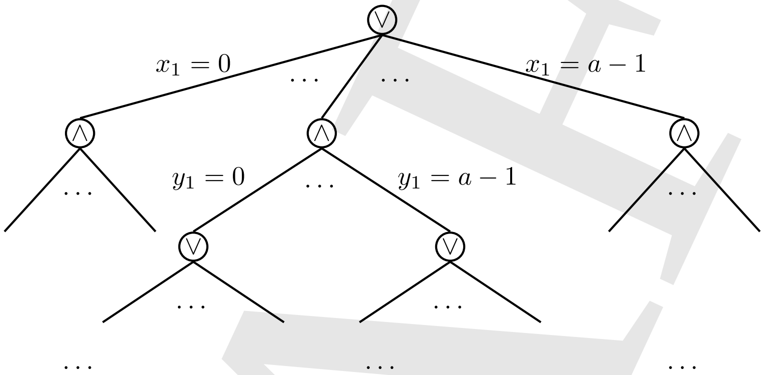

FIGURE 13. The $\vee$-$\wedge$ tree

Every node in the tree is specified by the path from the root to that node. The path to an $\wedge$-node has odd length and has edges with labels

$$x_1 = u_1, y_1 = v_1, x_2 = u_2, y_2 = v_2, \dots, y_{j-1} = v_{j-1}, x_j = u_j$$

We encode such path by

$$\langle u_1, v_1, \dots, v_{j-1}, u_j\rangle$$

Similarly, an $\vee$-node is specified by a tuple of the form

$$\langle x_1, y_1, \dots, y_{j-1}\rangle$$

(Here $1 \le j \le k$ and $0 \le u_i, v_j < a$.) The root is encoded by the empty tuple.

The depth-first search is encoded as follows. The fact that our search visits the $\wedge$-node $\langle x_1, y_1, \dots, y_{j-1}, x_j\rangle$ is indicated by

$$\forall y_j < a \dots \forall y_{k-1} < a\, S(x_1, y_1, \dots, y_{j-1}, x_j, y_j, 0, y_{j+1}, 0, \dots, 0, y_{k-1}, 0)$$

Also, the fact that an $\vee$-node $\langle x_1, y_1, \dots, y_{j-1}\rangle$ is evaluated to true is coded by

$$\neg S(x_1, y_1, \dots, y_{j-1}, a, 0, 0, \dots, 0)$$

The formula ψ' below asserts that the search S

- starts at the leftmost child of the root (294),
- terminates successfully (295),
- backtracks at a leaf (which is always a $\wedge$-node) if ψ is false (296), and

- backtracks to the next sibling of the parent of the current node whenever it encounters a "false" child of an $\wedge$-node (297).

$$\forall \vec{x} < \vec{a} \forall \vec{y} < \vec{a} \forall \overrightarrow{y'} < \vec{a}$$

$$(S(0, y_1, 0, y_2, \ldots, 0, y_{k-1}, 0) \wedge \tag{294}$$

$$\neg S(a, 0, 0, \ldots, 0) \wedge \tag{295}$$

$$((S(x_1, y_1, \ldots, x_k) \wedge \neg\psi(x_1, y_1, \ldots, x_k, y_k)) \supset S(x_1, y_1, \ldots, x_k + 1)) \wedge \tag{296}$$

$$\bigwedge_{i=1}^{k-1} \big(S(x_1, y_1, \ldots, x_i, y_i, a, \vec{0}) \supset S(x_1, y_1, \ldots, x_i + 1, y'_i, 0, y'_{i+1}, \ldots, y'_k, 0)\big) \tag{297}$$

The formula φ' is defined to be $\exists S \psi'$. We will show in $\mathbf{V}^0$ that $\varphi' \leftrightarrow \varphi$, i.e., the $(2k-1)$-dimensional array S in φ' always correctly encodes an algorithm that finds the witnesses $x_1, \ldots, x_k$ for φ.

First we show that $\varphi' \supset \varphi$. Assume the existence of S that satisfies ψ'. We need to show that the root of the $\vee$-$\wedge$ tree (Figure 13) is evaluated to true.

For an $\vee$-node N that is encoded by the path $\langle x_1, y_1, \ldots, x_{i-1}, y_{i-1}\rangle$, let $\rho_i(x_1, y_1, \ldots, x_{i-1}, y_{i-1})$ be the formula expressing the fact that "the algorithm visits the leftmost child of N, and N is evaluated to true":

$$\begin{aligned}\rho_i(x_1, y_1, \ldots, x_{i-1}, y_{i-1}) \equiv\ \ & \forall \overrightarrow{y'} < \vec{a}\, S(x_1, \ldots, y_{i-1}, 0, y'_i, 0, \ldots, 0, y'_{k-1}, 0) \wedge \\ & \neg S(x_1, \ldots, y_{i-1}, a, 0, 0, \ldots, 0)\end{aligned}$$

Then by (294) and (295), $\rho_1()$ is true.

We will show that the algorithm is correct at every level (of the $\vee$-nodes), i.e.,

$$\rho_i(x_1, \ldots, y_{i-1}) \supset \exists x_i \forall y_i \ldots \exists x_k \forall y_k\, \psi(x_1, \ldots, y_k) \tag{298}$$

for $i = k, \ldots, 1$. When $i = 1$, this gives us φ.

EXERCISE 9.143. *Prove* (298) *by downward induction on* i. *(Hint: Use the fact that X-***MIN** *is provable in* $\mathbf{V}^1$-**KROM**. *This fact follows from Lemma 9.136.)*

Next we show that $\varphi \supset \varphi'$. Suppose that φ is true. We need to show the existence of a set S that satisfies φ'. One such S encodes the computation of the lexicographically smallest witnesses $x_1, \ldots, x_k$.

Define the functions $f_1()$, $f_2(y_1)$, $\ldots$, $f_k(y_1, \ldots, y_{k-1})$ as follows:

$$\begin{aligned} f_1() =\ & \min\{x_1 : \forall y_1 \ldots \exists x_k \forall y_k\, \psi(x_1, y_1 \ldots, x_k, y_k)\} \\ f_2(y_1) =\ & \min\{x_2 : \forall y_2 \ldots \exists x_k \forall y_k\, \psi(f_1(), y_1, x_2, \ldots, x_k, y_k)\} \\ & \ldots \\ f_k(y_1, \ldots, y_{k-1}) =\ & \min\{x_k : \forall y_k\, \psi(f_1(), y_1, f_2(y_1), \ldots, y_{k-1}, x_k, y_k)\}\end{aligned}$$

These functions are in $\mathbf{FAC}^0$ and hence are definable in $\mathbf{V}^1$-**KROM** because $\mathbf{V}^1$-**KROM** extends $\mathbf{V}^0$ (by Lemma 9.136 below). Now we define the set S. For a tuple $\langle x_1, y_1, \ldots, x_k, y_k\rangle$, let

$$i = \min\{j \leq k : x_j \neq f_j(y_1, \ldots, y_{j-1})\}$$

Then

$$S(x_1, y_1, \ldots, x_k, y_k) = \begin{cases} \top & \text{if } x_i \leq f_i(y_1, \ldots, y_{i-1}) \\ \bot & \text{otherwise} \end{cases}$$

The remaining of the proof is left as an exercise. ⊣

EXERCISE 9.144. *Show that the set S defined above satisfies* (294) — (297).

9F.3. The theories VL, $\widehat{\mathbf{VL}}$ and $\overline{\mathbf{VL}}$. Given a directed graph G whose vertices have outdegree at most one, and two vertices s, t of G, the PATH problem is to decide whether there is a path in G from s to t. (So PATH is the restriction of the st-CONN problem where the graphs have outdegree at most one.) Here we develop the theories **VL**, $\widehat{\mathbf{VL}}$ and $\overline{\mathbf{VL}}$ based on the fact that PATH is a complete problem for **L**.

Below, Exercise 9.153 shows that our definition of **VL** is equivalent to an earlier definition given in [86]. Then in Theorem 9.154 we show that $\mathbf{VNC}^1$ is a subtheory of **VL**.

First we formalize the PATH problem. As before, our "source" s is always the vertex 0. Recall the function $seq(v, P) = (P)^v$ that encodes a sequence of numbers by P (Definition 5.56 on page 111). Let $\delta_{PATH}(a, E, P)$ be the Σ_0^B equivalence of

$$(P)^0 = 0 \wedge \forall v < a\, E((P)^v, (P)^{v+1}) \tag{299}$$

Here P codes a path in G starting at 0: $(P)^v$ is the v-th vertex on the path.

The relation R_{PATH} below is $\mathbf{AC}^0$-many-one complete for **L**. Here the designated "target" vertex t is one.

THEOREM 9.145. *Let*

$$R_{PATH}(a, E) \equiv (\forall x < a \exists! y < a E(x, y)) \wedge \exists P(\delta_{PATH}(a, E, P) \wedge (P)^a = 1)$$

The relation R_{PATH} is in **L**, *and for every relation $R(\vec{x}, \vec{X})$ in* **L** *there are $\mathbf{AC}^0$ functions $a_0(\vec{x}, \vec{X})$, $E_0(\vec{x}, \vec{X})$ so that*

$$R(\vec{x}, \vec{X}) \leftrightarrow R_{PATH}(a_0(\vec{x}, \vec{X}), E_0(\vec{x}, \vec{X}))$$

PROOF SKETCH. The fact that R_{PATH} is in **L** is straightforward. The second fact can be proved as in Proposition 9.121 except for the vertices in the graph G now have outdegree at most one because the Turing machine is deterministic. ⊣

Definition 9.146 (**VL**). *Let PATH be the axiom*

$$\forall x < a \exists ! y < a E(x,y) \supset \exists P\, \delta_{PATH}(a, E, P) \tag{300}$$

VL *is the theory over* $\mathcal{L}_A^2$ *that is axiomatized by PATH and the axioms of* $\mathbf{V}^0$.

Now consider the function *Path* with the following quantifier-free defining axiom over $\mathcal{L}_{\mathbf{FAC}^0}$:

$$\begin{aligned} \text{(301)}\quad & \mathit{Path}(a, E) = P \leftrightarrow \big(|P| \le \langle a, a\rangle \wedge \\ & \qquad \big((y_1 \ne y_2 \wedge E(x, y_1) \wedge E(x, y_2) \wedge |P| = 0) \vee \\ & \qquad\qquad ((P)^0 = 0 \wedge v < a \supset E((P)^v, (P)^{v+1}))\big)\big) \end{aligned}$$

Proposition 9.147. *The function Path is* $\mathbf{AC}^0$*-many-one complete for* **FL**.

The following lemma is straightforward:

Lemma 9.148. *The function Path is* Σ_1^B*-definable in* **VL**.

Lemma 9.149. *The function* $\mathit{Path}^\star$ *is provably total in* **VL**, *and* (165) *(displayed below) is provable in* $\mathbf{VL}(\mathit{Row}, \mathit{Path}, \mathit{Path}^\star)$.

$$\forall i < b,\ \mathit{Path}^\star(b, \vec{Z}, \vec{X})^{[i]} = \mathit{Path}((Z_1)^i, \dots, (Z_k)^i, X_1^{[i]}, \dots, X_n^{[i]})$$

Proof. Given b graphs $G_0, \dots, G_{b-1}$ whose outdegree is exactly 1 and who are encoded by $(a, E^{[u]})$ (where $0 \le u < b$), we need to construct simultaneously in **VL** the paths $P^{[0]}, P^{[1]}, \dots, P^{[b-1]}$ so that for $0 \le u < b$, $P^{[u]}$ satisfies $\delta_{PATH}(a, E^{[u]}, P^{[u]})$.

Formally we need to prove that the following is a theorem of **VL**:

$$\begin{aligned} \text{(302)}\quad & \forall u < b \forall x < a \exists ! y < a\, E^{[u]}(x, y) \supset \\ & \qquad \exists P \forall u < b,\ (P^{[u]})^0 = 0 \wedge \forall v < a\, E^{[u]}((P^{[u]})^v, (P^{[u]})^{v+1}) \end{aligned}$$

We will construct a graph G' encoded by (a', E') that contains a path $Q = \mathit{Path}(a', E')$ from which we can define the paths $P^{[0]}, \dots, P^{[b-1]}$. In fact, we will define G' so that Q is just the concatenation of $P^{[u]}$, $0 \le u < b$. More precisely, the nodes of G' are encoded by triples $\langle u, v, x\rangle$ in such a way that if $P^{[u]}$ encodes the path $(0, x_1, \dots, x_a)$, then in Q there is the sub-path of the form

$$\langle u, 0, 0\rangle, \langle u, 1, x_1\rangle, \dots, \langle u, a, x_a\rangle$$

In other words, we will have

$$(P^{[u]})^v = x$$

for all nodes $\langle u, v, x\rangle$ on the path Q.

Thus we have the following edges in G' (for $0 \le u < b$):

$$(\langle u,v,x\rangle, \langle u,v+1,y\rangle) \in E' \qquad \text{for } 0 \le v,x,y < a \text{ and } (x,y) \in E^{[u]}$$
$$(\langle u,a,x\rangle, \langle u+1,0,0\rangle) \in E' \qquad \text{for } x < a$$

Let $a' = \langle b,a,a\rangle$, then the graph encoded by (a', E') satisfies the hypothesis of *PATH*. Let Q be the path for this graph. We can prove by induction that the $(u(a+1)+v)$-th node on the path is of the form $\langle u,v,x\rangle$:

$$(Q)^{u(a+1)+v} = \langle u,v,x\rangle \qquad \text{for some } x, 0 \le x < a$$

Define P so that

$$(P^{[u]})^v = x \text{ iff } (Q)^{u(a+1)+v} = \langle u,v,x\rangle$$

It is straightforward to show that each $P^{[u]}$ satisfies $\delta_{PATH}(a, E^{[u]})$. $\dashv$

Now we define the universal theories $\widehat{\mathbf{VL}}$ and $\overline{\mathbf{VL}}$.

Definition 9.150 ($\widehat{\mathbf{VL}}$). *The theory $\widehat{\mathbf{VL}}$ has vocabulary $\mathcal{L}_{\widehat{\mathbf{VL}}} = \mathcal{L}_{\mathbf{FAC}^0} \cup \{Path\}$. The axioms of $\widehat{\mathbf{VL}}$ consist of the axioms of $\mathbf{V}^0$ and* (301).

Definition 9.151 ($\overline{\mathbf{VL}}$). *The language $\mathcal{L}_{\mathbf{FL}}$ is the smallest set that contains $\mathcal{L}_{\widehat{\mathbf{VL}}}$ such that for every $\mathcal{L}_A^2$-term t and every quantifier-free $\mathcal{L}_{\mathbf{FL}}$formula φ there is a function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{FL}}$ with defining axiom* (85)*:*

$$F_{\varphi(z),t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X}) \tag{303}$$

The theory $\overline{\mathbf{VL}}$ has vocabulary $\mathcal{L}_{\mathbf{FL}}$ and axioms those of $\overline{\mathbf{V}}^0$ and (303) *for each function $F_{\varphi(z),t}$.*

We have a corollaries of the results from Section 9B the Definability Theorems for $\mathbf{VL}$, $\widehat{\mathbf{VL}}$ and $\overline{\mathbf{VL}}$:

Corollary 9.152. *Here either $\mathcal{L}$ is $\mathcal{L}_{\widehat{\mathbf{VL}}}$ and $\mathcal{T}$ is $\widehat{\mathbf{VL}}$, or $\mathcal{L}$ is $\mathcal{L}_{\mathbf{FL}}$ and $\mathcal{T}$ is $\overline{\mathbf{VL}}$.*

(a) *A function is in $\mathbf{FL}$ iff it is represented by a term in $\mathcal{L}_{\widehat{\mathbf{VL}}}$. A string function is in $\mathbf{FL}$ iff it is represented by a string function in $\mathcal{L}_{\mathbf{FL}}$. A relation is in $\mathbf{L}$ iff it is represented by an open (or a Σ_0^B) formula of $\mathcal{L}$.*
(b) *Every $\Sigma_0^B(\mathcal{L})$ formula is equivalent in $\mathcal{T}$ to a $\Sigma_1^B(\mathcal{L}_A^2)$ formula.*
(c) *$\mathcal{T} \vdash \Sigma_0^B(\mathcal{L})$-$\mathbf{COMP}$.*
(d) *$\overline{\mathbf{VL}}$ is a conservative extension of $\widehat{\mathbf{VL}}$ which is in turn a conservative extension of $\mathbf{VL}$.*
(e) *The Σ_1^B-definable functions of $\mathbf{VL}$ (or $\widehat{\mathbf{VL}}$, $\overline{\mathbf{VL}}$) are precisely functions in $\mathbf{FL}$.*
(f) *The Δ_1^B-definable relations of $\mathbf{VL}$ (or $\widehat{\mathbf{VL}}$, $\overline{\mathbf{VL}}$) are precisely relations in $\mathbf{L}$.*

In [86] Zambella introduced the theory Σ_0^B-**Rec** and showed that it characterizes **L**. It can be shown to be equivalent to **VL**. Here Σ_0^B-**Rec** is defined using the following axiom scheme:

(304)
$$\forall w < b \forall x < a \exists y < a\, \varphi(w, x, y) \supset \exists Z, \forall w < b\, \varphi(w, (Z)^w, (Z)^{w+1}).$$

for all Σ_0^B formulas φ not involving Z.

Note that our axiom (300) is an instance of (304). So to prove the above equivalence, the main task is to show that (304) is provable in our theory. This is left as an exercise.

Exercise 9.153. *Show that* **VL** *proves the axiom scheme* (304).

Finally we prove:

Theorem 9.154. $\mathbf{VNC}^1 \subseteq \mathbf{VL}$.

Proof. Since $\widehat{\mathbf{VL}}$ is conservative over **VL**, it suffices to show that $\mathbf{VNC}^1 \subseteq \widehat{\mathbf{VL}}$. Recall the formula *MFV* (Definition 9.92. We need to show that
$$\widehat{\mathbf{VL}} \vdash \mathit{MFV}$$
The idea is to formalize in $\widehat{\mathbf{VL}}$ a logspace algorithm that evaluates a Boolean sentence. The algorithm that we consider here makes a depth-first-search traversal on the tree structure of the sentence, skipping a whole subtree whenever possible (e.g., if A is true then $A \vee B$ is true, so we do not have to examine B).

Thus consider a balanced sentence specified by (a, G, I) as in Section 9E.1. For each $1 \leq x < a$ we construct a graph encoded by $(a_x, E^{[x]})$ so that the bit $\mathit{Fval}(a, G, I)(x)$ can be obtained from $\mathit{Path}(a_x, E^{[x]})$. Then by Lemma 9.149 all bits of $\mathit{Fval}(a, G, I)$ can be obtained simultaneously, and we are done.

We show how to obtain the bit $\mathit{Fval}(a, G, I)(1)$. Other bits can be obtained similarly. The graph $(a_1, E^{[1]})$ describes a depth-first search traversal in the circuit (a, G) to compute the output of the root. Each vertex is a (potential) state of the traversal. There is a starting node (vertex 0), and each other vertex is numbered by
$$\langle x, \mathsf{d}, 0\rangle \text{ or } \langle x, \mathsf{u}, v\rangle, \qquad \text{where } 1 \leq x < 2a, 0 \leq v \leq 1$$
(here $\mathsf{d} = 1, \mathsf{u} = 2$ indicate the direction of the traversal). A vertex $\langle x, \mathsf{d}, 0\rangle$ corresponds to the state when the depth-first traversal visits the gate numbered x for the first time (so in general it will go "down"). Similarly, a state $\langle x, \mathsf{u}, v\rangle$ is when the search visits gate x the second time (thus the direction is "up"); by this time the truth value of the gate is known, and v carries this truth value.

The edges of this graph represent the transition between the states of the search. The search starts at the root, thus we have the following edge:
$$(0, \langle 1, \mathsf{d}, 0\rangle)$$

When the search visits a gate x for the first time, it will travel down along the left-most branch from x:

$$(\langle x, \mathsf{d}, 0\rangle, \langle 2x, \mathsf{d}, 0\rangle) \text{ for } 1 \le x < a$$

And here are the transitions when it reaches the input gates:

$$\begin{array}{ll} (\langle x+a, \mathsf{d}, 0\rangle, \langle x+a, \mathsf{u}, 0\rangle) & \text{if } \neg I(x),\ 0 \le x < a \\ (\langle x+a, \mathsf{d}, 0\rangle, \langle x+a, \mathsf{u}, 1\rangle) & \text{if } I(x),\ 0 \le x < a \end{array}$$

For an $\vee$-gate x (i.e., if $\neg G(x)$, where $1 \le x < a$) notice that the search in the subtree rooted at x can be completed when (i) either child of x outputs $\top$, or (ii) the right child of x outputs $\bot$. Furthermore, if the left child of x outputs $\bot$, then the search continue at the right child. We have the following transitions:

$$\begin{array}{rl} \text{either child outputs } \top\text{:} & (\langle 2x, \mathsf{u}, 1\rangle, \langle x, \mathsf{u}, 1\rangle) \text{ and } (\langle 2x+1, \mathsf{u}, 1\rangle, \langle x, \mathsf{u}, 1\rangle) \\ \text{the right child outputs } \bot\text{:} & (\langle 2x+1, \mathsf{u}, 0\rangle, \langle x, \mathsf{u}, 0\rangle) \\ \text{the left child outputs } \bot\text{:} & (\langle 2x, \mathsf{u}, 0\rangle, \langle 2x+1, \mathsf{d}, 0\rangle) \end{array}$$

Exercise 9.155. *Give the transitions for an $\wedge$-gate.*

Notice that the graph described so far have outdegree *at most* 1. To make the outdegree *exactly* 1 we can create an extra node and connect all vertices with outdegree 0 to it. Let the resulting graph be encoded by $(a_1, E^{[1]})$. Note that our traversal does not visit all gates of the circuit (a, G, I). But if it does visit a gate, then the gate will be evaluated. In particular, the output of gate 1 (i.e. $Fval(a, G, I)(1)$) is

$$\exists v\, (Path(a_1, E^{[1]}))^v = \langle 1, \mathsf{d}, 0\rangle \wedge \exists w (Path(a_1, E^{[1]}))^w = \langle 1, \mathsf{u}, 1\rangle\big)$$

Similarly we construct a graph $(a_x, E^{[x]})$ to evaluate each node x of the circuit (a, G, I). By $\Sigma_0^B(\widehat{\mathbf{VL}})$-**COMP** there is a string Y so that for $x < 2a$,

$$Y(x) \leftrightarrow \big(\exists v\, (Path(a_x, E^{[x]}))^v = \langle x, \mathsf{d}, 0\rangle \wedge \exists w (Path(a_x, E^{[x]}))^w = \langle x, \mathsf{u}, 1\rangle\big)$$

It remains to prove (in $\widehat{\mathbf{VL}}$) that $\delta_{MFV}(a, G, I, Y)$ holds. This is left as an exercise. ⊣

Exercise 9.156. *Complete the proof above by showing that $\widehat{\mathbf{VL}}$ proves $\delta_{MFV}(a, G, I, Y)$.*

9F.4. The theory VLV. Recall the notion of polynomial-bounded bounded number recursion (pBNR) from Section 9C.3. We develop the universal theory **VLV** using the fact that the function class **FL** can be characterized using pBNR. Thus **VLV** has the same style as **VPV**. Its vocabulary contains symbols for all functions in **FL**. Here their defining axioms are given using the above fact. First we state the characterization of **FL**.

THEOREM 9.157 (Lind). *A function is in* **FL** *iff it can be obtained by* $\mathbf{AC}^0$*-reduction and pBNR iff it can be obtained from* $\mathbf{FAC}^0$ *by finitely many applications of composition, string comprehension, and pBNR.*

PROOF SKETCH. First, it is easy to see that $\mathbf{FAC}^0 \subseteq \mathbf{FL}$ and that **FL** is closed under composition, string comprehension and pBNR. By Theorem 9.7 it remains to show that functions in **FL** can be obtained from $\mathbf{FAC}^0$ by $\mathbf{AC}^0$-reduction and pBNR.

Suppose that $F(\vec{x}, \vec{X})$ is a function in **FL** and let M be a logspace polytime Turing machine that computes F. As in the proof of Propositions 9.121 and 9.145, the configurations of M (without the input and output tape content) are encoded by numbers $< t(\vec{x}, \vec{X})$ for some number term bounding the running time of M such that 0 and 1 are respectively the initial and (the only) accepting configuration.

Since M is deterministic, there is an $\mathbf{AC}^0$ function $next_{\mathsf{M}}(z, \vec{x}, \vec{X})$ such that for $z < t(\vec{x}, \vec{X})$, $next_{\mathsf{M}}(z, \vec{x}, \vec{X})$ is the next configuration of z if z is a non-final configuration of M, otherwise:

$$next_{\mathsf{M}}(z, \vec{x}, \vec{X}) = \begin{cases} 0 & \text{if } z \text{ does not code a configuration of } \mathsf{M} \\ z & \text{if } z \text{ is a final configuration of } \mathsf{M}\text{, e.g., } 1 \end{cases}$$

Let $conf_{\mathsf{M}}(y, \vec{x}, \vec{X})$ denote the configuration of M at time y. Then we have

$$conf_{\mathsf{M}}(0, \vec{x}, \vec{X}) = 0$$
$$conf_{\mathsf{M}}(y+1, \vec{x}, \vec{X}) = nextM(conf_{\mathsf{M}}(y, \vec{x}, \vec{X}), \vec{x}, \vec{X})$$

In other words, $conf_{\mathsf{M}}$ can be obtained from $\mathbf{AC}^0$ functions by pBNR.

Now the bits of the string $F(\vec{x}, \vec{X})$ computed by M can be extracted from the numbers

$$conf_{\mathsf{M}}(0, \vec{x}, \vec{X}), conf_{\mathsf{M}}(1, \vec{x}, \vec{X}), \ldots, conf_{\mathsf{M}}(t(\vec{x}, \vec{X}), \vec{x}, \vec{X})$$

First we need to determine the times at which M writes to its output tape. This can be done using pBNR as well.

EXERCISE 9.158. *Define using pBNR from* $conf_{\mathsf{M}}(y, \vec{x}, \vec{X})$ *the function*

$$next\_write_{\mathsf{M}}(y, \vec{x}, \vec{X})$$

which is the first time $y' > y$ *such that* M *writes to its output tape at time* y'. *Use this to define the function* $write_{\mathsf{M}}(y, \vec{x}, \vec{X})$ *which is the time at which* M *performs the* y*-th write.*

The bits $F(\vec{x}, \vec{X})(y)$ can be extracted from $conf_{\mathsf{M}}(write_{\mathsf{M}}(y, \vec{x}, \vec{X}), \vec{x}, \vec{X})$ by some $\mathbf{AC}^0$ functions. Consequently, F can be obtained by $\mathbf{AC}^0$-reduction and pBNR. $\dashv$

DEFINITION 9.159 (**VLV**). *The language* $\mathcal{L}_{\mathbf{VLV}}$ *is the smallest set that contains* $\mathcal{L}_{\mathbf{FAC}^0}$ *such that for every* $\mathcal{L}^2_A$*-term* t*, quantifier-free* $\mathcal{L}_{\mathbf{VLV}}$*-formula* φ *and number functions* g, h *in* $\mathcal{L}_{\mathbf{VLV}}$ *there are:*

- *a string function $F_{\varphi(z),t}$ in $\mathcal{L}_{\mathbf{VLV}}$ with defining axiom* (85)*:*

$$F_{\varphi(z),t}(\vec{x},\vec{X})(z) \leftrightarrow z < t(\vec{x},\vec{X}) \wedge \varphi(z,\vec{x},\vec{X})$$

- *a number function $f_{t,g,h}$ with defining axiom*

$$(305)\quad (g < t \supset f_{g,h}(0) = g) \wedge (g \geq t \supset f_{g,h}(0) = 0) \wedge \\ (h(y, f_{g,h}(y)) < t \supset f_{g,h}(y+1) = h(y, f_{g,h}(y))) \wedge \\ (h(y, f_{g,h}(y) \geq t \supset f_{g,h}(y+1) = 0)$$

VLV *is the theory with vocabulary $\mathcal{L}_{\mathbf{VLV}}$ and is axiomatized by the axioms of $\overline{\mathbf{V}}^0$ and* (85) *for every function $F_{\varphi(z),t}$,* (305) *for every function $f_{t,g,h}$.*

The next corollary follows from Theorem 9.157.

COROLLARY 9.160. *A function is in* **FL** *iff it is represented by a term in $\mathcal{L}_{\mathbf{VLV}}$. A relation is in* **L** *iff it is represented by an open (or a Σ_0^B) formula of $\mathcal{L}_{\mathbf{VLV}}$.*

The following facts can be proved as in Section 9C.4 and we leave the proofs as exercises.

EXERCISE 9.161. (a) *Show that* **VLV** *proves the axiom scheme*

$$\Sigma_0^B(\mathcal{L}_{\mathbf{VLV}})\textbf{-COMP}, \Sigma_0^B(\mathcal{L}_{\mathbf{VLV}})\textbf{-IND}, \Sigma_0^B(\mathcal{L}_{\mathbf{VLV}})\textbf{-MIN}$$

(b) *Show that every $\Sigma_0^B(\mathcal{L}_{\mathbf{VLV}})$ formula is equivalent over* **VLV** *to a $\Sigma_1^B(\mathcal{L}_A^2)$ formula.*

EXERCISE 9.162. *Show that*

(a) *A function is in* **FL** *iff it is Σ_1^B-definable in* **VLV***.*
(b) *A relation is in* **L** *iff it is Δ_1^B-definable in* **VLV***.*

Finally, the relationship between **VL** and **VLV** is also left as an exercise.

EXERCISE 9.163. **VLV** *is a universal conservative extension of* **VL***.*

9G. Open Problems

9G.1. Proving Cayley–Hamilton Theorem in $\mathbf{VNC}^2$? START

9G.2. VSL and VSL $\stackrel{?}{=}$ VL. The class **SL** consists of languages that are accepted by a *symmetric* nondeterministic Turing machines working in logspace. A nondeterministic Turing machine M is said to be symmetric iff for any two configurations c_1, c_2 of M:

if c_2 is a next configuration of c_1, then c_1 is a next configuration of c_2.

It can be shown that the *st*-connectivity problem for undirected graph is $\mathbf{AC}^0$-many-one complete for **SL**.

A deterministic Turing machine can be seen as a symmetric nondeterministic Turing machine, so

$$\mathbf{L} \subseteq \mathbf{SL}$$

Recent breakthrough by Reingold [76] shows that indeed

$$\mathbf{L} = \mathbf{SL}$$

Before this was shown, the fact that $\mathbf{SL} = co\text{-}SL$ is established in [66].

The Distance Problem for undirected graph (UDP) is to decide, given a undirected graph G and two of its vertices s, t and a positive integer d, whether the distance between s and t is exactly d. It turns out that UDP is complete for **NL**, so the function *Conn* (Section 9F.1) restricted to undirected graphs is complete for **NL**, hence we cannot use it to define a theory for **VSL**.

Here we define **VSL** as follows. Recall the formula $\delta_{PATH}(a, E, P)$ from (299) (on page 326) which asserts that P encodes a path starting at the vertex 0 in the graph specified by (a, E). Let $\delta_{UCONN}(a, E, C, P)$ be the formula given below that states that $C(u)$ holds iff u is in the transitive closure of vertex 0, in that case $P^{[u]}$ encodes a path from 0 to u.

$$\delta_{UCONN}(a, E, C, P) \equiv C(0) \wedge \forall u < a \forall v < a\big((C(u) \wedge E(u, v)) \supset C(v)\big) \wedge$$
$$\forall u < a\big(C(u) \supset (\delta_{PATH}(a, E, P^{[u]}) \wedge P^{[u]}(a) = u)\big)$$

Definition 9.164. **VSL** *is the theory over* $\mathcal{L}_A^2$ *that is axiomatized by the axioms of* $\mathbf{V}^0$ *together with the axiom UCONN:*

$$\forall a \forall E \exists C \exists P\ \delta_{UCONN}(a, E, C, P)$$

The fact that the functions Σ_1^B-definable of **VSL** are exactly functions in **FSL** can be proved using the fact that the relation R_{UCONN} defined below is complete for **SL**. Let R_{UCONN} be the following relation

$$R_{UCONN}(a, E) \leftrightarrow \exists C \leq a \exists P \leq \langle a, a, a\rangle\big(\delta_{UCONN}(a, E, C, P) \wedge C(1)\big)$$

Exercise 9.165. *Show that the relation* R_{UCONN} *above is complete for* **SL**.

Exercise 9.166. *Develop universal conservative extensions* $\overline{\mathbf{VSL}}$ *and* $\widehat{\mathbf{VSL}}$ *of* **VSL** *(in the style of* $\overline{\mathbf{VC}}$ *and* $\widehat{\mathbf{VC}}$*) and show that their* Σ_1^B*-definable functions are precisely the functions in* **FSL**.

Despite the fact [76] that $\mathbf{SL} = \mathbf{L}$, it is an open question whether the corresponding theories are the same.

Open Problem 9.167. *Is* $\mathbf{VSL} = \mathbf{VL}$*?*

9G.3. Defining $\lfloor X/Y \rfloor$ in $\mathbf{VTC}^0$. The string division function $\lfloor X/Y \rfloor$ (or also $X \div Y$) is defined so that

$$\lfloor X/Y \rfloor \times Y \leq X < S(\lfloor X/Y \rfloor) \times Y \tag{306}$$

where S is the string successor function (Example 5.42). Exercise 6.12 shows that $\lfloor X/Y \rfloor$ is Σ_1^B-definable in $\mathbf{V}^1$ by formalizing in $\mathbf{V}^1$ a polytime algorithm that computes $\lfloor X/Y \rfloor$.

A breakthrough result by Hesse et. al. [42] shows that this function is computable in $\mathbf{TC}^0$. However, it has been an open problem whether this algorithm can be formalized and proved correct in the theory $\mathbf{VTC}^0$.

OPEN PROBLEM 9.168. *Is the function $\lfloor X/Y \rfloor$ with defining axiom* (306) Σ_1^B*-definable in* $\overline{\mathbf{VTC}}^0$*?*

9H. Notes

The string comprehension operation can be seen as a two-sorted version of the *concatenation recursion on notation* (CRN) operation for single-sorted classes [25].

The families $\mathbf{VC}$ and $\overline{\mathbf{VC}}$ are from [64]. The theories $\widehat{\mathbf{VC}}$ are new.

The number recursion operations (in Theorems 9.38, 9.65, 9.84, 9.108 and 9.157) are from [63] and are based on previous work of Lind [60] (for $\mathbf{FL}$) and Clote and Takeuti's [27] (for $\mathbf{FAC}^0(2)$, $\mathbf{FAC}^0(6)$ and $\mathbf{FNC}^1$). The characterizations in [27] go back to [69] (for $\mathbf{FAC}^0(2)$ and $\mathbf{FAC}^0(6)$) and [6] (for $\mathbf{FNC}^1$). The proof of the Theorems 9.84 and 9.108 can be found in [63].

Various problems computable in $\mathbf{TC}^0$ are discussed in [23, 42].

The descriptive complexity characterizations of $\mathbf{TC}^0$, $\mathbf{AC}^0(m)$ are from [7], and Grädel's characterization of $\mathbf{NL}$ by second-order logic is in [38].

Section 9C.5 (proving the Pigeonhole Principle in $\mathbf{VTC}^0$) formalizes a "folklore" fact that the PHP can be proved using counting, which goes back to Buss's proof of the PHP in the **Frege** proof system [13]. Section 9C.6 (defining $X \times Y$ in $\mathbf{VTC}^0$) is based on [11, 23].

The proof of the Discrete Jordan Curve Theorem (Section 9D.4) is from [65] which contains also the more complicated proof of the sequence version of JCT in $\mathbf{V}^0$.

The theory $\mathbf{VNC}^1$ is first defined in [33] and is based on Arai's theory **AID** [5]. The current axiomatization is from [63]. Both $\mathbf{VNC}^1$ and **AID** are based on Buss's Theorem that the Boolean Formula Value Problem is complete for $\mathbf{NC}^1$ [14]. The fact that $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$ (Section 9E.3) is based on [13]. The theory $\mathbf{VNC}^1\mathbf{V}$ is called **VALV** in [63]. It is developed based on Barrington's Theorem which is from [6]. A proof of Theorem 9.114 can be found in [63].

The theory **VNL** is from [64] and $\mathbf{V}^1$-**KROM** is from [32]. The results in Section 9F.2 are from [51]. Immerman-Szelepcsényi Theorem (that **NL** is closed under complement) is from [44] and [80]. The theory **VL** is from [63]. The equivalent theory Σ_0^B-**Rec** is from [86].

Chapter 10

THE REFLECTION PRINCIPLE

An association between $\mathbf{V}^i$ and the proof system $\mathbf{G}_i^\star$ (for $i \geq 1$) is shown in Chapter 7 by the fact that each bounded theorem of the theory $\mathbf{V}^i$ translates into a family of tautologies that have polynomial-size $\mathbf{G}_i^\star$ proofs. Our theories and their associated proof systems are more deeply connected than as shown by just the propositional translation theorems. In this chapter we will present some more connections between the proof systems, their associated theories and the underlying complexity classes.

In general, for each proof system $\mathcal{F}$ we study the principle that asserts that the system is sound, i.e, that formulas that have $\mathcal{F}$-proofs are valid. This is known as the Reflection Principle (RFN) for $\mathcal{F}$. We will show in this chapter that the theories $\mathbf{V}^i$ and $\mathbf{TV}^i$ proves the RNF for their associated theories when the principles are stated for Σ_i^q formulas. Together with the Propositional Translation Theorems, these show that the systems $\mathbf{G}_i^\star$ and $\mathbf{G}_i$ are the strongest systems (for proving Σ_i^q formulas) whose RFN are provable in the theories $\mathbf{V}^i$ and $\mathbf{TV}^i$, respectively.

A connection between a propositional proof system $\mathcal{F}$ and the complexity class $\mathbf{C}$ definable in the theory $\mathcal{T}$ associated with $\mathcal{F}$ will be seen by the fact that the Witnessing Problem for $\mathcal{F}$ is complete for $\mathbf{C}$. Recall Theorem 7.51 which shows that the Witnessing Problem for $\mathbf{G}_1^\star$ (and equivalently for **eFrege**) are solvable by a polytime algorithm. (In fact here we will formalize this algorithm in $\mathbf{V}^1$ in order to show that the RFN of $\mathbf{G}_1^\star$ is provable in $\mathbf{V}^1$ as mentioned above.) The fact that the Witnessing Problem for $\mathbf{G}_1^\star$ is hard for $\mathbf{P}$ can be proved by using the Proposition Translation Theorem for $\mathbf{V}^1$ and the fact that theorems of $\mathbf{V}^1$ can be proved using the RFN for $\mathbf{G}_1^\star$.

We will also present some connections between subtheories of $\mathbf{TV}^0$ and their associated proof systems. Here $\mathbf{VNC}^1$ is associated with the sequent calculus **PK** introduced in Chapter 2 in the same way that $\mathbf{V}^0$ is associated with bounded depth **PK** or $\mathbf{V}^1$ is associated with **eFrege**. The theory $\mathbf{VTC}^0$ is associated with bounded depth **PTK**, the systems that extend bounded depth **PK** by a new kind of connectives corresponding to the counting gates in $\mathbf{TC}^0$ circuits.

The chapter is organized as follows. We start by formalizing propositional proofs in Section 10A. The formalizations are needed for stating the Reflection Principle. They also enable us to state the Propositional Translation Theorems as theorem in our theory $\mathbf{VTC}^0$. In fact we will prove (in Section 10A.3) the Propositional Translation Theorems for $\mathbf{TV}^i$ as a theorem of $\mathbf{VTC}^0$, and restate various theorems from Chapter 7 this way. The RFN and Witnessing Problems for $\mathbf{G}_i^\star$ and $\mathbf{G}_i$ will be discussed in Section 10B. Finally, in Sections 10C and 10D we discuss the Propositional Translation Theorems for the theories $\mathbf{VNC}^1$ and $\mathbf{VTC}^0$.

10A. Formalizing Propositional Translations

Recall (Definition 7.2) that a proof system is defined to be a polytime, surjective function:

$$F : \{0,1\}^* \longrightarrow \mathit{TAUT}$$

It turns out that all proof systems that we have discussed are $\mathbf{TC}^0$ functions. This is because for these systems, to compute $F(X)$ the main task is often to verify whether X is a legitimate proof. The verification in turn consists of recognizing (quantified) propositional formulas, sequents and proofs. The recognition can be done using the counting gates, for example, to check that the parentheses are properly nested in formulas, or that inference rules are properly applied in a proof. Therefore the property of being a legitimate proof (or formula, or sequent) is a $\mathbf{TC}^0$ relation.

Verifying proofs in polytime is often straightforward and therefore omitted. However, to show that it can be done in $\mathbf{TC}^0$ is less straightforward. So in Section 10A.1 below we will carry this out in some detail. Recall (Section 9C.2) that a relation is in $\mathbf{TC}^0$ iff it is Δ_1^B-definable in $\mathbf{VTC}^0$, iff it is represented by an open $\mathcal{L}_{\mathbf{FTC}^0}$ formula, and iff it is represented by a $\Sigma_0^B(\mathcal{L}_{\mathbf{FTC}^0})$ formula (Theorem 9.34).

The propositional translations of $\mathbf{LK}^2$ proofs from Chapter 7 produce uniform propositional proofs. In fact, in Section 10A.2 we will show that these translations are computable by $\mathbf{TC}^0$ functions. In other words, the propositional translation theorems are theorems of $\mathbf{VTC}^0$.

Then in Section 10A.3 we will prove the Propositional Translation Theorem for $\mathbf{TV}^i$. Following the discussion from the previous section, we will show that this is also a theorem of $\mathbf{VTC}^0$.

10A.1. Verifying proofs in $\mathbf{TC}^0$. We will consider proofs of $\mathbf{G}$. Other systems can be handled in similar way or with minor modifications. Recall the definition of $\mathbf{G}$ from Section 7C. First we present a simple encoding of proofs in our two-sorted language $\mathcal{L}_A^2$. The pairing function (Example 5.45) can be used to avoid using deliminators for sequents in a proof or formulas in a cedent.

Let

$$\pi = \mathcal{S}_\ell, \mathcal{S}_{\ell-1}, \dots, \mathcal{S}_1, \mathcal{S}_0$$

be a proof in **G** where $\mathcal{S}_0$ is the end sequent. A simple way of present π in our two-sorted language is to view it as an array whose rows $\pi^{[i]}$ encode the sequents $\mathcal{S}_i$. To simplify our verification $\pi^{[i]}$ will also contains the indices of all parents of $\mathcal{S}_i$. Thus, we let

$$\pi^{[i]} = \langle\langle j, k\rangle, \mathcal{S}_i\rangle \tag{307}$$

where either $j = k = 0$ or $j > i \wedge k = 0$ or $j > i \wedge k > i$. Here j, k are indices of the parents of $\mathcal{S}_i$ (if a parent is not present, the corresponding index is 0). Also, $\langle j, k\rangle$ is the pairing function from Example 5.45, and $\langle x, Y\rangle$ is the pairing function from Definition 8.84.

The number of sequents in π can be easily extracted from π. We will require that every sequent except for the end sequent is used at least once. This can be checked by stating that for all j, where $0 < j \le \ell$, there exists $i < j$ such that $\pi^{[i]}$ has the form

$$\langle\langle j, k\rangle, \mathcal{S}_i\rangle \qquad \text{or} \qquad \langle\langle k, j\rangle, \mathcal{S}_i\rangle$$

(for some k).

Verifying that the rules are properly applied in π will be discussed in the proof of Lemma 10.7 below. Now we briefly discuss the other ingredients of a proof, i.e., sequents and formulas. A sequent $\mathcal{S}$ is encoded as two arrays $\mathcal{S}^{[0]}$ and $\mathcal{S}^{[1]}$ that encode its antecedent and succedent, respectively. For example, $\mathcal{S}^{[0]}$ is $\varnothing$ if the antecedent is empty; otherwise $\mathcal{S}^{[0,0]}$ encodes the first formula of the antecedent, etc.

Next, assume that all propositional variables are either x_k (bound variables) or p_k (free variables), for $k \ge 0$. Using the letters x and p, these can be written respectively as $x11\dots1$ and $p11\dots1$ with k 1's in each string (when $k = 0$ the strings are just x and p, respectively). Thus quantified propositional formulas are written as strings over the alphabet

$$\{\top, \bot, p, x, 1, (,), \wedge, \vee, \neg, \exists, \forall\} \tag{308}$$

(Note that we use unary notation for writing the indices of variables. For a propositional formula of size n there are at most n variables and their indices are at most n. Thus by writing indices in unary notation the size of the formula is within a polynomial of n.)

Formal proofs for most of the results below are straightforward but at the same time tedious. So our arguments will often be informal or sketched. Interested readers are encouraged to carry out the proofs in detail themselves. For example, we will use without proving the fact that the strings over (308) above can be encoded by binary strings by some simple method.

Our $\mathbf{TC}^0$ algorithm for recognizing formulas requires the following notion.

Notation. A string over the vocabulary (308) is called a *pseudo formula* if it has the form

$$(309)\quad \underbrace{\neg\ldots\neg}_{n_1} Q_1 x \underbrace{11\ldots1}_{i_1} \underbrace{\neg\ldots\neg}_{n_2} Q_2 x \underbrace{11\ldots1}_{i_2} \ldots Q_k x \underbrace{11\ldots1}_{i_k} \underbrace{\neg\ldots\neg}_{n_{k+1}} Y$$

where $Q_j \in \{\exists, \forall\}$ for $1 \le j \le k$; $k, n_1, n_2, \ldots, n_{k+1}, i_1, i_2, \ldots, i_k \ge 0$ (i.e., the string preceding Y might be empty); and the substring Y satisfies the condition:

1) either Y is one of the following strings:

$$\top, \qquad \bot, \qquad p\underbrace{11\ldots1}_{\ell}, \qquad x\underbrace{11\ldots1}_{\ell}$$

(for some $\ell \ge 0$), or

2) Y contains the same positive number of left and right parentheses.

Note that any formula is also a pseudo formula.

Notation. For a string $s_0 s_1 \ldots s_n$ over the vocabulary in (308) that is encoded by an $\mathcal{L}_A^2$ string (i.e., set) X, we use $X[i]$ for the symbols s_i, for $0 \le i \le n$. Also, for $i \le j$, let $X[i,j]$ denotes the substring of X that consists of the symbols $X[i], X[i+1], \ldots, X[j]$.

The next two lemmas imply that there is a $\mathbf{TC}^0$ algorithm that accepts precisely proper encodings of formulas. Recall the language $\mathcal{L}_{\mathbf{FTC}^0}$ from Definition 9.33, and recall that a relation is in $\mathbf{TC}^0$ iff it is representable by an open $\mathcal{L}_{\mathbf{FTC}^0}$ (or also $\Sigma_0^B(\mathcal{L}_{\mathbf{FTC}^0})$) formula (Theorem 9.34).

LEMMA 10.1. *Let X be a string of length $n+1$ over the alphabet* (308)*. Then there is an open $\mathcal{L}_{\mathbf{FTC}^0}$ formula $\varphi(i, \ell, r, X)$ that is true iff $0 \le i \le n$ and:*

- *if there is a pseudo formula of the form $X[\ell', r']$ where $\ell' \le i \le r'$, then $X[\ell, r]$ is one such pseudo formula with lexicographically smallest pair $(r - \ell, r)$;*
- *otherwise, $\ell_i = 0$ and $r_i = n$.*

PROOF SKETCH. We will describe informally a uniform family of $\mathbf{TC}^0$ circuits that accept (i, ℓ, r, X) as in the lemma. Then it is straightforward to construct the required open $\mathcal{L}_{\mathbf{FTC}^0}$ formula.

First, note that using the counting gates all pseudo formulas in X can be identified. This is because the strings preceding Y in (309) can be in fact identified by $\mathbf{AC}^0$ circuits, the strings Y as in (1) above can also be identified by $\mathbf{AC}^0$ circuits, and the counting gates can be used to identify the strings Y in (2).

Now we can identify all pseudo formulas of the form $X[\ell, r]$ that contain $X[i]$. Suppose that this list is nonempty, then we can also extract the one with lexicographically smallest pair $(r - \ell, r)$ by $\mathbf{AC}^0$ circuits. Otherwise, if the list is empty, the circuits will accept only $(i, 0, n, X)$. ⊣

Note that in the lemma above, for each i $(0 \leq i \leq n)$ there is exactly one pair (ℓ_i, r_i) such that $\varphi(i, \ell_i, r_i, X)$ holds.

Lemma 10.2. *Let X be a string of length $n+1$ over the alphabet* (308)*, let φ be the open $\mathcal{L}_{\mathbf{FTC}^0}$ formula from Lemma 10.1, and let*

$$\langle \ell_0, r_0 \rangle, \langle \ell_1, r_1 \rangle, \ldots, \langle \ell_n, r_n \rangle$$

be such that $\varphi(i, \ell_i, r_i, X)$ hold for all $0 \leq i \leq n$. Then X properly encodes a formula if and only if all conditions below hold:

1) *X is a pseudo formula,*
2) *if any two substrings $X[\ell_i, r_i]$, $X[\ell_j, r_j]$ overlap, then one is included in the other,*
3) *any two disjoint substrings $X[\ell_i, r_i]$, $X[\ell_j, r_j]$ are separated by at least a connective $\wedge$ or $\vee$,*
4) *if $X[i] = p$ or $X[i] = x$ then $\ell_i = r_i = i$,*
5) *if $X[i] = 1$ then either $X[\ell_i, r_i] = p11\ldots1$ or $X[\ell_i, r_i] = x11\ldots1$,*
6) *if $X[i] = ($ then $\ell_i = i$ and there is j such that $\ell_i < j < r_i$ and either $X[j] = \vee$ or $X[j] = \wedge$,*
7) *if $X[i] =)$ then $r_i = i$ and there is j such that $\ell_i < j < r_i$ and either $X[j] = \vee$ or $X[j] = \wedge$,*
8) *if $X[i] = \wedge$ or $X[i] = \vee$ then $X[\ell_i, r_i]$ has the form "$(\ldots)$", and*

$$\ell_{i-1} = \ell_i + 1, r_{i-1} = i - 1, \ell_{i+1} = i + 1, r_{i-1} + 1 = r_i$$

9) *if $X[i] \in \{\neg, \exists, \forall\}$ then $\ell_i = i$,*
10) *all maximal substring of X of the form $x11\ldots1$ (with k 1's) is contained in a smallest pseudo formula of the form* (309) *where $i_1 = k$.*

Note that by the lexicographical minimality of $(r_i - \ell_i, r_i)$, in (5) above it is necessary that $r_i = i$. Similarly, in (6) it is necessary that $X[r_i] =)$, and in (7) we must have $X[\ell_i] = ($.

Exercise 10.3. *Prove the lemma. (Hint: for the "if" direction, prove by induction on the lengths of the pseudo formulas*

$$X[\ell_0, r_0], \ldots, X[\ell_n, r_n]$$

that they are all proper formulas except for the bound variables x_k are treated as free variables. Condition (10) guarantees that in X no x_k is free.)

It follows from Lemmas 10.1 and 10.2 that there is a $\mathbf{TC}^0$ algorithm (i.e., a $\mathbf{TC}^0$ circuit) that accepts precisely the encodings of formulas.

Definition 10.4. *For a proof system $\mathcal{F}$ let $FLA_{\mathcal{F}}(X)$ denote the property that X encodes a formula in $\mathcal{F}$. Let $PRF_{\mathcal{F}}(\pi, X)$ hold iff the string π encodes an $\mathcal{F}$-proof of the formula X. (We will omit the subscript $\mathcal{F}$ when it is clear from context.)*

In general, a proof system is a polytime function, and hence it is Σ_1^B-definable in $\mathbf{TV}^0$ and its graph $PRF_{\mathcal{F}}(\pi, X)$ is Δ_1^B-definable in $\mathbf{TV}^0$ (see Chapter 8). Here we are interested in formulas of special forms that represent $PRF_{\mathcal{F}}(\pi, X)$ and $FLA_{\mathcal{F}}(X)$. These forms will be useful for our proof of Lemma 10.9 and several results in Section 10B.2. Recall (Section 8C.2) that for each Σ_0^B formula $\varphi(y, \vec{x}, \vec{X}, Y)$ the **BIT-REC** axiom for φ is defined using the Σ_0^B formula

$$\varphi^{rec}(y, \vec{x}, \vec{X}, Y) \equiv \forall i < y(Y(i) \leftrightarrow \varphi(i, \vec{x}, \vec{X}, Y^{<i}))$$

(The notation $Y^{<i}$ stands for $Cut(i, Y)$ and is defined in (96) on page 133.) For the exercise below, the idea is that the formula

$$\varphi^{rec}(t+1, \vec{x}, \vec{X}, Y)$$

states that Y encodes a polytime computation for the relation $R(\vec{x}, \vec{X})$, and the bit $Y(t)$ is the "check bit": $Y(t)$ is true iff $R(\vec{x}, \vec{X})$ holds. (Note also that for strings Y of length $|Y| \le t+1$, $Y(t) \leftrightarrow |Y| = t+1$.)

EXERCISE 10.5. *Show that a relation $R(\vec{x}, \vec{X})$ is in* **P** *iff there are a Σ_0^B formula $\varphi(y, \vec{x}, \vec{X}, Z)$ and an $\mathcal{L}_A^2$ term $t(\vec{x}, \vec{X})$ so that both formulas below represent R:*

$$\varphi_1(\vec{x}, \vec{X}) \equiv \exists Y \le t+1\big(\varphi^{rec}(t+1, \vec{x}, \vec{X}, Y) \wedge Y(t)\big) \tag{310}$$

$$\varphi_2(\vec{x}, \vec{X}) \equiv \forall Y \le t+1\big(\varphi^{rec}(t+1, \vec{x}, \vec{X}, Y) \supset Y(t)\big) \tag{311}$$

and that

$$\mathbf{TV}^0 \vdash \varphi_1(\vec{x}, \vec{X}) \leftrightarrow \varphi_2(\vec{x}, \vec{X})$$

Generally, for each proof system $\mathcal{F}$ there are a Σ_0^B formula $\varphi_{\mathcal{F}}(y, \pi, X, Y)$ and a term $t_{\mathcal{F}}(\pi, X)$ such that

$$Prf_{\mathcal{F}}^{\Sigma}(\pi, X) \equiv \exists Y \le t_{\mathcal{F}}+1\big(\varphi_{\mathcal{F}}^{rec}(t_{\mathcal{F}}+1, \pi, X, Y) \wedge Y(t_{\mathcal{F}})\big) \tag{312}$$

$$Prf_{\mathcal{F}}^{\Pi}(\pi, X) \equiv \forall Y \le t_{\mathcal{F}}+1\big(\varphi_{\mathcal{F}}^{rec}(t_{\mathcal{F}}+1, \pi, X, Y) \supset Y(t_{\mathcal{F}}))\big) \tag{313}$$

both represent $PRF_{\mathcal{F}}(\pi, X)$ and

$$\mathbf{TV}^0 \vdash Prf_{\mathcal{F}}^{\Sigma}(\pi, X) \leftrightarrow Prf_{\mathcal{F}}^{\Pi}(\pi, X) \tag{314}$$

(Lemma 10.7 below shows that for the case of **G** and its subsystems, the theory $\mathbf{TV}^0$ can be replaced by $\mathbf{VTC}^0$. This fact will be useful, for example, for Theorem 10.44.)

We will also take the above forms for the formulas that represent the FLA relation.

COROLLARY 10.6. *There is an open $\mathcal{L}_{\mathbf{FTC}^0}$ formula $\psi(X)$ that represent $FLA_{\mathbf{G}}(X)$. The relation $FLA_{\mathbf{G}}(X)$ is Δ_1^B-definable in $\mathbf{VTC}^0$. Moreover, there are a Σ_0^B formula $\varphi_{FLA}(y, X, Y)$ and an $\mathcal{L}_A^2$ term*

t_{FLA} such that both formulas $Fla^{\Sigma}(X)$ and $Fla^{\Pi}(X)$ below represent $FLA_{\mathbf{G}}(X)$ and $\mathbf{VTC}^0 \vdash Fla^{\Sigma}(X) \leftrightarrow Fla^{\Pi}(X)$:

(315)
$$Fla^{\Sigma}(\pi, X) \equiv \exists Y \leq t_{FLA} + 1\big(\varphi^{rec}_{FLA}(t_{FLA}+1, X, Y) \wedge Y(t_{FLA})\big)$$
(316)
$$Fla^{\Pi}(\pi, X) \equiv \forall Y \leq t_{FLA} + 1\big(\varphi^{rec}_{FLA}(t_{FLA}+1, X, Y) \supset Y(t_{FLA})\big)\big)$$

Proof sketch. For the open $\mathcal{L}_{\mathbf{FTC}^0}$ formula $\psi(X)$, the idea is to combine the open $\mathcal{L}_{\mathbf{FTC}^0}$ formula $\varphi(i, \ell, r, X)$ from Lemma 10.1 and an $\Sigma^B_0(\mathcal{L}_{\mathbf{FAC}^0})$ formula that ensures that the conditions listed in Lemma 10.1 hold. The correctness of the formula follows from Lemma 10.2.

To prove the existences of the formulas Fla^{Σ} and Fla^{Π} as required, it is easier to start with a $\Sigma^B_0(\mathcal{L}_{\widehat{\mathbf{VTC}^0}})$ (i.e., $\Sigma^B_0(Numones')$) formula $\eta(X)$ that is equivalent to the above open $\mathcal{L}_{\mathbf{FTC}^0}$ formula $\psi(X)$ (see Theorem 9.34). The idea is to successively remove the occurrences of *Numones′* in η using the axiom *NUMONES* (which can be used as a defining axiom for *Numones′*). Note that *NUMONES* is already an instance of Σ^B_0-**BIT-REC**. ⊣

Now we show that for the subsystems of **G** in (314) we can use $\mathbf{VTC}^0$ instead of $\mathbf{TV}^0$.

Lemma 10.7. *For each proof system $\mathcal{F}$ that we have discussed (e.g., $\mathbf{G}$, $\mathbf{G}^\star_i$, $\mathbf{G}_i$, **eFrege**, etc.) there are a Σ^B_0 formula $\varphi_{\mathcal{F}}$ and an $\mathcal{L}^2_A$ term $t_{\mathcal{F}}$ so that the formulas $Prf^{\Sigma}_{\mathcal{F}}(\pi, X)$ and $Prf^{\Pi}_{\mathcal{F}}(\pi, X)$ as in (312) and (313) both represent the relation $PRF_{\mathcal{F}}(\pi, X)$, and such that*

$$\mathbf{VTC}^0 \vdash Prf^{\Sigma}_{\mathcal{F}}(\pi, X) \leftrightarrow Prf^{\Pi}_{\mathcal{F}}(\pi, X)$$

Proof sketch of Lemma 10.7. We will argue for **G**. The arguments for other proof systems F are similar.

First we sketch a $\mathbf{TC}^0$ algorithm that verifies that (i) π properly encodes a proof, and (ii) the last sequent in π is

$$\longrightarrow X$$

In fact, it can be shown that there is a $\Sigma^B_0(\mathcal{L}_{\mathbf{FTC}^0})$ formula $\psi'(\pi, X)$ that is true iff the above algorithm accepts (π, X). Therefore, as in Corollary 10.6, it is straightforward to obtain $Prf^{\Sigma}_{\mathcal{F}}$ and $Prf^{\Pi}_{\mathcal{F}}$ as desired.

Verifying (ii) is straightforward. For (i), note that by our encoding of proofs, the i-th sequent in π can be easily extracted from π, see (307). So first we check that each row $\pi^{[i]}$ of π is of the form (307) where $\mathcal{S}_i$ consists of two lists of formulas $\mathcal{S}^{[0]}_i$ and $\mathcal{S}^{[1]}_i$. Next, it remains to verify locally that for each i, either $\mathcal{S}_i$ is an axiom and $j = k = 0$, or $\mathcal{S}_i$ follows from sequent(s) $\mathcal{S}_j$ (and $\mathcal{S}_k$) by an inference rule.

Verifying that $\mathcal{S}_i$ is an axiom is straightforward. Now consider the case where $\mathcal{S}_i$ follows from $\mathcal{S}_j$ by the $\exists$-left rule; other rules are similar or easier. This case can be checked by

1) verifying that $\mathcal{S}_j$ and $\mathcal{S}_i$ are identical except for $\mathcal{S}_j$ contains a formula of the form $A(p_k)$ and $\mathcal{S}_i$ contains a formula of the form $\exists x_t A(x_t)$ at the same location in the antecedent,
2) verifying that p_k does not occur in $\mathcal{S}_i$.

It is easy to see that task (2) can be done by an $\mathbf{AC}^0$ algorithm. For (1) we first need to identify the scope of the existential quantifier $\exists x_t$: this is identified by using the pair (ℓ_s, r_s) as in Lemma 10.2, where s is the position of the $\exists$ symbol. Then we need to check that all occurrences of p_k have been properly replaced by x_t. For this the counting gates are used, for example, to count the number of occurrences of p_k and x_t in subformulas of $A(p_k)$ and $A(x_t)$, respectively.

For treelike proofs (e.g., $\mathbf{G}_i^\star$) we have to verify in addition that every sequent is used at most once. For this we simply check that all nonzero indices j, k as in (307) appear at most once. ⊣

Now we show, informally, that the polytime recognitions of formulas and proofs translates into uniform $\mathbf{G}_1^\star$ recognitions. We need the following notation.

DEFINITION 10.8. *For an $\mathcal{L}_A^2$ formula $\varphi(X)$ that might contain other free variables, and a constant string X_0 of length n, we use*

$$\varphi(X_0)[n]$$

to denote the propositional formula that is obtained from the translation $\varphi(X)[n]$ by plugging the values ($\top$, $\bot$) of the bits $X_0(j)$ for p_j^X.

(Thus, if X is the only free variable in $\varphi(X)$, then $\varphi(X_0)[n]$ is a sentence.)

LEMMA 10.9. *Let $\mathcal{F}$ be a proof system with defining formulas as in* (312) *and* (313). *Then there are polytime algorithm that outputs, for each inputs a string X_0 that encodes a formula and a string π_0 that encodes an $\mathcal{F}$-proof of X_0, a $\mathbf{G}_1^\star$ proof of the following sequent:*

$$\longrightarrow \mathit{Fla}^\Sigma(X_0)[m] \tag{317}$$

$$\longrightarrow \mathit{Fla}^\Pi(X_0)[m] \tag{318}$$

$$\longrightarrow \mathit{Prf}_{\mathcal{F}}^\Sigma(\pi_0, X_0)[n, m] \tag{319}$$

$$\longrightarrow \mathit{Prf}_{\mathcal{F}}^\Pi(\pi_0, X_0)[n, m] \tag{320}$$

In order to prove this lemma, we need the following result whose proof is left as an exercise.

EXERCISE 10.10. *Show that there is a polytime algorithm that, given any true Boolean sentence A, outputs a cut-free $\mathbf{PK}^\star$ proof of A. Hint: generate for each subformula B of A, a proof of*

$$\textit{either} \qquad \longrightarrow B \qquad \textit{or} \qquad B \longrightarrow .$$

Proof of Lemma 10.9. First we show how to compute in polytime a cut-free $\mathbf{G}_0^\star$ derivation of (317). Then note that

$$\mathbf{TV}^0 \vdash \mathit{Fla}^\Sigma(X) \longrightarrow \mathit{Fla}^\Pi(X)$$

Hence by the Translation Theorem for $\mathbf{V}^1$ there is a polytime function that computes a $\mathbf{G}_1^\star$ derivation of

$$\mathit{Fla}^\Sigma(X)[m] \longrightarrow \mathit{Fla}^\Pi(X)[m]$$

Then, given X_0 of length m, we obtain a derivation of

$$\mathit{Fla}^\Sigma(X_0)[m] \longrightarrow \mathit{Fla}^\Pi(X_0)[m]$$

simply by substituting the values of $X_0(i)$ for the variables p_i^X. From this and (317) we obtain a $\mathbf{G}_1^\star$ proof of (318) by a cut on the Σ_1^q formula $\mathit{Fla}^\Sigma(X_0)[m]$.

Deriving (319) and (320) is similar.

Now we prove (317). The idea is that by a polytime algorithm we can compute the (unique) string Y_0 of length $n = \mathit{val}(t_{FLA} + 1)$ that witnesses (315). Then by Exercise 10.10 we can generate a cut-free $\mathbf{PK}^\star$ for this witness. Finally (317) can be derived by a series of applications of the $\exists$ right rule.

Formally, by definition $\mathit{Fla}^\Sigma(X)[m]$ is the translation of

$$\exists Y \le t_{FLA} + 1(\varphi_{FLA}^{rec}(t_{FLA} + 1, X, Y) \wedge Y(t_{FLA}))$$

Let $r = \mathit{val}(t_{FLA})$. First we translate

$$\varphi_{FLA}^{rec}(t_{FLA} + 1, X, Y) \wedge Y(t_{FLA}) \tag{321}$$

for X of length m and each Y of length $k \le r + 1$. Note that when $k = r + 1$, $Y(t_{FLA})$ translates into $\top$ and hence (321) translates into

$$\varphi_{FLA}^{rec}(t_{FLA} + 1, X, Y)[m, r + 1]$$

On the other hand, for $k \le r$, $Y(t_{FLA})$ translates into $\perp$ and therefore (321) also translates into $\perp$. As a result, $\mathit{Fla}^\Sigma(X)[m]$ is

$$\exists p_0^Y \exists p_1^Y \dots \exists p_{r-1}^Y \big(\varphi_{FLA}^{rec}(t_{FLA} + 1, X, Y)[m, r + 1]\big)$$

So, in order to derive $\mathit{Fla}^\Sigma(X_0)[m]$ we first derive

$$\varphi_{FLA}^{rec}(t_{FLA} + 1, X_0, Y_0)[m, r + 1] \tag{322}$$

for the (unique) string Y_0 that satisfies $\varphi_{FLA}^{rec}(t_{FLA}+1, X_0, Y_0)$ and then apply the $\exists$ right rule r times.

Note that Y_0 can be computed in polytime by computing the bits $Y_0(0)$, $Y_0(1)$, … in that order. Now (322) is true, and by Exercise 10.10 we can compute a cut-free $\mathbf{PK}^\star$ proof for it in polytime. $\dashv$

The formulas Fla^{Σ}, Fla^{Π}, $Prf_{\mathcal{F}}^{\Sigma}$ and $Prf_{\mathcal{F}}^{\Pi}$ are important for this chapter. They will be used to define the Reflection Principle (Definition 10.35). Note that

$$\mathbf{V}^0 \vdash \forall X(Fla^{\Sigma}(X) \supset Fla^{\Pi}(X))$$

and (by Y-**IND** on the string Y in (313)):

$$\mathbf{V}^0 \vdash \forall\pi\forall X(Prf_{\mathcal{F}}^{\Sigma}(\pi, X) \supset Prf_{\mathcal{F}}^{\Pi}(\pi, X))$$

We also use Fla^{Σ} and $Prf_{\mathcal{F}}^{\Sigma}$ for the following notions.

DEFINITION 10.11. *Let $F(\vec{n})$ be a function in the language $\mathcal{L}$ of a theory $\mathcal{T}$. Suppose that $F(\vec{n})$ is the encoding of a formula $A_{\vec{n}}$, for all $\vec{n}$. We say that F* provably in $\mathcal{T}$ *computes $A_{\vec{n}}$ if*

$$\mathcal{T} \vdash \forall\vec{n}Fla^{\Sigma}(F(\vec{n})) \tag{323}$$

Similar, we say that a function $G(\vec{n})$ provably in $\mathcal{T}$ *computes an $\mathcal{F}$-proof $\pi_{\vec{n}}$ if $G(\vec{n}) = \pi_{\vec{n}}$ for all $\vec{n}$, and*

$$\mathcal{T} \vdash \forall\vec{n}Prf_{\mathcal{F}}^{\Sigma}(G(\vec{n})) \tag{324}$$

In these cases, we also say that the formulas $A_{\vec{n}}$ (resp. proofs $\pi_{\vec{n}}$) are provably in $\mathcal{T}$ computable by F (resp. G)*, or just* provably computable in $\mathcal{T}$.

We will often view a formula A as a tree whose leaves are labeled with the constants $\top$, $\bot$ or atomic subformulas p_k, x_k, and whose inner nodes are labeled with the Boolean connectives or quantifiers. Then all paths from the root to the leaves can be identified as follows. For each leaf B of the tree we can identify all pseudo formulas that contains B. Then it can be shown that these pseudo formulas are indeed all subformulas of A that contain B, and hence they form the path from the root of A to B. For this path, using the counting gates we can compute, for example, the alternation depths of quantifiers or connectives. It follows in particular that there is a $\mathbf{TC}^0$ number function, called *qdepth*, that computes the maximum alternation depth of quantifiers in X.

Some basic properties of proofs can be proved as theorems of our theories. We leave these as exercises.

EXERCISE 10.12. *Show that $\mathbf{VTC}^0$ proves the subformula property of $\mathbf{G}_i$ proofs, i.e., that if π is a $\mathbf{G}_i$ proof of a formula A, then all formulas in π are either in $(\Sigma_i^q \cup \Pi_i^q)$ or a subformula of A.*

EXERCISE 10.13. *Recall the notion of free variable normal form proofs from Section 2B.4. Show that there is a polytime function G so that for every treelike proof π, $G(\pi)$ is provably in* **VPV** *(Definition 8.18) a treelike proof in free variable normal form of the same endsequent. (Hint: we need to find all paths in π.)*

10A.2. Computing propositional translations in $\mathbf{TC}^0$. Recall from Chapter 7 (Sections 7B and 7E) that each bounded $\mathcal{L}_A^2$ formula $\varphi(\vec{x},\vec{X})$ is translated into a family $\|\varphi\|$ of propositional formulas $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$, for $\vec{m},\vec{n} \in \mathbb{N}$. Each formula $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$ is obtained from $\varphi(\vec{x},\vec{X})$ by substituting the numerals $\underline{\vec{m}}$ for $\vec{x}$ and introducing for each string variable X of intended length n the propositional variables p_i^X that represent the bits $X(i)$ of X (for $0 \le i < n-1$).

Thus, for a $\mathcal{L}_A^2$ formula φ the family $\|\varphi\|$ may involves bounded variables

$$p_0^X, p_1^X, \dots, \qquad p_0^Y, p_1^Y, \dots, \qquad \text{etc.}$$

and free variables

$$p_0^\alpha, p_1^\alpha, \dots, \qquad p_0^\beta, p_1^\beta, \dots, \qquad \text{etc.}$$

Encoding and verifying the formulas in $\|\varphi\|$ will be as described in Section 10A.1 above, except for here we may have different letters p^X, p^Y, etc. for bounded variables (instead of just x) and different letters p^α, p^β, etc. for free variables (instead of just p). A fixed formula φ contains a constant number of (bound and free) variables, so the encoding and verifying in Section 10A.1 can be easily extended to deal with the this situation. We leave the details to the reader.

Following the inductive definition of the propositional translations

$$\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$$

from Section 7B.1 and 7E we can show that the length of the encoding Y of $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$ can be expressed as some $\mathcal{L}_{\mathbf{FTC}^0}$ function $t_\varphi(\vec{m},\vec{n})$ and each bit of Y can be determined by some $\mathbf{TC}^0$ circuit. Intuitively, here we need the function *numones* in order to compute the lengths, e.g., of $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$ when φ is $\exists y \le t\psi(\vec{x},y,\vec{X})$; in this case:

$$\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}] \equiv \bigvee_{i=0}^{v} \psi(\vec{x},y,\vec{X})[\vec{m},i;\vec{n}]$$

(where $v = val(t)$) and

$$t_\varphi(\vec{m},\vec{n}) = (3v+2) + \sum_{i=0}^{v} t_\psi(\vec{m},i,\vec{n})$$

($3v+2$ is the number of parentheses and the binary connectives $\vee$.)

Formally, we prove:

Lemma 10.14. *For every $\mathcal{L}_A^2$ formula $\varphi(\vec{x},\vec{X})$ there is a $\Sigma_0^B(\mathcal{L}_{\mathbf{FTC}^0})$ formula $\psi(\vec{m},\vec{n},Y)$ such that for all $\vec{m}$, $\vec{n}$ and Y, $\psi(\vec{m},\vec{n},Y)$ is true iff Y encodes $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$.*

Proof idea. We can prove by structural induction on φ that there are an $\mathcal{L}_{\mathbf{FTC}^0}$ function $t_\varphi(\vec{m},\vec{n})$ as above and a $\Sigma_0^B(\mathcal{L}_{\mathbf{FTC}^0})$ formula

$\psi'(i, \vec{m}, \vec{n}, \ell, r, Y)$ that is true iff the bit $Y(i)$ has the right value when the substring

$$Y(\ell)Y(\ell+1)\dots Y(r \dot{-} 1)$$

encodes $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$. Then the formula $\psi(\vec{m}, \vec{n}, Y)$ can be defined as

$$|Y| = t_\varphi(\vec{m}, \vec{n}) \wedge \forall i < |Y| \psi'(i, \vec{m}, \vec{n}, 0, |Y|, Y) \qquad \dashv$$

Notation. For a (quantified) propositional formula A, we use $\widehat{A}$ to denote the string that encodes A.

The next corollary follows easily. (Recall Definition 10.11.)

Corollary 10.15. *For every $\mathcal{L}_A^2$ formula $\varphi(\vec{x}, \vec{X})$ there is an $\mathbf{FTC}^0$ function $T_\varphi(\vec{m}, \vec{n})$ that provably in $\overline{\mathbf{VTC}}^0$ computes $\widehat{\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]}$. Moreover, $\overline{\mathbf{VTC}}^0$ proves the definitions of the translation given in Section 7B.1 and 7E, such as*

$$T_{\exists y < t \varphi(y, X)}(n) = \widehat{A}$$

where

$$A = \bigvee_{0 \le i \le m-1} B_i, \qquad \text{for } B_i \text{ such that } T_{\varphi(\underline{i}, X)}(n) = \widehat{B_i}$$

Proof idea. Using the formula ψ and the function t_φ from Lemma 10.14, T_φ can be defined as follows:

$$T_{\varphi(\vec{x}, \vec{X})}(\vec{m}, \vec{n}) = Y \leftrightarrow \big(|Y| \le t_\varphi(\vec{m}, \vec{n}) \wedge \psi(\vec{m}, \vec{n}, Y)\big)$$

It is easy to see that T_φ is in $\mathcal{L}_{\mathbf{FTC}^0}$.

The fact that

$$\overline{\mathbf{VTC}}^0 \vdash Fla^\Sigma(T_\varphi(\vec{m}, \vec{n}))$$

and that $\overline{\mathbf{VTC}}^0$ proves the definitions of the translation as required are straightforward. $\dashv$

In Chapter 7 we have proved a number of Propositional Translation Theorems of the following form for a theory $\mathcal{T}$ and an associated proof system $\mathcal{P}$: for certain theorems $\varphi(\vec{x}, \vec{X})$ of $\mathcal{T}$, the families $\|\varphi\|$ of propositional tautologies $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ have polynomial-size proofs in $\mathcal{P}$. Here we will strengthen these theorems by showing that the $\mathcal{P}$-proofs of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ are in fact provably in $\overline{\mathbf{VTC}}^0$ computable by some $\mathbf{FTC}^0$ function $F_\varphi(\vec{m}, \vec{n})$ that depends on φ. In Section 10A.3 will prove one more such theorem for the theories $\mathbf{TV}^i$ and the proof systems $\mathbf{G}_i$ (where $i \ge 1$).

Theorem 10.16 below strengthens Theorem 7.61 in the way mentioned above. Here our propositional proofs of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ are computable in $\mathbf{TC}^0$ is because they consist of disjoint components each can be computed by an $\mathbf{TC}^0$ function. For example, suppose that $\mathcal{S}$ is a first-order sequent that is obtained from the sequent(s) $\mathcal{S}_1$ (and $\mathcal{S}_2$).

Then the propositional proof of $\mathcal{S}[\vec{m};\vec{n}]$ is obtained from the propositional proof(s) of $\mathcal{S}_1[\vec{m};\vec{n}]$ (and $\mathcal{S}_2[\vec{m};\vec{n}]$) by adding some derivations that can also be computed in $\mathbf{TC}^0$.

Theorem 10.16. *Suppose that $\varphi(\vec{x},\vec{X})$ is a bounded theorem of $\mathbf{V}^0$. Then there is a constant d and an $\mathbf{FTC}^0$ function F_φ so that provable in $\overline{\mathbf{VTC}}^0$, $F_\varphi(\vec{m},\vec{n})$ is a d-$\mathbf{G}_0^\star$ proof of $\varphi(\vec{a},\vec{\alpha})[\vec{m};\vec{n}]$, for all $\vec{m}$, $\vec{n}$.*

Proof Sketch. The constant d will be the same as in Theorem 7.61, and we will follow the proof of Theorem 7.61 to construct F_φ. Let π be the $\mathbf{LK}^2$-$\mathbf{V}^0$ proof of φ as in the proof of Theorem 7.61. For each sequent $\mathcal{S}$ in π we will construct an $\mathbf{FTC}^0$ function $F_{\mathcal{S},\pi}(\vec{m},\vec{n})$ that computes the d-$\mathbf{G}_0^\star$ proofs of the translations $\mathcal{S}[\vec{m};\vec{n}]$. Then $F_\varphi = F_{\mathcal{S}_0,\pi}$ for the last sequent $\mathcal{S}_0$ of π.

For each sequent $\mathcal{S}$, the function $F_{\mathcal{S},\pi}(\vec{m},\vec{n})$ is obtained by composition from some $\mathbf{FTC}^0$ functions (and earlier functions $F_{\mathcal{S}_1,\pi}$, $F_{\mathcal{S}_2,\pi}$ for parents $\mathcal{S}_1$, $\mathcal{S}_2$ of $\mathcal{S}$). Therefore it will be straightforward that $F_{\mathcal{S},\pi}$ are in $\mathbf{FTC}^0$. Moreover, the fact (324):

$$\overline{\mathbf{VTC}}^0 \vdash \forall\vec{m}\forall\vec{n}\mathit{Prf}^{\Sigma}(F_\varphi(\vec{m},\vec{n}))$$

can be proved by verifying at each step that

$$\overline{\mathbf{VTC}}^0 \vdash \forall\vec{m}\forall\vec{n}\mathit{Prf}^{\Sigma}(F_{\mathcal{S},\pi}(\vec{m},\vec{n}))$$

First, Exercise 7.62 can be strengthened to show that the translations of formulas in π are provably in $\mathbf{VTC}^0$ computable by an $\mathbf{FTC}^0$ functions that depends only on π. Details are left as an exercise (see also Corollary 10.15 above).

Exercise 10.17. *Show that for each Σ_0^B formula $\psi(\vec{x},\vec{X})$ in π there is an $\mathbf{FTC}^0$ function $G_{\psi,\pi}(\vec{m},\vec{n})$ that depends only on π and that, provably in $\overline{\mathbf{VTC}}^0$, computes the translation $\psi(\vec{x},\vec{X})[\vec{m};\vec{n}]$ of ψ.*

Following the proof of Theorem 7.61, it can be shown here that there are $\mathbf{TC}^0$ computable proofs of the tautology (155). This is left as an exercise (see also Exercise 7.58 and Theorem 7.8).

Exercise 10.18. *Suppose that $T_\varphi(i)$ is an $\mathbf{FTC}^0$ function that provably in $\mathbf{VTC}^0$ computes the translation $\varphi(x)[i]$ for an Σ_0^B formula $\varphi(x)$ as in Corollary 10.15. Show that there is an $\mathbf{FTC}^0$ function $H(\ell)$ that provably in $\overline{\mathbf{VTC}}^0$ computes a $\mathbf{PK}^\star$ proof of the sequent*

$$\longrightarrow \bigwedge_{i=0}^{\ell} \neg A_i, A_0 \wedge \bigwedge_{i=1}^{\ell} \neg A_i, A_1 \wedge \bigwedge_{i=2}^{\ell} \neg A_i, \dots, A_{\ell-1} \wedge \neg A_\ell, A_\ell$$

where A_j denotes $T_\varphi(j)$.

Now we proceed inductively as in the proof of Theorem 7.61. Here we can show that if $\mathcal{S}$ is derived from $\mathcal{S}_1$ (and $\mathcal{S}_2$), then the proof $F_{\mathcal{S},\pi}$

of $\mathcal{S}[\vec{m};\vec{n}]$ can be obtained by compositions from $F_{\mathcal{S}_1,\pi}$ (and $F_{\mathcal{S}_2,\pi}$) and some other $\mathbf{FTC}^0$ functions. $\dashv$

Formalizing the $\mathbf{V}^i$ Translation Theorem (Theorem 7.57) is similar and is left as an exercise.

Exercise 10.19. *Show that for each bounded theorem $\varphi(\vec{x},\vec{X})$ of $\mathbf{V}^i$ there is an $\mathbf{FTC}^0$ function $F_\varphi(\vec{m},\vec{n})$ that, provably in $\overline{\mathbf{VTC}}^0$, computes a proof of $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$.*

10A.3. Propositional Translation Theorem for $\mathbf{TV}^i$. Recall the theories $\mathbf{TV}^i$ from Section 8C. Analogous to the $\mathbf{V}^i$ Translation Theorem (Theorem 7.57) we will show here that theorems of $\mathbf{TV}^i$ translate into families of tautologies that have polynomial size $\mathbf{G}_i$ proofs. In fact, we will show that provably in $\overline{\mathbf{VTC}}^0$ these $\mathbf{G}_i$ proofs can be computed by $\mathbf{FTC}^0$ functions that depend only on the theorems of $\mathbf{TV}^i$. First we need the following facts whose proofs are left as exercises. (Recall that $\widehat{A}$ is the encoding of a propositional formula A.)

Exercise 10.20. *Let the functions $T_{\varphi(Z)}(n)$ and $T_{\psi(x)}(m)$ as in Corollary 10.15 such that, provably in $\overline{\mathbf{VTC}}^0$, $T_{\varphi(Z)}(n)$ computes*

$$A(p_0^Z,p_1^Z,\dots,p_{n-2}^Z) =_{\mathrm{def}} \varphi(Z)[n]$$

and $T_{\psi(x)}(m)$ computes

$$B_m =_{\mathrm{def}} \psi(x)[m]$$

Then the formulas $A(B_0,B_1,\dots,B_{n-2})$ is also provably computable in $\overline{\mathbf{VTC}}^0$ by some $\mathbf{FTC}^0$ function of the form $T_{\varphi'(y)}(n)$.

Below, Exercise 10.21 formalizes Lemma 7.49 and Exercise 10.22 formalizes Lemma 7.48 (for our translation formulas $\varphi[\vec{m};\vec{n}]$).

Exercise 10.21. *Suppose that the $\mathbf{FTC}^0$ functions $T_{\varphi(Z)}(n)$, $T_{\psi_1(x)}(i)$ and $T_{\psi_2(x)}(i)$ provably in $\overline{\mathbf{VTC}}^0$ compute a (quantified) formula $A(\vec{p})$ and quantifier-free formulas $\overrightarrow{B_i^1}$ and $\overrightarrow{B_i^2}$. Then there is an $\mathbf{FTC}^0$ function that provably in $\overline{\mathbf{VTC}}^0$ computes some $\mathbf{G}_0^\star$ proofs of*

$$A(\overrightarrow{B^1}), B_0^1 \leftrightarrow B_0^2, \dots, B_{n-2}^1 \leftrightarrow B_{n-2}^2 \longrightarrow A(\overrightarrow{B^2})$$

Exercise 10.22. *Consider a sequent of formulas in $(\Sigma_i^q \cup \Pi_i^q)$:*

$$\Gamma(\vec{p}),\Gamma' \longrightarrow \Delta(\vec{p}),\Delta'$$

Suppose that all formulas in $\Gamma(\vec{p})$ and $\Delta(\vec{p})$ are provably in $\overline{\mathbf{VTC}}^0$ computable by $\mathbf{FTC}^0$ functions of the form $T_{\varphi(Z)}(n)$, and all formulas in Γ' and Δ' are provably in $\overline{\mathbf{VTC}}^0$ computable by $\mathbf{FTC}^0$ functions of the form $T_{\varphi'}$. Suppose that B_i are quantifier-free formulas that are *provably in* $\overline{\mathbf{VTC}}^0$ *computable by an* $\mathbf{FTC}^0$ *function* $T_{\psi(x)}(i)$*. Then provably in* $\overline{\mathbf{VTC}}^0$ *there is a* $\mathbf{G}_i^\star$ *derivation of the form*

$$\dfrac{\Gamma(\vec{p}), \Gamma' \longrightarrow \Delta(\vec{p}), \Delta'}{\Gamma(\vec{B}), \Gamma' \longrightarrow \Delta(\vec{B}), \Delta'}$$

Now we prove the main theorem of this section.

Theorem 10.23 ($\mathbf{TV}^i$ Propositional Translation Theorem). *Let* $i \geq 1$*. For each bounded theorem* $\varphi(\vec{x}, \vec{X})$ *of* $\mathbf{TV}^i$ *there is an* $\mathbf{FTC}^0$ *function* $F(\vec{m}, \vec{n})$ *that, provably in* $\overline{\mathbf{VTC}}^0$*, computes a* $\mathbf{G}_i$ *proof of* $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$*, for all* $\vec{m}, \vec{n} \in \mathbb{N}$*.*

Proof. First we will translate first-order proofs of theorems of $\mathbf{TV}^i$ into propositional proofs as in Theorems 7.20 and 7.57. Then we will argue that the propositional proofs can be provably in $\overline{\mathbf{VTC}}^0$ computed by $\mathbf{FTC}^0$ functions that depend on the theorems of $\mathbf{TV}^i$. We will consider the case where $i = 1$; other cases are similar.

Recall that $\mathbf{LK}^2$-$\mathbf{TV}^1$ (Definition 8.78) is a complete system for $\mathbf{TV}^1$. To simplify our translation we modify the string induction rule **SIND** as follows. Let $S(X, Y)$ be a formula representing the graph of the string successor function, i.e., the "successor relation" (we redefine the symbol S used in Example 5.42 where it denotes the successor function; the exact meaning is easily understood from context):

$$S(X, Y) \equiv \forall i \leq |X| + |Y|$$
$$Y(i) \leftrightarrow \big(i \leq |X| \wedge \big((X(i) \wedge \exists j < i \neg X(j)) \vee (\neg X(i) \wedge \forall j < i X(j))\big)\big)$$

Now let Σ_1^B-$\mathbf{SIND}'$ be the rule:

$$\frac{\mathcal{S}_1}{\mathcal{S}_2} = \frac{\Gamma, A(\alpha), S(\alpha, \beta) \longrightarrow A(\beta), \Delta}{\Gamma, A(\varnothing) \longrightarrow A(\gamma), \Delta} \tag{325}$$

In this rule, A is a Σ_1^B formula, and α and β do not appear in Γ, Δ.

It is straightforward to verify that the modified $\mathbf{LK}^2$-$\mathbf{TV}^1$ system is also complete for $\mathbf{TV}^1$. (See the discussion for $\mathbf{LK}^2$-$\mathbf{TV}^1$ following Definition 8.78 and also the arguments for $\mathbf{LK}^2$-$\tilde{\mathbf{V}}^1$ in Section 6D.1.) In other words, a formula is a theorem of $\mathbf{TV}^1$ if and only if it has an anchored $\mathbf{LK}^2$ proof where all nonlogical axioms are instances of axioms of $\mathbf{V}^0$ and instances of the Σ_1^B-$\mathbf{SIND}'$ rule are allowed. (Here a proof is anchored if the cut formulas are instances of axioms of $\mathbf{V}^0$ or instances of $A(\varnothing)$ or $A(\gamma)$ in the bottom sequent of (325).)

Let π be an anchored $\mathbf{LK}^2$ proof of $\varphi(\vec{x}, \vec{X})$ where the rule (325) is allowed. For each sequent $\mathcal{S}(\vec{x}, \vec{X})$ in π we will define the propositional proofs $F_{\mathcal{S}}(\vec{m}, \vec{n})$ for the tautologies $\mathcal{S}[\vec{m}; \vec{n}]$. This is done inductively for all sequents in π, starting with the axioms. The base case (where $\mathcal{S}$ is an axiom) and most of the induction step have been dealt with in the proof of Theorem 7.57 (see also Exercise 10.19). The only remaining case for the induction step is the case of the Σ_1^B-**SIND**$'$ rule above.

Thus consider an instance of the rule Σ_1^B-**SIND**$'$ rule. Suppressing other free variables in $\mathcal{S}_1$ and $\mathcal{S}_2$, for the lengths ℓ, m, n of α, β, γ we have

$$\mathcal{S}_1[\ell, m, n] \equiv \Gamma[n], A(\alpha)[\ell], S(\alpha, \beta)[\ell, m] \longrightarrow A(\beta)[m], \Delta[n]$$
$$\mathcal{S}_2[n] \equiv \Gamma[n], A(\varnothing)[] \longrightarrow A(\gamma)[n], \Delta[n]$$

We need to show that for each n, $\mathcal{S}_2[n]$ can be derived from $\mathcal{S}_1[\ell, m, n]$ (for polynomially many values of ℓ and m) by some polynomial size $\mathbf{G}_1$ derivation. (Furthermore, the derivation is computable by $\mathbf{FTC}^0$ functions.)

At first sight such a derivation might seem impossible. Informally, assuming that both Γ and Δ are empty, then $\mathcal{S}_1$ allows us to obtain $A(\alpha + 1)$ from $A(\alpha)$ (here 1 is really the set $\{0\}$ and $+$ is the string addition function). So it appears that in order to get $A(\gamma)$ from $A(0)$ (i.e., $A(\varnothing)$) we need to use $\mathcal{S}_1$ γ times, i.e., we need exponentially many cuts.

Lemma 10.24 below shows that the number of cuts can be effectively reduced to just a polynomial in $|\gamma|$, by showing roughly that using $\mathcal{S}_1$ we can obtain $A(\beta)$ from $A(\alpha)$ for any β of length $|\beta| = |\alpha| + 1$.

Formally we use the following notation:

Notation. Let $S_k(X, Y)$ be the Σ_0^B formula

$$X \leq Y \wedge Y \leq X + \{k\}$$

(Recall that $\{k\}$, or also $POW2(k)$, is an $\mathbf{AC}^0$ function defined by $\{k\}(x) \leftrightarrow x = k$. See Example 8.46.)

LEMMA 10.24. *For each Σ_1^B formula $A(X)$ and distinct string variables α, β, σ, δ there is a $\mathbf{FTC}^0$ function $H(k, d)$ which is provably in $\overline{\mathbf{VTC}}^0$ a $\mathbf{G}_1$ derivation whose nonlogical axioms are from the set*

$$\{A(\alpha)[\ell], S(\alpha, \beta)[\ell, m] \longrightarrow A(\beta)[m] \ : \ \ell, m \leq d\}$$

and that contains all sequents in the set

$$\{A(\sigma)[s], S_k(\sigma, \delta)[s, n] \longrightarrow A(\delta)[n] \ : \ s, n \leq d\}$$

Lemma 10.24 completes the induction step for describing the proofs $F_{\mathcal{S}}(\vec{m}, \vec{n})$ of the translations $\mathcal{S}[\vec{m}; \vec{n}]$ of sequents $\mathcal{S}$ in π. It can be verified that $F_{\mathcal{S}}$ is in $\mathbf{FTC}^0$ when $\mathcal{S}$ is an axiom in π. When $\mathcal{S}$ is derived from $\mathcal{S}_1$ (and $\mathcal{S}_2$) then as in Theorem 10.16 and Exercise 10.19 it can be shown that $F_{\mathcal{S}}$ is obtained by composition from $F_{\mathcal{S}_1}$ (and $F_{\mathcal{S}_2}$) and some other $\mathbf{FTC}^0$ functions (here we need also the $\mathbf{FTC}^0$ functions from Lemma 10.24). Thus $F_{\mathcal{S}}$ are in $\mathbf{FTC}^0$ for all $\mathcal{S}$ in π. The fact that $\overline{\mathbf{VTC}}^0$ proves that $F_{\mathcal{S}}(\vec{m}, \vec{n})$ are proofs of $\mathcal{S}[\vec{m}; \vec{n}]$ can be proved by induction on the sequent $\mathcal{S}$. ⊣

Proof of lemma 10.24. First we will describe $H(k,d)$ simply as a polynomial-size derivation. The definition is by induction on k. Then we will argue that H is in fact a $\mathbf{FTC}^0$ function that provably in $\overline{\mathbf{VTC}}^0$ computes the desired derivation. From now on we will denote the desired derivation by

$$\frac{\{A(\alpha)[\ell], S(\alpha,\beta)[\ell,m] \longrightarrow A(\beta)[m] \;:\; \ell,m \le d\}}{\{A(\sigma)[s], S_k(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n] \;:\; s,n \le d\}} \tag{326}$$

Consider the base case, $k=0$. Note that $S_0(\sigma,\delta)[s,n]$ is false if $n<s$ or $n>s+1$, and in these cases

$$A(\sigma)[s], S_0(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n] \tag{327}$$

can easily be shown to have polynomial size $\mathbf{G}_0^\star$ proofs. So we focus on the cases $n=s$ or $n=s+1$.

By Exercise 10.21 there is provably in $\overline{\mathbf{VTC}}^0$ an $\mathbf{FTC}^0$-computable $\mathbf{G}_0^\star$ derivation of

$$A(\sigma)[s], (\sigma=\delta)[s,s] \longrightarrow A(\delta)[s] \tag{328}$$

Also, note that for $n \ne s$, the formula $(\sigma=\delta)[s,n]$ is false, and the sequent

$$A(\sigma)[s], (\sigma=\delta)[s,n] \longrightarrow A(\delta)[n] \tag{329}$$

can be shown to have polynomial size proof in $\mathbf{G}_0^\star$.

By Exercise 10.22 there are (provably in $\overline{\mathbf{VTC}}^0$) $\mathbf{FTC}^0$-computable $\mathbf{G}_1^\star$ derivations

$$\frac{A(\alpha)[s], S(\alpha,\beta)[s,n] \longrightarrow A(\beta)[n]}{A(\sigma)[s], S(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n]}$$

for $n=s$ and $n=s+1$. Combine these derivations we obtain $\mathbf{G}_1^\star$ derivations

(330)

$$\frac{\{A(\alpha)[s], S(\alpha,\beta)[s,m] \longrightarrow A(\beta)[m] \;:\; m \in \{s,s+1\}\}}{\{A(\sigma)[s], (\sigma=\delta)[s,n] \vee S(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n] \;:\; n \in \{s,s+1\}\}}$$

Now note that $\mathbf{V}^0$ proves

$$S_0(X,Y) \leftrightarrow (X=Y \vee S(X,Y))$$

So by Theorem 10.16 there is provably in $\overline{\mathbf{VTC}}^0$ an $\mathbf{FTC}^0$-computable $\mathbf{G}_0^\star$ derivation of

$$S_0(\sigma,\delta)[s,n] \longrightarrow (\sigma=\delta)[s,n] \vee S(\sigma,\delta)[s,n] \tag{331}$$

From this and (330) above we obtain a $\mathbf{G}_1^\star$ derivation

$$\frac{\{A(\alpha)[s], S(\alpha,\beta)[s,m] \longrightarrow A(\beta)[m] \;:\; s \le d, \text{ and } m=s \text{ or } m=s+1\}}{\{A(\sigma)[s], S_0(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n] \;:\; s \le d, \text{ and } n=s \text{ or } n=s+1\}}$$

Combine this and the derivations in (327) we obtain the derivation for the base case.

For the induction step, suppose that there is a polynomial size $\mathbf{G}_1$ derivations of the form (326):

$$\frac{\{A(\alpha)[\ell], S(\alpha,\beta)[\ell,m] \longrightarrow A(\beta)[m] \ : \ \ell, m \le d\}}{\{A(\sigma)[s], S_k(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n] \ : \ s,n \le d\}} \tag{332}$$

We will augment this derivation with additional derivations in order to obtain one that contains also all sequents in the set

$$\{A(\sigma)[s], S_{k+1}(\sigma,\delta)[s,n] \longrightarrow A(\delta)[n] \ : \ s,n \le d\} \tag{333}$$

By Exercise 10.22 there are $\mathbf{FTC}^0$ functions that provably in $\overline{\mathbf{VTC}}^0$ compute some derivations of the following sequents from the bottom sequents in (332):

$$\{A(\gamma)[n], S_k(\gamma,\delta)[n,p] \longrightarrow A(\delta)[p] \ : \ n,p \le d\} \tag{334}$$

From the sequents in (334) and the sequents at the bottom of (332) we obtain

$$\{A(\sigma)[s], S_k(\sigma,\gamma)[s,n] \wedge S_k(\gamma,\delta)[n,p] \longrightarrow A(\delta)[p] \ : \ s,n,p \le d\}$$

For each pair (s,p) $(s,p \le d)$, from the above sequents with $n = 0,1,\ldots,p$ using the $\vee$-left and $\exists$-left rules we obtain

$$A(\sigma)[s], \big(\exists Z \le |\delta|(S_k(\sigma,Z) \wedge S_k(Z,\delta))\big)[s,p] \longrightarrow A(\delta)[p] \tag{335}$$

Notice that $\{k\} + \{k\} = \{k+1\}$, and

$$\overline{\mathbf{VTC}}^0 \vdash S_{k+1}(\sigma,\delta) \longrightarrow \exists Z \le |\delta|(S_k(\sigma,Z) \wedge S_k(Z,\delta))$$

Therefore by Theorem 10.16 there is provably in $\overline{\mathbf{VTC}}^0$ an $\mathbf{FTC}^0$-computable $\mathbf{G}_0^\star$ proof of

$$S_{k+1}(\sigma,\delta)[s,p] \longrightarrow \big(\exists Z \le |\delta|(S_k(\sigma,Z) \wedge S_k(Z,\delta))\big)[s,p] \tag{336}$$

From this and (335) we obtain the following member of (333):

$$A(\sigma)[s], S_{k+1}(\sigma,\delta)[s,p] \longrightarrow A(\delta)[p]$$

This completes the description of the polynomial size derivation (326). Observe that the top sequents in (332) are used more than once, so the resulting derivation is daglike.

Now we show that $H \in \mathbf{FTC}^0$. The fact that provably in $\overline{\mathbf{VTC}}^0$ the function $H(k,d)$ computes the desired derivations is straightforward. That $H \in \mathbf{FTC}^0$ can be seen by observing that (i) $H(0,d)$ is a $\mathbf{FTC}^0$ function, and (ii) $H(k+1,d)$ is obtained from $H(k,d)$ by augmenting additional derivations that are computed by functions in $\mathbf{FTC}^0$. In other words, the string $H(k,d)$ consists of disjoint fragments that can be defined independently by $\mathbf{FTC}^0$ functions. Therefore $H(k,d)$ is in $\mathbf{FTC}^0$. ⊣

Recall that $\mathbf{V}^1$ is Σ_1^B-conservative over $\mathbf{TV}^0$ (Theorem 8.44 and Corollary 8.34) and $\mathbf{G}_1^\star$ is equivalent to **ePK** for proving prenex Σ_1^q formula (Theorem 7.54). Thus the $\mathbf{V}^i$ Translation Theorem (Theorem 7.57) shows that Σ_1^B theorems of $\mathbf{TV}^0$ translate into families of propositional tautologies that have polynomial-size **ePK** proofs. The next exercise is to formalize in $\overline{\mathbf{VTC}}^0$ a more direct proof of this fact.

Exercise 10.25 (Propositional Translation Theorem for $\mathbf{TV}^0$). *Show, by translating the axiom MCV (Definition 8.1) and using Theorem 10.16, that for each Σ_1^B theorem $\varphi(\vec{x},\vec{X})$ of $\mathbf{TV}^0$ there is a function F_φ in $\mathbf{FTC}^0$ that provably in $\overline{\mathbf{VTC}}^0$ computes an* **ePK** *proof of $\varphi(\vec{x},\vec{X})[\vec{m};\vec{n}]$.*

10B. The Reflection Principle

The Reflection Principle (RFN) for a proof system F states that F is sound, i.e., the endsequent of any F-proof is a valid sequent. In order to state the principle we need to formalize truth definitions, i.e., the relation $(Z \models X)$ that holds iff the truth assignment Z satisfies a propositional formula X. It is straightforward that for $i \geq 1$ the relation $(Z \models X)$ is in Σ_i^P (resp. Π_i^P) whenever X is a Σ_i^q (resp. Π_i^q) formula. When X is a quantifier-free propositional formula, it is also straightforward that $(Z \models X)$ is a polytime relation and is Δ_1^B-definable in $\mathbf{TV}^0$. (A difficult result, due to Buss, states that $(Z \models X)$ is an $\mathbf{NC}^1$ relation when X is a quantifier-free. See Section 10C.2.) Formulas that represent the relations $(Z \models X)$ (for different classes of X) are presented in Section 10B.1.

Using the formulas expressing $(Z \models X)$ we can state and prove the following "back and forth" properties. Let $\widehat{A}$ denote the string (of $\mathcal{L}_A^2$) that encode a propositional formula $A(\vec{p})$. Then for all truth assignments Z, intuitively the propositional translations of $(Z \models \widehat{A})$ are equivalent to $A(\overrightarrow{p^Z})$, where $\overrightarrow{p^Z}$ are the values of $\vec{p}$ under Z. In fact, we will give polytime algorithms that compute some $\mathbf{G}_0^\star$ proofs for these tautologies.

On the other hand, let $\varphi(Z)$ be a formula of $\mathcal{L}_A^2$ and $\widehat{A}$ be the string encoding the propositional translation $\varphi(Z)[n]$ of φ. Then for Z of length $|Z| = n$ we must have

$$(Z \models \widehat{A}) \leftrightarrow \varphi(Z)$$

In fact, we will show that this is a theorem of $\overline{\mathbf{VTC}}^0$. Detailed discussions are given in Section 10B.2.

The RFN will be defined in Section 10B.3. There we will show that the RFN for each system $\mathbf{G}_i^\star$ or $\mathbf{G}_i$ is provable in the associated theory. In Section 10B.4 we will show that for $i \geq 1$ the RFN for $\mathbf{G}_i^\star$ (resp.

$\mathbf{G}_i$) can be used to axiomatize the associated theories $\mathbf{V}^i$ (resp. $\mathbf{TV}^i$). Then in Section 10B.5 we will show that $\mathbf{G}_i^\star$ and $\mathbf{G}_i$ are the strongest (w.r.t. p-simulation) proof systems whose RFN can be proved in $\mathbf{V}^i$ and $\mathbf{TV}^i$, respectively.

Recall the Witnessing Theorem for $\mathbf{V}^1$ (Theorem 7.51). In Section 10B.6 we consider generally the problem of finding witness for a Σ_j^q formula

$$A(\vec{p}) \equiv \exists \vec{x} B(\vec{p}, \vec{x})$$

given a truth assignment to $\vec{p}$ and a $\mathbf{G}_i$ (or $\mathbf{G}_i^\star$) proof π of A. The Witnessing Problem is closely related to the RFN. In fact, our proof of the fact that $\mathbf{V}^1$ proves the Σ_1^q-RFN for $\mathbf{G}_1^\star$ is by formalizing the proof of Theorem 7.51. We will show in Section 10B.6 that the Witnessing Problems for the systems $\mathbf{G}_i$ and $\mathbf{G}_i^\star$ are complete for the classes that are definable in the associated theories.

10B.1. Truth Definitions. Suppose that X encodes a (quantified) propositional formula. Then each string Z specifies a truth assignment to the variables p_i in X as follows:

$$p_i \text{ is assigned the value of } Z(i)$$

Thus all possible truth assignment to variables in X can be specified by strings Z of length $|Z| \leq |X|$.

Here we present formulas that represent the relation

$$(Z \models X)$$

that holds for a truth assignment Z and a formula X iff Z satisfies X. We will consider separate cases where X is quantifier-free or belongs to Σ_i^q or Π_i^q where $i \geq 1$.

First, let

$$(Z \models_0 X)$$

hold iff X encodes a quantifier-free formula, and Z is a satisfying truth assignment to X. For Lemma 10.26 below we will present a polytime algorithm for $(Z \models_0 X)$ and conclude that the relation is Δ_1^B-definable in $\mathbf{TV}^0$. (Recall that a relation is polytime iff it is Δ_1^B-definable in $\mathbf{TV}^0$.) In Section 10C.2 we will show that $(Z \models_0 X)$ is indeed in $\mathbf{NC}^1$ and Δ_1^B-definable in $\mathbf{VNC}^1$.

Recall the axiom Σ_0^B-**BIT-REC** from Section 8C.2 (also the formulas $\mathit{Prf}_{\mathcal{F}}^{\Sigma}$, $\mathit{Prf}_{\mathcal{F}}^{\Pi}$ (312), (313)).

LEMMA 10.26. *There is a Σ_0^B formula $\varphi_0(y, X, E)$ and an $\mathcal{L}_A^2$ term $t_0(y, X)$ so that both*

$$(Z \models_0^{\Sigma} X) \equiv \exists E \leq t_0 + 1\big(\varphi_0^{rec}(t_0 + 1, X, Z, E) \wedge E(t_0)\big)$$
$$(Z \models_0^{\Pi} X) \equiv \forall E \leq t_0 + 1\big(\varphi_0^{rec}(t_0 + 1, X, Z, E) \supset E(t_0))\big)$$

represent $(Z \models_0 X)$ and such that

$$\mathbf{TV}^0 \vdash (Z \models_0^{\Sigma} X) \leftrightarrow (Z \models_0^{\Pi} X)$$

Proof idea. We start with the formula $\varphi_{FLA}(y, X, Y)$ and the term t_{FLA} from Corollary 10.6 that essentially states that Y encodes a computation of the relation $FLA(X)$. The "check bit" $Y(t_{FLA})$ indicates whether the computation accepts.

The idea is to augment such Y with an array E' that encode a polytime evaluation of the formula X. This evaluation only starts if the check bit $Y(t_{FLA})$ of Y is true, otherwise it simply rejects. So suppose that $Y(t_{FLA})$ is true. Suppose that X has length $(n+1)$ when being viewed as a string over the alphabet (308). Note that we can use a 2-dimensional array E' to evaluate the formula X bottom up in such a way that if the substring $X[i, j]$ is a subformula, then $E'(i, j)$ is the truth value of the subformula encoded by $X[i, j]$ (for $0 \leq i \leq j \leq n$). Then the truth value of X will be given by $E'(0, n)$.

To augment Y with E we will offset the string E' by some term $t' > t_{FLA}$. In addition, in order to conform with the axiom scheme **BIT-REC**, where the bits $E(i)$ is computed from $E^{<i}$, we can order the subformulas of X in increasing order of their lengths. Thus, for example, we can have the bit

$$E(a_{i,j}) \qquad \text{where } a_{i,j} = t' + r\langle n, n\rangle + \langle i, j\rangle$$

to represent the value of the subformula $X[i, j]$, where $r = j - i$ is roughly the length of $X[i, j]$.

For example, suppose that $X[i, j]$ is an atom p_s, then we have

$$E(a_{i,j}) \leftrightarrow Z(s)$$

For another example, suppose that $X[i, j]$ is the formula

$$(C \wedge D)$$

where $C = X[i+1, \ell]$ and $D = X[\ell+2, j-1]$ for some ℓ. Then φ_0 will contain a subformula

$$E(a_{i,j}) \leftrightarrow E(a_{i+1,\ell}) \wedge E(a_{\ell+2,j-1})$$

Informally, the formula φ_0 contain φ_{FLA} together the above subformulas and necessary Σ_0^B formulas that extract the subformulas $X[i, j]$ of X. The "check bit" for E is $E(t_0)$, where $t_0 = t' + n\langle n, n\rangle + \langle 0, n\rangle$. Finally, it is easy to see that $\mathbf{TV}^0$ proves the equivalence between $(Z \models_0^\Sigma X)$ and $(Z \models_0^\Pi X)$. $\dashv$

Recall that $\widehat{A}$ denotes the encoding of a propositional formula A.

Exercise 10.27. *Show that the theory* $\mathbf{V}^0$ *proves*

1) $(Z \models_0^\Sigma \widehat{A \wedge B}) \leftrightarrow \big((Z \models_0^\Sigma \widehat{A}) \wedge (Z \models_0^\Sigma \widehat{B})\big)$
2) $(Z \models_0^\Sigma \widehat{A \vee B}) \leftrightarrow \big((Z \models_0^\Sigma \widehat{A}) \vee (Z \models_0^\Sigma \widehat{B})\big)$
3) $(Z \models_0^\Sigma \widehat{\neg A}) \leftrightarrow \neg(Z \models_0^\Sigma \widehat{A})$

Now we consider the classes of formulas Σ_i^q and Π_i^q (for $i \geq 1$). Here it can be seen that evaluating Σ_i^q (resp. Π_i^q) sentences can be done in

Σ_i^P (resp. in Π_i^P). So in this case the formulas that represent $(Z \models X)$ belong to Σ_i^B (resp. Π_i^B).

Lemma 10.28. *Let $1 \le i \in \mathbb{N}$. There is a Σ_i^B formula $(Z \models_{\Sigma_i^q} X)$ that is true iff X encodes a Σ_i^q formula and the truth assignment Z satisfies X. Similarly, there is a Π_i^B formula $(Z \models_{\Pi_i^q} X)$ that is true iff X encodes a Π_i^q formula and the truth assignment Z satisfies X.*

Proof Idea. We show how to construct $(Z \models_{\Sigma_i^q} X)$; the formula $(Z \models_{\Pi_i^q} X)$ is constructed in the same way. The idea is to encode quantified propositional variables by bits of quantified string variables. Let A be the Σ_i^q formula encoded by X.

First, suppose that A is a prenex formula of the form

$$\exists \vec{x}_i \forall \vec{x}_{i-1} \dots Q \vec{x}_1 B$$

where B is a quantifier-free formula, and $Q \in \{\exists, \forall\}$: if i is odd then Q is $\exists$, otherwise Q is $\forall$. Then $(Z \models_{\Sigma_i^q} X)$ has the form

$$\exists X_i \le n \; \forall X_{i-1} \le n \; \dots Q X_1 \le n \psi(\widehat{B}, X_1, \dots, X_i, Z)$$

where $n = |X|$, ψ is in Σ_1^B if i is odd, and ψ is in Π_1^B if i is even. Here ψ is obtained as in Lemma 10.26 to express the fact that the truth assignment defined by Z and $X_1, X_2, \dots, X_i$ satisfies the formula B.

Now suppose that A is not in prenex form. Note that by definition no string quantifier in a Σ_i^B formula is in the scope of a number quantifier or a Boolean connective. So first we have to put A into prenex form. The procedure described in Theorem 2.75 may result in a formula not in Σ_i^q; for example, when applied it to the following Σ_2^q formula:

$$((\exists x_1 \; x_1 \wedge \forall x_2 \; (x_2 \vee \top)) \wedge \forall x_3 \; x_3)$$

we obtain

$$\forall x_3 \exists x_1 \forall x_2 ((x_1 \wedge (x_2 \vee \top)) \wedge x_3)$$

which is a Π_3^q formula.

$\mathbf{TC}^0$ prenexification: Another way of getting a prenex Σ_i^q formula equivalent to A is as follows. First we compute the quantifier depth of each quantified variable using the function *qdepth* mentioned on page 346. (Here we can assume that A has an outer most existential quantifier.) After renaming the quantified variables (so that they are distinct) we can safely move the quantifiers into their proper block in the prenex. Consider a quantifier $\exists x_i$ or $\forall x_i$ that occurs in X at position t. We will simply rename simultaneously all occurrences of x_i that are caught by this quantifier to x_{n+t}, where n is the length of the original formula A. Note that in A all variables have indices at most n. Also, all variables (including both bounded and free variables) in the new formula will have distinct indices.

Consider for example the following scenario:

$$\dots \exists x_2 (\dots \forall x_2 (\dots x_2 \dots) \dots x_2 \dots) \dots$$

where the $\exists$ is at position 7 and the $\forall$ is at position 20, and n is 100. Then the first and the fourth occurrences of x_2 are renamed to x_{107}, while the other two occurrences of x_2 are renamed to x_{120}.

It can be seen that the length of the resulting formula is at most n^2. It can be seen that the transformation can be done by a $\mathbf{TC}^0$ algorithm. In fact, it can be shown that there are a Σ_1^B formula $\varphi_1(X, X')$ and a Π_1^B formula $\varphi_2(X, X')$ that are true iff X' is the result of the transformation of X described above, and such that

$$\mathbf{VTC}^0 \vdash \varphi_1(X, X') \leftrightarrow \varphi_2(X, X')$$

and

$$\mathbf{VTC}^0 \vdash \exists X' \le n^2 \varphi_1(X, X')$$

Now the Σ_i^B formula $Z \models_{\Sigma_i^q} X$ has the form

$$\exists X_i \le n\, \forall X_{i-1} \le n\, \ldots Q X_1 \le n Q X' \le n^2 \psi(X', \vec{X}, Z) \tag{337}$$

where Q is $\exists$ if i is odd and Q is $\forall$ otherwise. Suppose that i is odd. Then ψ is a Σ_1^B formula; it is is obtained from φ_1 and the Σ_1^B formula (obtained as in Lemma 10.26) that expresses the fact that the truth assignment defined by Z and $X_1, X_2, \ldots, X_i$ satisfies the formula coded by X'. The case where i is even is similar. $\dashv$

Exercise 10.29. *Show that the following are theorems of* $\mathbf{V}^0$*:*

1) $(Z \models_{\Sigma_i^q} \widehat{A \wedge B}) \leftrightarrow \big((Z \models_{\Sigma_i^q} \widehat{A}) \wedge (Z \models_{\Sigma_i^q} \widehat{B})\big)$
2) $(Z \models_{\Sigma_i^q} \widehat{A \vee B}) \leftrightarrow \big((Z \models_{\Sigma_i^q} \widehat{A}) \vee (Z \models_{\Sigma_i^q} \widehat{B})\big)$
3) $(Z \models_{\Sigma_i^q} \widehat{\neg A}) \leftrightarrow \neg(Z \models_{\Sigma_i^q} \widehat{A})$
4) $(Z \models_{\Sigma_i^q} \widehat{\exists x A(x)}) \leftrightarrow \big((Z \models_{\Sigma_i^q} \widehat{A(\bot)}) \vee (Z \models_{\Sigma_i^q} \widehat{A(\top)})\big)$
5) $(Z \models_{\Sigma_i^q} \widehat{\forall x A(x)}) \leftrightarrow \big((Z \models_{\Sigma_i^q} \widehat{A(\bot)}) \wedge (Z \models_{\Sigma_i^q} \widehat{A(\top)})\big)$ *(if* $\forall x A(x)$ *is a* Σ_i^q *formula).*

Give similar theorems of $\mathbf{V}^0$ *that involve* $(Z \models_{\Pi_i^q} X)$*.*

10B.2. Formalization vs Propositional Translation. In this section we consider a kind of back and forth relationship between propositional translation (from first-order theories to proof systems) and the formalization of propositional proofs in our theories.

Consider, for example, a quantifier-free propositional formula

$$A(p_0, p_1, \ldots, p_{n-1})$$

As before let $\widehat{A}$ be the encoding of A. Recall that for Z of length $|Z| = n + 1$, the intended meaning of $(Z \models_0 \widehat{A})$ defined in Section 10B.1 is

$$A(p_0^Z, p_1^Z, \ldots, p_{n-1}^Z)$$

Therefore, intuitively, the propositional formulas

$$(Z \models_0^\Pi \widehat{A})[n+1] \qquad \text{and} \qquad (Z \models_0^\Sigma \widehat{A})[n+1]$$

should both be equivalent to $A(p_0^Z, p_1^Z, \ldots, p_{n-1}^Z)$.

We will show that there are polytime algorithms that compute $\mathbf{G}_1^\star$ proofs of these equivalences. In addition, if A is a Φ formula (where $\Phi \in \{\Sigma_i^q, \Pi_i^q\}$ for $i \geq 1$) then there is a polytime algorithm that computes a $\mathbf{G}_i^\star$ proof of the equivalence between $A(\overrightarrow{p^Z})$ and $(Z \models_\Phi \widehat{A})[n+1]$. In other words, the systems $\mathbf{G}_i^\star$ proves the correctness of the composition of our encoding and translation (and the $\mathbf{G}_i^\star$ proofs can be computed in polytime).

Finally, we will turn the above observation around and show that the theories $\mathbf{VTC}^0$ proves the correctness of the composition of propositional translation and encoding. In particular, we will show that $\overline{\mathbf{VTC}}^0$ proves equivalences of the form

$$(Z \models_{\Sigma_i^q} \widehat{\varphi(Z)[n]}) \leftrightarrow \varphi(Z)$$

for first-order formulas $\varphi(Z)$ (here $n = |Z|$).

For the next theorem, recall (Definition 10.8) that for a constant string $\widehat{A_0}$, of length m, the notation

$$(Z \models_0^\Sigma \widehat{A_0})[m, n+1]$$

denotes the propositional formula with variables $\overrightarrow{p^Z}$ that is obtained from the translation $(Z \models_0^\Sigma X)[m, n+1]$ by plugging the values of the bits $\widehat{A_0}(j)$ for p_j^X.)

THEOREM 10.30. *There are polytime algorithms that, given a quantifier-free formula $A_0(\vec{p})$ where $\widehat{A_0}$ is of length m, compute $\mathbf{G}_1^\star$ proofs of the sequents:*

$$(Z \models_0^\Sigma \widehat{A_0})[m, n+1] \longrightarrow A_0(\overrightarrow{p^Z}) \tag{338}$$

and

$$(Z \models_0^\Pi \widehat{A_0})[m, n+1] \longrightarrow A_0(\overrightarrow{p^Z}) \tag{339}$$

PROOF IDEA. We will give an algorithm that computes a $\mathbf{G}_0^\star$ proofs of (339). Then since

$$\mathbf{TV}^0 \vdash (Z \models_0^\Pi \widehat{A_0}) \longrightarrow (Z \models_0^\Sigma \widehat{A_0})$$

by the Propositional Translation Theorem for $\mathbf{V}^1$ there is a polytime function that computes a $\mathbf{G}_1^\star$ proof of

$$(Z \models_0^\Sigma \widehat{A_0})[m, n+1] \longrightarrow (Z \models_0^\Pi \widehat{A_0})[m, n+1]$$

From these we get a $\mathbf{G}_1^\star$ proof of (338).

We will outline a construction for a proof of (339) and leave the details as an exercise. Let m be the length of A_0. Recall (Lemma 10.26) that $(Z \models_0^\Pi \widehat{A_0})$ is the Π_1^B formula

$$(Z \models_0^\Pi \widehat{A_0}) \equiv \forall E \leq t_0 + 1\big(\varphi_0^{rec}(t_0 + 1, \widehat{A_0}, Z, E) \supset E(t_0)\big)$$

The idea is to prove an instance of the sequent (339) where the bits of E have the right values, and then apply the $\forall$ left rule repeatedly.

In particular, note that the first $(t_{FLA}+1)$ bits of E parses $\widehat{A_0}$ and the remaining bits evaluate A_0 bottom up: if the substring $\widehat{A_0}[i,j]$ of $\widehat{A_0}$ encodes a subformula $A_{i,j}$ of A_0, then

$$E(a_{i,j}) \leftrightarrow A_{i,j}$$

for the term $a_{i,j}$ described in the proof of Lemma 10.26 (note that $a_{0,m} = t_0$).

Thus, as in Lemma 10.9 we can compute the "parsing" bits of E (and compute a cut-free $\mathbf{PK}^\star$ proof for their correctness) in polytime. For the "evaluating" bits in E, the only relevant bits are bits of the form $E(a_{i,j})$ as above, and here we substitute the subformulas $A_{i,j}$ for them.

More precisely, let $r = val(t_0)$. Consider the translation

$$\big(\varphi_0^{rec}(t_0+1, \widehat{A_0}, Z, E) \supset E(t_0)\big)[m, n+1, r+1] \tag{340}$$

Here the Boolean variables are

$$p_0^Z, \ldots, p_{n-1}^Z, p_0^E, \ldots, p_{r-1}^E$$

Let $B(\overrightarrow{p^Z})$ be the formula obtained from (340) by the substitution described above (i.e., replacing p_j^E by the appropriate constants or formulas). Also, for i,j such that $\widehat{A_0}[i,j]$ encodes a subformula $A_{i,j}$ of A_0, let $C_{i,j}(\overrightarrow{p^Z})$ denote the result of this transformation for the following subformula of (340):

$$\varphi_0(t_0+1, \widehat{A_0}, Z, E^{a_{i,j}})[m, n+1, r+1]$$

Now we can obtain a $\mathbf{G}_0^\star$ proof of the sequents of the form

$$B, C_{i,j}(\overrightarrow{p^Z}) \longrightarrow A_{i,j}$$

for all subformulas $A_{i,j}$ of A_0. When $A_{i,j}$ is A_0 the formula $C_{i,j}(\overrightarrow{p^Z})$ is a conjunct in B, and hence we can derive $B \longrightarrow A$. $\dashv$

Exercise 10.31. *Give details for the above construction and verify that it can be done in polytime.*

Theorem 10.32. *Let $i \geq 1$ and $\Phi \in \{\Sigma_i^q, \Pi_i^q\}$. There is a polytime algorithm that on input a prenex Φ formula $A(\vec{p})$ computes. $\mathbf{G}_1^\star$ proofs of the following sequents:*

$$(Z \models_\Phi \widehat{A})[n+1] \longrightarrow A(\overrightarrow{p^Z})$$

If A is is not a prenex formula, then the proofs are in $\mathbf{G}_i^\star$.

Proof sketch. We prove the first statement by induction on $i \geq 0$. The base case is established in Theorem 10.30 above. For the induction step, suppose for example that A has the form $\exists x B(x)$. Then it can be seen that there is a polynomial size derivation of the form

$$\frac{(Z \models_\Phi \widehat{B})[n+1] \longrightarrow B(\overrightarrow{p^Z})}{(Z \models_\Phi \widehat{A})[n+1] \longrightarrow A(\overrightarrow{p^Z})}$$

using the $\exists$ introduction rules.

For the second statement, first let A' be the prenex formula equivalent to A as output by the $\mathbf{TC}^0$ prenexification procedure described in the proof of Lemma 10.28. We leave the proofs of the following facts as an exercise:

EXERCISE 10.33. *Show that there are polytime algorithms that compute cut-free* $\mathbf{G}^\star$ *proofs of the following sequents:*

$$A' \longrightarrow A \qquad \textit{and} \qquad A \longrightarrow A'$$

Now by the first statement there is a polynomial size $\mathbf{G}_0^\star$ proof of

$$(Z \models_\Phi \widehat{A'})[n+1] \longrightarrow A'(\overrightarrow{p^Z})$$

The desired sequent can now be derived as follows:

$$\frac{(Z \models_\Phi \widehat{A'})[n+1] \longrightarrow A'(\overrightarrow{p^Z}) \qquad A'(\overrightarrow{p^Z}) \longrightarrow A(\overrightarrow{p^Z})}{(Z \models_\Phi \widehat{A})[n+1] \longrightarrow A(\overrightarrow{p^Z})}$$

(We need to cut on the Σ_i^q prenex formula A'.) $\dashv$

Now we prove the category-theoretic reverse direction of Theorems 10.30 and 10.32. Let $\varphi(Z)$ be a Σ_i^B formula whose only free variable is Z (for some $i \geq 0$). By Corollary 10.6 there is an $\mathbf{FTC}^0$ function $T_\varphi(n)$ that provably in $\overline{\mathbf{VTC}}^0$ computes the encoding of $\varphi(Z)[n]$, for all n. Formally,

$$T_\varphi(n) = \widehat{\varphi(Z)[n]}$$

and

$$\overline{\mathbf{VTC}}^0 \vdash \forall n Fla^\Sigma(T_\varphi(n))$$

(Note that here $\varphi(Z)[n]$ is already in prenex form, so for the formula $(Z \models_{\Sigma_i^q} T_\varphi(n))$ we do not have to apply the $\mathbf{TC}^0$ prenexification described in the proof of Lemma 10.28.)

Now intuitively it should be clear that

$$(Z \models_{\Sigma_i^q} \widehat{\varphi(Z)[n]}) \iff \varphi(Z)$$

We will show that this equivalence is indeed provable in our theory $\overline{\mathbf{VTC}}^0$.

Notation. $(Z \models_{\Sigma_0^q} X)$ and $(Z \models_{\Pi_0^q} X)$ are defined to be $Z \models_0^\Sigma X$ and $Z \models_0^\Pi X$, respectively.

Theorem 10.34. *Let $i \geq 0$ and $\varphi(Z)$ be a Σ_i^B formula with a single free variable Z as shown. Then*

$$\overline{\mathbf{VTC}}^0 \vdash (Z \models_{\Sigma_i^q} \widehat{\varphi(Z)[n]}) \leftrightarrow \varphi(Z)$$

Similarly, if $\varphi(Z)$ is Π_i^B, then

$$\overline{\mathbf{VTC}}^0 \vdash (Z \models_{\Pi_i^q} \widehat{\varphi(Z)[n]}) \leftrightarrow \varphi(Z)$$

Proof idea. Let $n = |Z|$. First suppose that $i \geq 1$ and φ is a Σ_i^B formula. Then the block of bounded string quantifiers in φ also occurs in $(Z \models_{\Sigma_i^q} \widehat{\varphi(Z)[n]})$ (see (337) on page 359); in the latter it is followed by either a Σ_1^B or Π_1^B formula that expresses the $\models_0$ relation. Similarly when φ is a Π_i^B formula. So the case where $i \geq 1$ follows from the case where $i = 0$.

Thus suppose that φ is a Σ_0^B formula. We will show that

$$\overline{\mathbf{VTC}}^0 \vdash (Z \models_0^\Sigma \widehat{\varphi(Z)[n]}) \leftrightarrow \varphi(Z)$$

The fact that

$$\overline{\mathbf{VTC}}^0 \vdash (Z \models_0^\Pi \widehat{\varphi(Z)[n]}) \leftrightarrow \varphi(Z)$$

is similar.

Let A denote $\varphi(Z)[n]$. Recall from the proof of Lemma 10.26 that $(Z \models_0^\Sigma \widehat{A})$ has the form:

$$\exists E \leq t_0 + 1(\varphi_0^{rec}(t_0 + 1, \widehat{A}, Z, E) \wedge E(t_0))$$

where the first $(t_{FLA} + 1)$ bits of E encode a computation that parses the formula $\widehat{A}$, and the remaining bits in E evaluate A in a bottom up fashion, where the value of a subformula $A_{i,j}$ (encoded by $\widehat{A}[i,j]$) is stored as the bit $E(a_{i,j})$ for the term $a_{i,j}$ as in the proof of Lemma 10.26 (note that $a_{0,m} = t_0$, where m is the length of $\widehat{A}$).

Reasoning in $\overline{\mathbf{VTC}}^0$. First we show that

$$(Z \models_0^\Sigma \widehat{A}) \supset \varphi(Z)$$

Let E satisfy $\varphi_0^{rec}(t_0+1, \widehat{A}, Z, E) \wedge E(t_0)$. Then we can show by structural induction on the subformula φ_k of φ that

(341) $$\varphi_k \leftrightarrow E(a_{\ell_k, r_k})$$

where ℓ_k, r_k are the indices so that $\widehat{A}[\ell_k, r_k]$ encodes the translation of φ_k. As a result, from $E(t_0)$ (i.e., $E(a_{0,m})$) we conclude $\varphi(Z)$.

Now we show that

$$\varphi(Z) \supset (Z \models_0^\Sigma \widehat{A})$$

Here, informally, we need to prove the existences of the string E that parses and then evaluates $\widehat{A}$. Recall Corollary 10.15 that $\widehat{A} = \widehat{\varphi(Z)[n]}$

is provably computable in $\overline{\mathbf{VTC}}^0$ by an $\mathbf{FTC}^0$ function. So, informally, the "parsing" part in E (i.e., upto bit $E(t_{FLA})$) exists because

$$\overline{\mathbf{VTC}}^0 \vdash Fla^{\Sigma}(\widehat{A})$$

(This part of E can be extracted from the string Y that satisfies $\varphi^{rec}_{FLA}(t_{FLA}+1, \widehat{A}, Y) \wedge Y(t_{FLA})$.)

The "evaluating" part in E can be proved exist by $\Sigma^B_0(\mathcal{L}_{\mathbf{FTC}^0})$-**COMP** using the observation (341). The fact that these bits satisfy φ^{rec}_0 is straightforward. Note that by assuming that $\varphi(Z)$ is true we also have that $E(a_{0,m})$ (i.e., $E(t_0)$) is true. ⊣

10B.3. RFN for Subsystems of G. For a class Φ of formulas and a propositional proof system $\mathcal{F}$, the Φ-Reflection Principle for $\mathcal{F}$, denoted by Φ-$RFN_{\mathcal{F}}$, asserts that all formulas of Φ that have an $\mathcal{F}$-proof is valid. Here we will show that for $i \geq 1$ the RFN for $\mathbf{G}^{\star}_i$ (resp. $\mathbf{G}_i$) is provable in the associated theories $\mathbf{V}^i$ (resp. $\mathbf{TV}^i$). In Section 10B.4 we will show that indeed the theories can be axiomatized using the RFN of the associated proof systems.

To state the principle we need the formulas $(Z \models_{\Sigma^q_i} X)$ and $(Z \models_{\Pi^q_i} X)$ from Section 10B.2. Recall that $(Z \models_{\Sigma^q_0} X)$ and $(Z \models_{\Pi^q_0} X)$ stands for $(Z \models^{\Sigma}_0 X)$ and $(Z \models^{\Pi}_0 X)$, respectively. Recall also the formulas Fla^{Σ}, Fla^{Π}, $Prf^{\Sigma}_{\mathcal{F}}$, and $Prf^{\Pi}_{\mathcal{F}}$ (see Corollary 10.6 and Lemma 10.7).

Notation. For $i \geq 0$ and $\Phi \in \{\Sigma^q_i, \Pi^q_i\}$, let $Fla^{\Pi}_{\Phi}(X)$ (resp. $Fla^{\Sigma}_{\Phi}(X)$) be the Π^B_1 (resp. Σ^B_1) formula that represents the relation $FLA(X)$ for formulas X in Φ.

DEFINITION 10.35 (The Reflection Principle). *For a proof system $\mathcal{F}$ and $\Phi \in \{\Sigma^q_i, \Pi^q_i\}$ ($i \geq 0$) the Φ-Reflection Principle for $\mathcal{F}$, denoted Φ-$RFN_{\mathcal{F}}$, is the $\mathcal{L}^2_A$ sentence defined as follows:*

$$\Sigma^q_i\text{-}RFN_{\mathcal{F}} \equiv \forall\pi\forall X\forall Z\big((Fla^{\Pi}_{\Sigma^q_i}(X) \wedge Prf^{\Pi}_{\mathcal{F}}(\pi, X)) \supset (Z \models_{\Sigma^q_i} X)\big)$$

$$\Pi^q_i\text{-}RFN_{\mathcal{F}} \equiv \forall\pi\forall X\forall Z\big((Fla^{\Sigma}_{\Pi^q_i}(X) \wedge Prf^{\Sigma}_{\mathcal{F}}(\pi, X)) \supset (Z \models_{\Pi^q_i} X)\big)$$

Note that for $i \geq 1$ Σ^q_i-$RFN_{\mathcal{F}}$ is equivalent to a $\forall\Sigma^B_i$ sentence, and Π^q_i-$RFN_{\mathcal{F}}$ is equivalent to a $\forall\Pi^B_i$ sentence. Also, Σ^q_0-$RFN_{\mathcal{F}}$ (or simply 0-$RFN_{\mathcal{F}}$) is equivalent to a $\forall\Sigma^B_1$ sentence, while Π^q_0-$RFN_{\mathcal{F}}$ is equivalent to a $\forall\Sigma^B_0$ sentence.

Observe that asserting that a formula $A(\vec{p})$ of the form

$$\forall\vec{x}B(\vec{p}, \vec{x})$$

is valid is essentially equivalent to asserting that $B(\vec{p}, \vec{q})$ is valid. So, if an $\mathcal{F}$-proof of any Π^q_{i+1} formula $A(\vec{p})$ can be transformed (in a theory $\mathcal{T}$) into a proof of B (or some other Σ^q_i formula), then $\mathcal{T}$ proves

$$\Sigma^q_i\text{-}RFN_{\mathcal{F}} \supset \Pi^q_{i+1}\text{-}RFN_{\mathcal{F}}$$

We illustrate this in the next lemma where we prove the implication for treelike proof systems. The transformation in this case can be computed by a polytime function, and this explains why the implication is provable in $\mathbf{TV}^0$.

Lemma 10.36. *For $i, j \geq 0$,*

$$\mathbf{TV}^0 \vdash \Sigma_i^q\text{-}RFN_{\mathbf{G}_j^\star} \supset \Pi_{i+1}^q\text{-}RFN_{\mathbf{G}_j^\star}$$

Proof sketch. Assuming Σ_i^q-$RFN_{\mathbf{G}_j^\star}$ we need to prove Π_{i+1}^q-$RFN_{\mathbf{G}_j^\star}$. Thus let π be a $\mathbf{G}_j^\star$ proof of a Π_{i+1}^q formula $A(\vec{p})$. Informally we need to show that A is valid.

If A is in Σ_i^q then we can use Σ_i^q-$RFN_{\mathbf{G}_j^\star}$ and the conclusion is trivial. So suppose that A is in $(\Pi_{i+1}^q - \Sigma_i^q)$. Using the fact that π is a treelike proof, we will transform π into a proof of a Σ_i^q formula $A'(\vec{p}, \vec{q})$ so that

$$\forall A' \supset \forall A \tag{342}$$

and such that the transformation is in polytime. Then by Σ_i^q-$RFN_{\mathbf{G}_j^\star}$ we have that A' is valid, and hence A is valid.

Below we will describe the transformation and the formula A'. They can be computed from π and A in polytime. So (342) can be formalized and proved in $\mathbf{TV}^0$ and we are done.

There are two cases depending on whether A is a prenex formula or not. We consider the simpler case first.

Case I: A is a prenex formula. Here A has the form

$$\forall x_m \ldots \forall x_1 B(\vec{p}, x_1, \ldots, x_m) \tag{343}$$

where B is a prenex Σ_i^q formula. Since π is treelike, we can assume that π is in free variable normal form (recall Section 2B.4 and see Exercise 10.13).

The idea is to first transform π into a proof of a sequent of the form

$$\longrightarrow B(\vec{p}, \overrightarrow{q^1}), B(\vec{p}, \overrightarrow{q^2}), \ldots, B(\vec{p}, \overrightarrow{q^k}) \tag{344}$$

for some k. Here $\overrightarrow{q^t}$ are all eigenvariables that introduce the universal variables $\vec{x}$ shown in (343). Intuitively, we retain these eigenvariables by ignoring the $\forall$-right rule.

Formally, suppose that C is a $(\Pi_{i+1}^q - \Sigma_i^q)$ ancestor of A in the succedent of a sequent $\mathcal{S}$ in π. Then C has the form

$$\forall x_t \ldots \forall x_1 B(\vec{p}, x_1, \ldots, x_t, q_{t+1}, \ldots, q_m)$$

(for some t, $1 \leq t \leq m$). Note that C can only be in the succedent of $\mathcal{S}$. We transform $\mathcal{S}$ by replacing C by a list of formulas as in (344) that contains all ancestors of C of the type $B(\vec{p}, \vec{q})$.

The replacement above is performed for all such C. Let $\mathcal{S}'$ denote the transformed sequent, and π' denote the transformed proof. We can easily turn π' into a legitimate proof by (i) deleting the $\forall$-right that introduces the variables $\forall\vec{x}$ of A as shown in (343) as well as contraction right involving C, and (ii) inserting necessary weakenings.

Finally, from a proof of the sequent of the form (344) using the $\vee$-right we obtain a proof of $A'(\vec{p},\vec{q})$ where A' has the form

$$\bigvee B(\vec{p},\overrightarrow{q^{\ell}}) \tag{345}$$

Case II: A is not a prenex formula. The description of A' in this case is more complicated, so we only outline the arguments here. For illustration, consider a subformula A_1 of A of the form (343) where here B is in Σ_i^q but is not necessarily in prenex form. Then following the above procedure, A_1 is replaced by a formula A_1' of the form (345).

We need to extend the above transformation to other $(\Pi_{i+1}^q - \Sigma_i^q)$ subformulas of A. The transformation will be done in a top-down fashion. Thus, for example, a superformula of A_1 may be replaced by several different copies all containing A_1. These copies of A_1 can then be replaced by different disjunctions of the form (345).

This motivates the following definition. For simplicity, assume that in A all $\neg$ connective only occur in front of atoms.

DEFINITION 10.37. *For a formula A in $(\Pi_{i+1}^q - \Sigma_i^q)$, a Σ_i^q-expansion of A is any Σ_i^q formula that can be obtained from A by finitely many repeated applications of the operations that consists of the following steps:*

1) *let A_1 be a non-Σ_i^q subformula of A;*
2) *replace A_1 by a formula as follows:*
   - *if A_1 has the form $\forall x B(x)$ then let $q_1, q_2, \ldots, q_r$ be a list of new free variables (q_t need not be distinct), and replace A_1 by the disjunction*

$$\bigvee_{1\le t\le r} B(q_t)$$

   - *otherwise A_1 is replaced by $(A_1 \vee A_1)$.*

For example, the formula in (345) is a Σ_i^q-expansion of (343). For another example, suppose

$$A \equiv \forall x_1\big(\exists y_1 B(x_1,y_1) \wedge \forall x_2 \exists y_2 C(x_1,x_2,y_2)\big)$$

where B, C are quantifier-free formulas. Then the following formula is a Σ_1^q-expansion of A:

$$\exists y_1 B(x_1,q_1) \wedge \big(\exists y_2 C(q_1,q_2,y_2) \vee \exists y_2 C(q_1,q_3,y_2)\big)$$

Now, the $\mathbf{G}_j^\star$ proof π of $A(\vec{p})$ can be transformed into a $\mathbf{G}_j^\star$ proof π' of an Σ_i^q-expansion $A'(\vec{p},\vec{q})$ of A. The transformation can be seen to be computable by a polytime function, and the formalization of (342) can be shown to be provable in $\mathbf{TV}^0$. $\dashv$

Now we prove the RFN of the systems $\mathbf{G}_i^\star$ and $\mathbf{G}_i$ in our theories. Informally, in the next theorem we use two approaches for showing that the endsequent of given proof π (in $\mathbf{G}_i^\star$ or $\mathbf{G}_i$) is valid. The first approach is to proceed by induction on the length of π to show that all sequents in π are valid. Notice that if $A(\vec{p})$ is a Σ_i^q or Π_{i+1}^q formula and $\widehat{A}$ encodes A, then the statement of A is valid:

$$\forall Z \leq |\widehat{A}|(Z \models_{\Sigma_i^q} \widehat{A})$$

is in Π_{i+1}^B. Thus the inductive argument requires Π_{i+1}^B-**IND**.

Another approach for proving the RFN for $\mathbf{G}_1^\star$ is to formalize the proof of the Witnessing Theorem for $\mathbf{G}_1^\star$ (Theorem 7.51). In general, suppose that $A(\vec{p})$ is a Σ_i^q formula of the form

$$\exists \vec{x} B(\vec{p}, \vec{x})$$

where B is a Π_{i-1}^q formula (here $i \geq 1$). We wish to define a witnessing function for A that, given the values for $\vec{p}$, computes $\vec{x}$ that satisfy $B(\vec{p}, \vec{x})$. The graph of the function is Π_{i-1}^q, so this suggests that the witnessing function is in $\mathcal{L}_{\mathbf{FP}^i}$ (see Definition 8.88). Below we will outline the formalization of the proof of the Witnessing Theorem for $\mathbf{G}_1^\star$. The proof of the general case is similar and will be left as an exercise.

Theorem 10.38. *For $i \geq 1$:*
(a) $\mathbf{V}^i \vdash \Pi_i^q\text{-}RFN_{\mathbf{G}_{i-1}}$.
(b) $\mathbf{V}^i \vdash \Sigma_i^q\text{-}RFN_{\mathbf{G}_i^\star}$

Proof. (a) Reasoning in $\mathbf{V}^i$. Let π be a $\mathbf{G}_{i-1}$ proof of a Π_i^q formula. By Exercise 10.12 all formulas in π are Π_i^q. Moreover, it can be shown that all formulas in the antecedents of sequents in π are in $(\Sigma_{i-1}^q \cup \Pi_{i-1}^q)$.

The idea is to prove by induction on t that the t-th sequent in π is valid. Suppose first that $i \geq 2$. Let

$$\mathcal{S}_t \;=\; A_0, \ldots, A_n \longrightarrow B_0, \ldots, B_m \tag{346}$$

be the t-th sequent in π. Here all A_j are in $(\Sigma_{i-1}^q \cup \Pi_{i-1}^q)$ and all B_k are in Π_i^q.

Definition 10.39. *For $i \geq 1$ define $(Z \models_i X)$ to be the formula*

$$((Z \models_{\Sigma_i^q} X) \vee (Z \models_{\Pi_i^q} X))$$

Note that $(Z \models_i X)$ is in $(\Sigma_{i+1}^B \cap \Pi_{i+1}^B)$.

Formally we will prove the following formula:

$$\forall Z \leq |\pi|\big(\forall j \leq n(Z \models_{i-1} A_j) \supset (Z \models_{\Pi_i^q} \bigvee_k B_k)\big) \tag{347}$$

Since both formulas $(Z \models_{i-1} X)$ and $Z \models_{\Pi_i^q} X)$ are Π_i^B, it follows from Corollary 6.24 that (347) is equivalent in $\mathbf{V}^i$ to a Π_i^B formula. As a result, we can prove (347) by induction on t (using Π_i^B-**IND**, see

Corollary 6.4). Both the base case and the induction step are straightforward.

Now we prove

$$\mathbf{V}^1 \vdash \Pi_1^q\text{-}RFN_{\mathbf{G}_0}$$

Let π be a $\mathbf{G}_0$-proof of a Π_1^q formula. Let $\mathcal{S}_t$ as in (346) be the t-th sequent in π, here all A_j are quantifier-free and all B_k are Π_1^q formulas. We prove in $\mathbf{V}^1$ the following formula:

$$\forall Z \leq |\pi|\big((Z \models_0^\Sigma \bigwedge_j A_j) \supset (Z \models_{\Pi_1^q} \bigvee_k B_k)\big) \tag{348}$$

Because $(Z \models_0^\Sigma X)$ is a Σ_1^B formula and $(Z \models_{\Pi_1^q} X)$ is a Π_1^B formula, (348) is equivalent in $\mathbf{V}^1$ to a Π_1^B formula. Therefore (348) can be proved in $\mathbf{V}^1$ by induction on t (using Π_1^B-**IND**, see by Corollary 6.4).

(b) Let π be a $\mathbf{G}_i^\star$ proof of a Σ_i^q formula A. First consider the case $i = 1$, and consider the interesting case where A is in $(\Sigma_1^q - \Sigma_0^q)$. Note that by the subformula property (see Exercise 10.12) all formulas in π are Σ_1^q.

To show that this formula is valid, the idea is to prove the Witnessing Theorem for $\mathbf{G}_1^\star$ (Theorem 7.51) that there is a polytime function that produces the witnesses for the existentially quantified variables. Recall that this requires Theorem 7.45 and the second half of Theorem 7.54. It is straightforward to formalize in $\mathbf{TV}^0$ the proof of both theorems, and hence also the proof of Theorem 7.51.

The proof for the case where $i > 1$ is similar. Here the outermost existentially quantified variables in A can be witnessed by some $\mathbf{FP}^{\Sigma_{i-1}^P}$. These witnessing functions can in fact be defined by examining π directly (without introducing an analogue of **ePK**). Details are left as an exercise. ⊣

EXERCISE 10.40. *Prove part (b) above for the case where $i > 1$.*

COROLLARY 10.41. *For $i \geq 0$:*

(a) $\mathbf{TV}^i \vdash \Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_{i+1}^\star}$

(b) $\mathbf{TV}^i \vdash \Pi_{i+1}^q\text{-}RFN_{\mathbf{G}_i}$.

PROOF. (a) By Lemma 10.36 it suffices to show that

$$\mathbf{TV}^i \vdash \Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_{i+1}^\star}$$

Now $\Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_{i+1}^\star}$ is a $\forall\Sigma_{i+1}^B$ sentence and by Theorem 10.38 (b) it is provable in $\mathbf{V}^{i+1}$. By Theorem 8.95 $\mathbf{V}^{i+1}$ is Σ_{i+1}^B-conservative over $\mathbf{TV}^i$, hence $\mathbf{TV}^i$ also proves $\Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_{i+1}^\star}$.

(b) Similar to part (a) here for $i \geq 1$ the sentence $\Pi_{i+1}^q\text{-}RFN_{\mathbf{G}_i}$ is $\forall\Pi_{i+1}^B$, which is the same as $\forall\Sigma_i^B$. So the fact that $\mathbf{TV}^i$ proves $\Sigma_i^q\text{-}RFN_{\mathbf{G}_i}$ follows from the fact that $\mathbf{V}^{i+1}$ proves $\Sigma_i^q\text{-}RFN_{\mathbf{G}_i}$ (Theorem 10.38 (a)) and the fact that $\mathbf{V}^{i+1}$ is Σ_{i+1}^B-conservative over $\mathbf{TV}^i$.

For the case where $i = 0$, Π^q_1-$RFN_{\mathbf{G}_0}$ is a $\forall\Pi^B_1$ sentence that is provable in $\mathbf{V}^1$. Since $\mathbf{V}^1$ is Σ^B_1-conservative over $\mathbf{TV}^0$, Σ^q_0-$RFN_{\mathbf{G}_0}$ is also provable in $\mathbf{TV}^0$. ⊣

Exercise 10.42. (a) *For $i \geq j \geq 0$, show that*

$$\mathbf{TV}^0 \vdash \Sigma^q_j\text{-}RFN_{\mathbf{G}^\star_{i+1}} \supset \Sigma^q_j\text{-}RFN_{\mathbf{G}_i}$$

(b) *For $i \geq 0$ and $j \geq 0$, show that*

$$\mathbf{TV}^0 \vdash \Sigma^q_j\text{-}RFN_{\mathbf{G}_i} \supset \Sigma^q_j\text{-}RFN_{\mathbf{G}^\star_{i+1}}$$

(Hint: formalize the p-simulations given in the proofs of Theorems 7.41 and 7.46.)

Exercise 10.43. *Let $Fla^\Pi_{\mathbf{PK}}$ and $Prf^\Pi_{\mathbf{ePK}}$ be Π^B_1 formulas that represent the relations $FLA_{\mathbf{PK}}$ and $PRF_{\mathbf{ePK}}$, respectively. The Reflection Principle for* **ePK** *is defined as follows:*

$$RFN_{\mathbf{ePK}} \equiv \forall\pi\forall X\forall Z\big((Fla^\Pi_{\mathbf{PK}}(X) \wedge Prf^\Pi_{\mathbf{ePK}}(\pi, X)) \supset (Z \models^\Sigma_0 X)\big)$$

Note that $RFN_{\mathbf{ePK}}$ is a $\forall\Sigma^B_1$ sentence. Show that

$$\mathbf{TV}^0 \vdash RFN_{\mathbf{ePK}}$$

10B.4. Axiomatizations using RFN. In this section we present results of the following type. We will show that the RFN of a proof system $\mathcal{F}$ ($\mathbf{G}^\star_i$ or $\mathbf{G}_i$) can be used together with a base theory (e.g., $\mathbf{VTC}^0$) to axiomatize the associated theory $\mathcal{T}$ ($\mathbf{V}^i$ or $\mathbf{TV}^i$). In Section 10B.3 above we have shown one direction, i.e., the RFN of $\mathcal{F}$ is provable in $\mathcal{T}$. For the other direction we need to show that all theorems of $\mathcal{T}$ are provable from the base theory and the RFN of $\mathcal{F}$. Informally, this can be seen as follows. First, the propositional version of each theorem of $\mathcal{T}$ have been shown to have proofs in $\mathcal{F}$ that are definable in $\mathbf{VTC}^0$ (Section 10A.2). So by the RFN for $\mathcal{F}$ these propositional translations are valid. Next, $\overline{\mathbf{VTC}}^0$ proves that the validity of the propositional translations implies the validity of the first-order formulas (Theorem 10.34 in Section 10B.2). As the result, the theorem of $\mathcal{T}$ can be proved using $\mathbf{VTC}^0$ and the RFN of $\mathcal{F}$ (because $\overline{\mathbf{VTC}}^0$ is a conservative extension of $\mathbf{VTC}^0$).

First, we prove:

Theorem 10.44. (a) *Let $i \geq j \geq 1$. Then $\Sigma^B_j(\mathbf{V}^i)$, the Σ^B_j consequences of $\mathbf{V}^i$, can be axiomatized by the axioms of $\mathbf{VTC}^0$ together with Σ^q_j-$RFN_{\mathbf{G}^\star_i}$.*

(b) *For $i \geq 1$, all axioms of $\mathbf{V}^i$ can be prove using $\mathbf{VTC}^0$ and Σ_{i+1}-$RFN_{\mathbf{G}^\star_i}$.*

Proof. (a) Note that by Theorem 10.38 (b) $\mathbf{V}^i$ proves Σ^q_j-$RFN_{\mathbf{G}^\star_i}$. Therefore Σ^q_j-$RFN_{\mathbf{G}^\star_i}$ is a Σ^B_j consequence of $\mathbf{V}^i$. Since $j \geq 1$, all axioms of $\mathbf{VTC}^0$ are also in $\Sigma^B_j(\mathbf{V}^i)$. Thus it remains to show that every Σ_j^B consequence of $\mathbf{V}^i$ can be proved in $\mathbf{VTC}^0 + \Sigma_j^q\text{-}RFN_{\mathbf{G}_i^\star}$. We prove this for $i = j = 1$, since other cases are similar. Here we have to show that all Σ_1^B theorems of $\mathbf{V}^1$ are provable using $\mathbf{VTC}^0 + \Sigma_j^q\text{-}RFN_{\mathbf{G}_i^\star}$.

Suppose that φ is a Σ_1^B theorem of $\mathbf{V}^1$. Assume without loss of generality that it has a single free variable Z. Let $T_{\varphi(Z)}(n)$ be the $\mathbf{FTC}^0$ function as in Corollary 10.15 that provably in $\overline{\mathbf{VTC}}^0$ computes the encoding of $\varphi(Z)[n]$. Thus we have

$$T_{\varphi(Z)}(n) = \widehat{\varphi(Z)[n]}$$

and

$$\overline{\mathbf{VTC}}^0 \vdash \mathit{Fla}^{\Pi}_{\Sigma_1^q}(T_{\varphi(Z)}(n)) \tag{349}$$

Now by Exercise 10.19 there is an $\mathbf{FTC}^0$ function $F_{\varphi(Z)}(n)$ that provably in $\overline{\mathbf{VTC}}^0$ computes a $\mathbf{G}_1^\star$ proof of $\varphi(Z)[n]$. In other words, we have

$$\overline{\mathbf{VTC}}^0 \vdash \mathit{Prf}^{\Pi}_{\mathbf{G}_1^\star}(F_{\varphi(Z)}(n), T_{\varphi(Z)}(n)) \tag{350}$$

From (349) and (350), using $\Sigma_1^q\text{-}RFN_{\mathbf{G}_1^\star}$ we obtain

$$\forall Z(Z \models_{\Sigma_1^q} T_{\varphi(Z)}(n))$$

From this and Theorem 10.34 we obtain $\forall Z\varphi(Z)$.

(b) The proof is similar to the proof in part (a) of the fact that $\Sigma_j^B(\mathbf{V}^i)$ can be proved from $\mathbf{VTC}^0$ and $\Sigma_j^q\text{-}RFN_{\mathbf{G}_i^\star}$. Here note that all axioms of $\mathbf{V}^i$ are equivalent (in $\mathbf{V}^0$) to Σ_{i+1}^B formulas. ⊣

As a corollary, we obtain a finite axiomatization of $\mathbf{TV}^i$ as follows.

COROLLARY 10.45. *For $i \geq 0$, the theory $\mathbf{TV}^i$ can be axiomatized by the axioms of $\mathbf{VTC}^0$ together with $\Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_{i+1}^\star}$.*

PROOF. The Corollary follows from Theorem 10.44 and the fact that $\mathbf{TV}^i$ can be axiomatized by the Σ_{i+1}^B consequences of $\mathbf{V}^{i+1}$, because $\mathbf{TV}^i$ have Σ_{i+1}^B axioms and $\mathbf{V}^{i+1}$ is Σ_{i+1}^B-conservative over $\mathbf{TV}^i$ (Theorem 8.95). ⊣

We obtain an alternative proof for the finite axiomatizability of the theories $\mathbf{TV}^i$ (see Theorem 8.85):

COROLLARY 10.46. *For $i \geq 0$ the theories $\mathbf{TV}^i$ are finitely axiomatizable. For $i \geq j \geq 1$, the Σ_j^B consequences of $\mathbf{V}^i$ are finitely axiomatizable.*

PROOF. The conclusion follows from Corollary 10.45 and Theorem 10.44, and the fact that $\mathbf{VTC}^0$ is finitely axiomatizable. ⊣

In Theorem 10.44 above we have considered Σ_j^q-$RFN_{\mathbf{G}_i^\star}$ only for the values of j such that $1 \le j \le i$. Now we consider Σ_j^q-$RFN_{\mathbf{G}_i^\star}$ for $j > i$. It turns out that in this case Σ_j^q-$RFN_{\mathbf{G}_i^\star}$ is equivalent to Σ_j^q-$RFN_{\textit{cut-free } \mathbf{G}^\star}$, because any $\mathbf{G}_i^\star$ proof of a Σ_j^q formula can be transformed into a cut free $\mathbf{G}^\star$ proof of an equivalent Σ_j^q formula A'. This observation is due to Perron [70]. Here we need $\mathbf{VTC}^0$ to prove the equivalence, essentially because the transformation given in the proof below is computable in $\mathbf{TC}^0$ (while the equivalence between A and A' is provable in $\mathbf{V}^0$).

Theorem 10.47. *Let $i \ge 0$. The theory $\mathbf{VTC}^0$ proves that the following are all equivalent:*

$$\Sigma_{i+1}^q\text{-}RFN_{\textit{cut-free } \mathbf{G}^\star},\ \Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_0^\star},\ \ldots,\ \textit{and } \Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_i^\star}$$

Proof. Since *cut-free* $\mathbf{G}^\star$ is a subsystem of $\mathbf{G}_0^\star$, which in turn is a subsystem of $\mathbf{G}_1^\star$, etc., and since $\overline{\mathbf{VTC}}^0$ is a conservative extension of $\mathbf{VTC}^0$, it suffices to show that

$$\overline{\mathbf{VTC}}^0 + \Sigma_{i+1}^q\text{-}RFN_{\textit{cut-free } \mathbf{G}^\star} \vdash \Sigma_{i+1}^q\text{-}RFN_{\mathbf{G}_i^\star}$$

The idea is as follows. Let π be a $\mathbf{G}_i^\star$ proof of a Σ_{i+1}^q formula A. We will transform π into a cut free $\mathbf{G}^\star$ proof of a Σ_{i+1}^q formula A' of the form (352) below. Our transformation can be seen to be in $\mathbf{TC}^0$. So, formally, the transformed prove is provably in $\overline{\mathbf{VTC}}^0$ computable by some $\mathbf{FTC}^0$ function F. Then using Σ_j^q-$RFN_{\textit{cut-free } \mathbf{G}^\star}$ we have that A' is valid. Finally, since $\mathbf{V}^0$ proves that A and A' are equivalent we conclude that A is valid.

We will first transform π into a *cut-free* $\mathbf{G}^\star$ proof of the following sequent:

$$\text{(351)} \qquad \longrightarrow A, \exists(C_1 \wedge \neg C_1), \exists(C_2 \wedge \neg C_2), \ldots, \exists(C_k \wedge \neg C_k)$$

where C_t (for $1 \le t \le k$) are all cuts formulas in π, and $\exists(C_t \wedge \neg C_t)$ is the sentence obtained from $(C_t \wedge \neg C_t)$ by existentially quantifying all free variables. Then, by using the $\vee$-right we get a proof of

$$\text{(352)} \qquad A' =_{\text{def}} A \vee \bigvee_{1 \le t \le k} (\exists(C_t \wedge \neg C_t))$$

Notice that each C_i is a Σ_i^q formula, so A' is in Σ_{i+1}^q.

The derivation from (351) to

$$\longrightarrow A'$$

is obvious, so we will focus on the derivation of (351).

Let Δ denote the sequence

$$\exists(C_1 \wedge \neg C_1), \exists(C_2 \wedge \neg C_2), \ldots, \exists(C_k \wedge \neg C_k)$$

as in (351). We transform π as follows. First, add Δ to the succedent of every sequent in π. For each sequent $\mathcal{S}$ in π let $\mathcal{S}'$ be the result of this addition.

To obtain a legitimate derivation, note that if $\mathcal{S}$ is derived from $\mathcal{S}_1$ (and $\mathcal{S}_2$) by an inference of $\mathbf{G}$, then $\mathcal{S}'$ can be derived from $\mathcal{S}_1'$ (and $\mathcal{S}_2'$) by the same inference with possibly some applications of the exchange rule. In addition, each axiom $B \longrightarrow B$ now becomes $B \longrightarrow B, \Delta$ so we also add the following derivation (using weakening)

$$\frac{B \longrightarrow B}{\overline{B \longrightarrow B, \Delta}}$$

Thus, the result, called π_1, is a $\mathbf{G}_i^\star$ proof.

Next, consider an instance of the cut rule in π_1:

$$\frac{\Lambda \longrightarrow \Gamma, \Delta, C \qquad C, \Lambda \longrightarrow \Gamma, \Delta,}{\Lambda \longrightarrow \Gamma, \Delta,}$$

We insert the following derivation

$$\cfrac{\cfrac{\cfrac{\Lambda \longrightarrow \Gamma, \Delta, C \qquad \cfrac{C, \Lambda \longrightarrow \Gamma, \Delta}{\Lambda \longrightarrow \Gamma, \Delta, \neg C}}{\Lambda \longrightarrow \Gamma, \Delta, (C \wedge \neg C)}}{\overline{\Lambda \longrightarrow \Gamma, \Delta, \exists(C \wedge \neg C)}}}{\overline{\Lambda \longrightarrow \Gamma, \Delta}}$$

Here the bottom double line represents applications of exchange and contraction right. The double line above it represent a series of $\exists$-right.

It can be seen that we obtain a cut-free $\mathbf{G}^\star$ proof of (351) as desired. We briefly verify that the above transformation is computable in $\mathbf{TC}^0$. The fact that it can be formalized in $\mathbf{VTC}^0$ is straightforward.

For the transformation, first we need to identify all cuts and cut formulas in π (here we need $\mathbf{TC}^0$ circuits to recognize formulas). Once this has been done, it is easy to see that the cut free $\mathbf{G}^\star$ proof of (351) described above can be computed by some $\mathbf{TC}^0$ circuit (here we need the counting gates, e.g., to put $\exists(C_t \wedge \neg C_t)$ into the list Δ). The last step is to obtain a derivation of A' from (351), and this can also be computed by a $\mathbf{TC}^0$ circuit. ⊣

It follows from Theorem 10.44 (b) and Theorem 10.47 above that the axioms of $\mathbf{V}^i$ can be proved from

$$\mathbf{VTC}^0 + \Sigma_{i+1}^q\text{-}RFN_{\text{cut-free } \mathbf{G}^\star}$$

Here we will strengthen this by considering the following subclass of Σ_{i+1}^q.

Definition 10.48. *For $i \geq 0$, a quantified Boolean formula $A(\vec{p})$ is said to be $\mathbf{s}\Sigma^q_{i+1}$ (for* simple *Σ^q_{i+1}) if it has the form*

$$\bigvee_{1 \leq j \leq k} \exists x_1 \exists x_2 \ldots \exists x_m \; B_j(\vec{x}, \vec{p}) \tag{353}$$

where $B_1, B_2, \ldots, B_k$ are Π^q_i formulas.

Notice that $\mathbf{s}\Sigma^q_{i+1}$ is a subclass of Σ^q_{i+1}. Also, the property that a string X encodes a $\mathbf{s}\Sigma^q_{i+1}$ formula is Δ^B_1-definable in $\mathbf{VTC}^0$. We define the $\forall\Sigma^B_{i+1}$ sentence $\mathbf{s}\Sigma^q_{i+1}$-$RFN_{\mathcal{F}}$ in the same way as in Definition 10.35.

The next corollary can be proved by a similar proof as Theorem 10.47 above.

Corollary 10.49. *Let $i \geq 0$. The following sentences are all equivalent over $\mathbf{VTC}^0$:*

$$\mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\textit{cut-free } \mathbf{G}^\star},\ \mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\mathbf{G}^\star_0},\ \ldots,\ \textit{and } \mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\mathbf{G}^\star_i}$$

From Corollary 10.49 and Theorem 10.44 (b) we derive:

Corollary 10.50. *For $i \geq 1$ the axioms of $\mathbf{V}^i$ can be proved from*

$$\mathbf{VTC}^0 + \mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\textit{cut-free } \mathbf{G}^\star}$$

Proof. Note that all axioms of $\mathbf{V}^i$ are Σ^B_{i+1} formulas. In Theorem 10.44 (b) we have shown that all axioms of of $\mathbf{V}^i$ are provable from

$$\mathbf{VTC}^0 + \Sigma^q_{i+1}\text{-}RFN_{\mathbf{G}^\star_i}$$

Observe that all bounded $\mathcal{L}^2_A$ formulas translate into prenex QPC formulas, so by the same argument we can show that all axioms of $\mathbf{V}^i$ are provable from

$$\mathbf{VTC}^0 + \mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\mathbf{G}^\star_i}$$

Now the corollary follows from Corollary 10.49 above. $\dashv$

Although both $\mathbf{s}\Sigma^q_{i+1}$-$RFN_{\mathcal{F}}$ and Σ^q_{i+1}-$RFN_{\mathcal{F}}$ are $\forall\Sigma^B_{i+1}$ sentences, analyzing proofs of a $\mathbf{s}\Sigma^q_{i+1}$ formula is easier than proofs of a Σ^q_{i+1} formula. As a result, proving $\mathbf{s}\Sigma^q_{i+1}$-$RFN_{\mathcal{F}}$ is often simpler. Here we are able to prove a partial converse to Corollary 10.50 above.

Theorem 10.51. *For $i = 1$ or $i = 2$, the theory $\mathbf{V}^i$ can be axiomatized by the axioms of $\mathbf{VTC}^0$ together with any of the following axioms:*

$$\mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\textit{cut-free } \mathbf{G}^\star},\ \mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\mathbf{G}^\star_0},\ \ldots,\ \mathbf{s}\Sigma^q_{i+1}\text{-}RFN_{\mathbf{G}^\star_i}$$

To prove the theorem, note that by Corollaries 10.50 and 10.49 the main task here is to show that $\mathbf{V}^i$ proves the axiom $\mathbf{s}\Sigma^q_{i+1}$-$RFN_{\textit{cut-free } \mathbf{G}^\star}$. Consider, for example, a cut free $\mathbf{G}^\star$ proof π of a Σ^q_2 formula

$$A(\vec{p}) \equiv \exists \vec{x} B(\vec{p}, \vec{x})$$

where $B(\vec{p}, \vec{x})$ is a Π^q_1 formula. Suppose that we want to prove in $\mathbf{V}^1$ that $A(\vec{p})$ is valid. The idea of the proof below is based on the fact that the witnesses for $\vec{x}$ can be defined from the target formulas that introduce them in π. So we will transform π by eliminating the applications of the $\exists$ right rule in order to retain all target formulas in the final sequent. The cost we pay is that some applications of $\forall$ right are no longer legitimate because the eigenvariables are present in the target formulas retained in the bottom sequent. As a result, we will not apply the $\forall$ right rule altogether. Thus the resulting proof is a proof of a Σ_0^B formula A' and we will use A' to obtain the witnesses for $\vec{x}$.

Note that for if A is a Σ_{i+1}^q formula where $i \geq 3$ then there are at least two "levels" of witnesses (e.g., proving the correctness of the witnesses for A requires finding witnesses for the Σ_{i-1}^q subformulas of A). Therefore obtaining the witnesses for A from the Σ_0^B formula A' is more complicated. For the case $i = 2$ (hence A is Σ_3^q) we can preserve all Σ_1^q formulas in π during our transformation, so effectively there is only one "level" of witnesses and we can use the same proof as for the case $i = 1$.

PROOF OF THEOREM 10.51. By Corollaries 10.50 and 10.49 it suffices to show that the axiom $\mathbf{s}\Sigma_{i+1}^q\text{-}RFN_{\textit{cut-free}\ \mathbf{G}^\star}$ is provable in $\mathbf{V}^i$.

Proof for the case $i = 1$. Suppose that π is a cut-free $\mathbf{G}^\star$ proof of a $\mathbf{s}\Sigma_1^q$ formula $A(\vec{p})$. For simplicity, we will assume that in $A(\vec{p})$ no $\forall$ quantifier occurs inside the scope of any $\neg$. Furthermore, we will assume that the conjunction in (353) consists of just one conjunct. The proof in the general case is similar. In other words, assume that $A(\vec{p})$ has the form

$$\exists x_1 \exists x_2 \ldots \exists x_m\ B(\vec{x}, \vec{p})$$

where B is a Π_1^q formula that does not contain the $\exists$ quantifier.

Reasoning in $\mathbf{V}^1$. Notice that there is a polytime algorithm that transforms π into a free variable normal form proof (see Exercise 10.13). Therefore we can assume that π is in free variable normal form.

We will transform π into a cut-free proof where no quantifier occurs. Intuitively, the universally quantified variables in π will be replaced by the corresponding eigenvariables that introduce them, and the existentially quantified variables will be witnessed by the target formulas that introduce them.

In particular, the final sequent in the transformation is of the form

$$\longrightarrow B_1'(\vec{W}^1, \vec{q}^{\,1}, \vec{p}), B_2'(\vec{W}^2, \vec{p}^{\,2}, \vec{p}), \ldots, B_k'(\vec{W}^k, \vec{q}^{\,k}, \vec{p}) \tag{354}$$

where for each i ($1 \leq i \leq k$), $\vec{W}^i$ (resp. $\vec{q}^{\,i}$) are the target formulas (resp. eigenvariables) that introduce the existential (resp. universal) quantifiers in A, and B_i' is a Σ_0^q formula obtained from B by replacing the quantified variables by the corresponding target formulas or eigenvariables. Note that the eigenvariables $\vec{q}^{\,i}$ do not occur in $\vec{W}^i$.

Before describing the transformation, we will show how to use (354) to find a set of witnesses for A. Since the transformed proof is (still) a cut free proof and since $\mathbf{V}^1$ proves Σ^q_1-$RFN_{\mathbf{G}^\star_1}$, $\mathbf{V}^1$ also proves that the endsequent (354) of π' is valid.

Observe that we can order the eigenvariables in (354) such that

(355) $\quad \vec{W}^j$ contains a variables in $\vec{q}^{\,i}$ only if $j < i$.

Given a truth assignment to $\vec{p}$. Using Σ^B_1-**MAX** we can find the minimum index s such that there is a truth assignment τ to the eigenvariables $\vec{q}^{\,i}$ that falsifies all

$$B'_s(\vec{W}^s, \vec{q}^{\,s}, \vec{p}), B'_{s+1}(\vec{W}^{s+1}, \vec{p}^{\,s+1}, \vec{p}), \ldots, B'_k(\vec{W}^k, \vec{q}^{\,k}, \vec{p})$$

More precisely, s is minimum so that

$$\exists\tau\forall j \leq k\big(s \leq j \supset \neg(\tau \models^\Pi_0 B'_j)\big)$$

(The length of τ can be easily bounded.) Although the above formula is not really Σ^B_1, because the Σ^B_1 formula

$$\big(r \leq j \supset \neg(\tau \models^\Pi_0 B'_j)\big)$$

is in the scope of the number quantifier $\forall j \leq k$, using Σ^B_1-**REPL** (which is provable in $\mathbf{V}^1$) we can find an equivalent Σ^B_1 formula.

Since (354) is valid, $s > 1$. We show that under τ, $\vec{W}^{s-1}$ are witnesses for A. Assume for the sake of contradiction that $\vec{W}^{s-1}$ are not witnesses for A. Then there are values $\vec{q}$ such that

$$B'_{s-1}(\vec{W}^{s-1}, \vec{q}, \vec{p})$$

is false. Modify τ so that $\tau(\vec{q}^{\,s-1}) = \vec{q}$. By (355) this modification does not change the truth values of $B'_s, B'_{s+1}, \ldots, B'_k$. Hence the new truth assignment falsifies all

$$B'_{s-1}, B'_s, B'_{s+1}, \ldots, B'_k$$

This violates the minimality of s.

Transformation of π: Now we formally describe the transformation. Note that, because π is in free variable normal form, each variable is used as eigenvariable exactly once, and an eigenvariable only occurs in the branch of the proof ending with the sequent where it introduces the quantifier. We transform π inductively, starting with the axioms which are left unchanged. Consider a sequent $\mathcal{S}$ in π that is obtained by an inference rule. We will define the transformation $\mathcal{S}'$ of $\mathcal{S}$. In general, $\mathcal{S}'$ is obtained from $\mathcal{S}$ by replacing each ancestor of A of the form

(356) $\quad \exists x_\ell \exists x_{\ell+1} \ldots \exists x_m\ B(W_1, \ldots, W_{\ell-1}, x_\ell, x_{\ell+1}, \ldots, x_m, \vec{p})$

(where $1 \leq \ell \leq m$) by a list of Σ^q_0 formulas of the form

(357) $\quad B'(W_1, \ldots, W_{\ell-1}, W'_\ell, W'_{\ell+1}, \ldots, W'_m, \vec{q}, \vec{p})$

where $\vec{q}$ are eigenvariables. Moreover, Σ^q_0 formulas in $\mathcal{S}$ remain in $\mathcal{S}'$, and each other Π^q_1 formula C is replaced by a Σ^q_0 formula C' that is obtained from C by replacing the universally quantified variables by distinct eigenvariables. Our result is also a cut-free tree-like proof. Consider the following cases:

Case I: $\mathcal{S}$ is obtained by an $\exists$-right (here the principal formula must be of the form (356)) or a $\forall$-right from a sequent $\mathcal{S}_1$. We simply take $\mathcal{S}' = \mathcal{S}_1'$.

Case II: $\mathcal{S}$ is obtained by contraction-right:

$$\frac{\mathcal{S}_1}{\mathcal{S}} = \frac{\Lambda \longrightarrow \Gamma, C, C}{\Lambda \longrightarrow \Gamma, C}$$

There are three subcases. First, if C is a quantifier free formula, then $\mathcal{S}_1'$ has the form

$$\Lambda \longrightarrow \Gamma', C, C$$

So define

$$\mathcal{S}' = \Lambda \longrightarrow \Gamma', C$$

Otherwise, if C is a Π_1^B formula, then $\mathcal{S}_1'$ has the form

$$\Lambda \longrightarrow \Gamma', C_1', C_2'$$

where C_1' and C_2' are obtained from C by replacing the universally quantifiers by the corresponding eigenvariables. The proof of $\mathcal{S}'$ is tree-like and we can rename the eigenvariables so that C_1' and C_2' are identical. Then define

$$\mathcal{S}' = \Lambda \longrightarrow \Gamma', C_1'$$

Finally, C is an ancestor of A and has the form (356):

$$\exists x_\ell \ldots \exists x_m B(W_1, \ldots, W_{\ell-1}, x_\ell, \ldots, x_m, \vec{p})$$

Each copy of C in $\mathcal{S}_1$ is transformed into a list of formulas of the form (357). We simply take $\mathcal{S}' = \mathcal{S}_1'$.

Case III: $\mathcal{S}$ is obtained by $\wedge$-right:

$$\frac{\mathcal{S}_1 \qquad \mathcal{S}_2}{\mathcal{S}} = \frac{\Lambda \longrightarrow \Gamma, C_1 \qquad \Lambda \longrightarrow \Gamma, C_2}{\Lambda \longrightarrow \Gamma, C_1 \wedge C_2}$$

Here C_1, C_2 are Π_1^q. Each C_i is transformed into C_i', and we take

$$(C_1 \wedge C_2)' = C_1' \wedge C_2'$$

Notice that there are implicit contractions for formulas in Γ here, so the transformations of formulas in Γ in $\mathcal{S}'$ is defined in the same way as in Case II.

Case IV: $\mathcal{S}$ is obtained by any other rule. Each other rule is either similar to one discussed above, or is straightforward.

Notice that for each case, $\mathcal{S}'$ is easily obtained from $\mathcal{S}_1'$ (and $\mathcal{S}_2'$). Thus, we get a cut-free proof π' of a sequent of the form (354):

$$\longrightarrow B_1'(\vec{W}^1, \vec{q}^{\,1}, \vec{p}), B_2'(\vec{W}^2, \vec{p}^{\,2}, \vec{p}), \ldots, B_k'(\vec{W}^k, \vec{q}^{\,k}, \vec{p})$$

This concludes the description of the transformation.

Proof for the case $i = 2$. Here A is in Σ_3^q. We transform the proof π as before except now we keep all applications of the $\exists$ right in π that produce Σ_1^q formula. The result is a sequent as in (354) where now B'_j are Σ_1^q formulas. The witnesses for A can be computed as before but using Σ_2^B-**MAX**. $\dashv$

In Section 10C.2 we show that the relation $(Z \models_0 X)$ is Δ_1^B-definable in $\mathbf{VNC}^1$. It has been shown [61, Section 6.2] that the Σ_1^q Witnessing Problem (see Section 10B.6) for $\mathbf{G}_0^\star$ and $\mathbf{G}_0$ are both complete for $\mathbf{NC}^1$. (We proved in Theorem 7.43 that $\mathbf{G}_0^\star$ and $\mathbf{G}_0$ are p-equivalent for proving prenex Σ_1^q formulas.) Using this facts we can show that $\mathbf{s}\Sigma_1^q$-$RFN_{\mathbf{G}_0^\star}$ and $\mathbf{s}\Sigma_1^q$-$RFN_{\mathbf{G}_0}$ are both provable in $\mathbf{VNC}^1$. In Section 10C.1 we will prove the propositional translation theorem for $\mathbf{VNC}^1$ (Theorem 10.58). Thus it can be shown that $\mathbf{VNC}^1$ can be axiomatized by $\mathbf{VTC}^0$ together with either $\mathbf{p}\Sigma_1^q$-$RFN_{\mathbf{G}_0^\star}$ or $\mathbf{p}\Sigma_1^q$-$RFN_{\mathbf{G}_0}$ (here $\mathbf{p}\Sigma_1^q$ denotes prenex Σ_1^q).

10B.5. Proving p-simulations using RFN. In this section we will show how to use the Propositional Translation Theorems (e.g., Theorems 7.57, 10.23) to prove p-simulations between proof systems. Informally, the result is as follows. Suppose that $\mathcal{G}$ (such as $\mathbf{G}_1^\star$) is a proof system associated with a theory $\mathcal{T}$ (such as $\mathbf{V}^1$) (where the association is by propositional translation). Then any proof system $\mathcal{F}$ (such as **eFrege**) which is definable in $\mathcal{T}$ and whose RFN is provable in $\mathcal{T}$ is p-simulated by $\mathcal{G}$ (see the precise statement in Theorem 10.53 below).

Intuitively, the reason why $\mathcal{G}$ p-simulates such $\mathcal{F}$ is as follows. Because $\mathcal{T}$ proves the soundness (i.e., the RFN) of $\mathcal{F}$, and $\mathcal{G}$ is a nonuniform version of $\mathcal{T}$ (by the Propositional Translation Theorem), there are short $\mathcal{G}$ derivation of the fact that $\mathcal{F}$ is sound. Formally this is an $\mathcal{G}$ derivation of the translation of the RFN for $\mathcal{F}$, where the formula

$$A(p_0^Z, p_1^Z, \dots, p_{n-2}^Z)$$

has been transformed into

$$(Z \models \widehat{A})[n+1]$$

Now, given a proof of $\mathcal{F}$ of A we can derive a short $\mathcal{G}$ derivation of A using the above $\mathcal{G}$ derivation and the derivations from Theorems 10.30 and 10.32. (we will also need to verify that these $\mathcal{G}$ derivations are computable by polytime functions.)

Precise statements for our proof systems and theories are in Theorem 10.53. First, we need to proves in $\mathbf{G}_i^\star$ the "honesty" of our encoding of the RFN. Here we translate the RFN for a system $\mathcal{F}$ by treating the variables π, X and Z (Definition 10.35) as free variables and introducing the bits

$$p_0^\pi, p_0^X, p_0^Z, \dots$$

as usual.

Recall Definition 10.8 that for some constant values of X_0 and π_0 of length n and m respectively, we use

$$\Sigma_i^q\text{-}RFN_{\mathcal{F}}(\pi_0, X_0, Z)[n, m, k] \tag{358}$$

to denote the result of substituting the values ($\top$ or $\bot$) of bits $X_0(t)$, $\pi_0(t)$ for the variables p_t^X and p_t^π in the propositional translation

$$\Sigma_i^q\text{-}RFN_{\mathcal{F}}(\pi, X, Z)[n, m, k]$$

(Thus the only free variables in (358) are $p_0^Z, p_1^Z, \dots$.) For a Σ_i^q formulas X_0, note that the translation (358) is

$$(Fla^{\Pi}_{\Sigma_i^q}(X_0) \wedge Prf^{\Pi}_{\mathcal{F}}(\pi_0, X_0))[n, m] \supset (Z \models_{\Sigma_i^q} X_0)[m, k]$$

For the proof of the theorem below, it is helpful to review Theorems 10.30 and 10.32 and their proofs.

THEOREM 10.52. *Let $i \geq 1$ and $\mathcal{F}$ be a proof system with defining formulas $Prf^{\Sigma}_{\mathcal{F}}$ and $Prf^{\Pi}_{\mathcal{F}}$ as in* (312) *and* (313) *(on page 342). Then there is a polytime algorithm that computes a $\mathbf{G}_i^\star$ proof of the sequent below for each Σ_i^q formula A_0 and $\mathcal{F}$-proof π_0 of A_0:*

$$\Sigma_i^q\text{-}RFN_{\mathcal{F}}(\pi_0, \widehat{A_0}, Z)[n, m, k] \longrightarrow A_0(\overrightarrow{p^Z})$$

PROOF. The desired sequent is obtain from the sequent (338) in Theorem 10.30 and the following sequent by a Σ_i^q cut:

$$\Sigma_i^q\text{-}RFN_{\mathcal{F}}(\pi_0, \widehat{A_0}, Z)[n, m, k] \longrightarrow (Z \models_{\Sigma_i^q} X)[m, k]$$

In turns, the sequent above can be derived from

$$\longrightarrow \big(Fla^{\Pi}(X_0) \wedge Prf^{\Pi}_{\mathcal{F}}(\pi_0, X_0)\big)[n, m]$$

A $\mathbf{G}_0^\star$ proof of this sequent can be computed in polytime as shown in Lemma 10.9. ⊣

THEOREM 10.53. *Let $i \geq j \geq 1$.*

(a) *Suppose that $\mathcal{F}$ is a proof system such that*

$$\mathbf{V}^i \vdash \Sigma_j^q\text{-}RFN_{\mathcal{F}}$$

Then $\mathbf{G}_i^\star$ p-simulates $\mathcal{F}$ w.r.t. Σ_i^q formulas.

(b) *The same is true for $\mathbf{TV}^i$ and $\mathbf{G}_i$ in place of $\mathbf{V}^i$ and $\mathbf{G}_i^\star$.*

PROOF. (a) Let $A_0(\vec{p})$ be a Σ_j^q formula and π_0 be an $\mathcal{F}$-proof of $A_0(\vec{p})$. As before, let $\widehat{A_0}$ denote the string encoding of $A_0(\vec{p})$. From the hypothesis and the Propositional Translation Theorem for $\mathbf{V}^i$ (Theorem 7.57) (see also Exercise 10.19), there are polytime computable $\mathbf{G}_i^\star$ proofs of the translations

$$\Sigma_i^q\text{-}RFN_{\mathcal{F}}(\pi_0, \widehat{A_0}, Z)[n, m, k]$$

Now using the $\mathbf{G}_j^\star$ proof from Theorem 10.52 we obtain a $\mathbf{G}_i^\star$ proof of $A_0(\overrightarrow{p^Z})$.

Part (b) is proved similarly. ⊣

Exercise 10.54. *Let $\mathcal{F}$ be a proof system for (unquantified) propositional formulas. Suppose that $RFN_\mathcal{F}$ (i.e., 0-$RFN_\mathcal{F}$, see Definition 10.35) is provable in $\mathbf{TV}^0$. Show that* $\mathbf{ePK}$ *p-simulates $\mathcal{F}$.*

We obtain as corollaries some results proved earlier in Chapter 7 (Corollary 7.47 and Theorem 7.54).

Corollary 10.55. *For $i \geq 1$, $\mathbf{G}_{i+1}^\star$ and $\mathbf{G}_i$ are p-equivalent when proving Σ_i^q formulas.* $\mathbf{ePK}$ *and $\mathbf{G}_1^\star$ are p-equivalent for proving prenex Σ_1^q formulas.*

10B.6. The witnessing problems for G. Recall the notion of search problems defined in Section 8E (see Definitions 8.66 and 8.75). Recall also the witnessing problem given in Theorem 7.51. In general, the witnessing problems for (subsystems of) $\mathbf{G}$ are search problems that are motivated by the following observation. Let $i \in \mathbb{N}$, $i \geq 1$, and consider a Σ_i^q tautology $A(\vec{p})$ of the form

$$A(\vec{p}) \equiv \exists \vec{x} B(\vec{x}, \vec{p})$$

where B is a Π_{i-1}^q formula (so A is indeed a $\exists\Pi_{i-1}^q$ formula). Given A and the values for $\vec{p}$ we wish to find a truth assignment for the existentially quantified variables $\vec{x}$ that satisfies $B(\vec{x}, \vec{p})$. Note that this problem is polytime complete for $\mathbf{FP}^{\Sigma_i^P}$. However, a given $\mathbf{G}$-proof π of $A(\vec{p})$ may help us find $\vec{x}$, so it becomes interesting to study the problems when different proofs π are given.

Formally, the problems are defined as follows.

Definition 10.56 (Witnessing Problem). *For a quantified propositional proof system $\mathcal{F}$ and $1 \leq i \in \mathbb{N}$, the Σ_i^q* Witnessing Problem for *$\mathcal{F}$, denoted by Σ_i^q-$WIT_\mathcal{F}$, is, given an $\mathcal{F}$-proof π of a Σ_i^q formula $A(\vec{p})$ of the form*

$$A(\vec{p}) \equiv \exists \vec{x} B(\vec{x}, \vec{p})$$

where B is a Π_{i-1}^q formula, and a truth assignment to $\vec{p}$, find a truth assignment for $\vec{x}$ that satisfies $B(\vec{x}, \vec{p})$.

Not surprisingly, the Witnessing Problems for $\mathbf{G}_i$ and $\mathbf{G}_i^\star$ are closely related to the classes of definable search problems in the associated theories. For the next theorem it is useful to refer to the the summary table on page 239.

Recall the notion of many-one reduction between search problems (Definition 8.67). For the next theorem we use the notion of $\mathbf{TC}^0$ many-one reduction between search problems. This is defined just as in Definition 8.67 with the exception that the functions $\vec{f}, \vec{F}, G$ are now in $\mathbf{FTC}^0$ (as opposed to $\mathbf{FAC}^0$). The reason that we need $\mathbf{TC}^0$

reductions here is basically because our translation functions (such as in Exercise 10.19) are $\mathbf{TC}^0$ functions.

Theorem 10.57. (a) *For* $i \geq 1$, Σ_i^q-$WIT_{\mathbf{G}_i^\star}$ *is complete for* $\mathbf{FP}^{\Sigma_{i-1}^P}$

(b) Σ_i^q-$WIT_{\mathbf{G}_i}$ *is complete for* $\mathbf{CC}(\mathbf{PLS})^{\Sigma_{i-1}^P}$.

(c) *For* $i \in \{1,2\}$, Σ_{i+1}^q-$WIT_{\mathbf{G}_i^\star}$ *is complete for* $\mathbf{FP}^{\Sigma_i^P}[wit, \mathcal{O}(\log n)]$.

Proof. (a) First we show that Σ_i^q-$WIT_{\mathbf{G}_i^\star}$ is in $\mathbf{FP}^{\Sigma_{i-1}^P}$. Consider the case $i = 1$. Here the Witnessing Theorem for $\mathbf{G}_1^\star$ (Theorem 7.51) already shows that Σ_1^q-$WIT_{\mathbf{G}_1^\star}$ is in $\mathbf{P}$. The case where $i > 1$ is similar. In fact, we have pointed out in the proof of Theorem 10.38 (b) that by analyzing π, a witnessing function can be defined in $\mathbf{V}^i$ by a Σ_i^B formula.

Now we show that Σ_i^q-$WIT_{\mathbf{G}_i^\star}$ is hard for $\mathbf{FP}^{\Sigma_{i-1}^P}$. Thus let $Q(\vec{x}, \vec{X})$ be a search problem in $\mathbf{FP}^{\Sigma_{i-1}^P}$ with graph $R(\vec{x}, \vec{X}, Z)$. We show that Q is reducible to Σ_i^q-$WIT_{\mathbf{G}_i^\star}$. By Theorem 8.94 Q is Σ_i^B-definable in $\mathbf{V}^i$ by a Σ_i^B formula $\varphi(\vec{x}, \vec{X}, Z)$, i.e.,

$$\varphi(\vec{x}, \vec{X}, Z) \supset R(\vec{x}, \vec{X}, Z)$$

and

$$\mathbf{V}^i \vdash \exists Z\, \varphi(\vec{x}, \vec{X}, Z)$$

By the $\mathbf{V}^i$ Translation Theorem (Theorem 7.57) the Σ_i^B theorem $\exists Z\, \varphi(\vec{x}, \vec{X}, Z)$ of $\mathbf{V}^i$ translates into a family of tautologies that have polynomial-size $\mathbf{G}_i^\star$ proofs. In fact, by Exercise 10.19 there is a $\mathbf{TC}^0$ function $F_\varphi(\vec{x}, \vec{X})$ that provably in $\overline{\mathbf{VTC}}^0$ computes a $\mathbf{G}_i^\star$ proof of the translation of φ. Thus, given $(\vec{x}, \vec{X})$ a value for Z that satisfies $\varphi(\vec{x}, \vec{X}, Z)$ can be easily obtained from the solution of the witnessing problem given by $F_\varphi(\vec{x}, \vec{X})$ and $(\vec{x}, \vec{X})$.

(b) The fact that Σ_i^q-$WIT_{\mathbf{G}_i}$ is complete for $\mathbf{CC}(\mathbf{PLS})^{\Sigma_{i-1}^P}$ is proved similarly using Corollary 10.41 (b), the $\mathbf{TV}^i$ Translation Theorem 10.23 and Theorem 8.96.

(c) Similar to part (a). Here Σ_{i+1}^q-$WIT_{\mathbf{G}_i^\star}$ is in $\mathbf{FP}^{\Sigma_i^P}[wit, \mathcal{O}(\log n)]$ because $\mathbf{s}\Sigma_{i+1}^q$-$RFN_{\mathbf{G}_i^\star}$ is provable in $\mathbf{V}^i$ (Theorem 10.51) and Σ_{i+1}^B-definable search problems in $\mathbf{V}^i$ are in $\mathbf{FP}^{\Sigma_i^P}[wit, \mathcal{O}(\log n)]$ (Theorem 8.99). The fact that Σ_{i+1}^q-$WIT_{\mathbf{G}_i^\star}$ is hard for $\mathbf{FP}^{\Sigma_i^P}[wit, \mathcal{O}(\log n)]$ also follows from Theorem 8.99 and the $\mathbf{V}^i$ Translation Theorem 7.57 as in (a). $\dashv$

10C. $\mathbf{VNC}^1$ and $\mathbf{G}_0^\star$

Recall (Section 9E.2) that the theory $\mathbf{VNC}^1$ is axiomatized by the axioms of $\mathbf{V}^0$ together with the axiom *MFV* (for *monotone formula* *value*) that asserts the existence of a polytime algorithm (Y) for evaluating a balanced formula (encoded by (a, E, G)):

$$\forall a \forall G \forall I \exists Y \leq 2a \delta_{MFV}(a, G, I, Y)$$

where

$$(359) \quad \delta_{MFV}(a, G, I, Y) \equiv \forall x < a\big((Y(x+a) \leftrightarrow I(x)) \ \wedge \\ 0 < x \supset \big(Y(x) \ \leftrightarrow \ \big((G(x) \wedge (Y(2x) \wedge Y(2x+1))) \vee \\ (\neg G(x) \wedge (Y(2x) \vee Y(2x+1)))\big)\big)\big)$$

In this section we will associate $\mathbf{VNC}^1$ with $\mathbf{G}_0^\star$ in the same way that the theories $\mathbf{V}^i$ are associated with the proof systems $\mathbf{G}_i^\star$ (where $i \geq 1$). It follows that Σ_0^B theorems of $\mathbf{VNC}^1$ translates into families of propositional tautologies that have $\mathbf{PK}$ proofs which are provably in $\overline{\mathbf{VTC}}^0$ computable by $\mathbf{FTC}^0$ functions (recall Section 2A.1 for $\mathbf{PK}$).

We mentioned in Section 10B.4 that $\mathbf{VNC}^1$ proves the RFN for $\mathbf{G}_0^\star$ and $\mathbf{G}_0$ for prenex Σ_1^q formulas. For this we need to show that the relation $(Z \models_0 X)$ (that the truth assignment Z satisfies a formula X, see Section 10B.1) is Δ_1^B-definable in $\mathbf{VNC}^1$. For this we will formalize in $\mathbf{VNC}^1$ an $\mathbf{NC}^1$ (or equivalently **ALogTime**) algorithm that computes the Boolean Sentence Value Problem (BSVP). Recall (page 298) that this is the problem of computing the value of a given Boolean sentence. In general, the given sentence is not necessarily balanced, so we cannot use the algorithm coded by the axiom MFV straightforwardly. In fact, the existences of $\mathbf{NC}^1$ circuits (i.e., **ALogTime** algorithms) that evaluate Boolean sentences are nontrivial. In Section 10C.2 we will formalize one such algorithm in $\mathbf{VNC}^1$ in order to show that prove that $(Z \models_0 X)$ is Δ_1^B-definable in $\mathbf{VNC}^1$. The algorithm is by Buss [17].

10C.1. Propositional translation for $\mathbf{VNC}^1$. Recall (Section 7D) that $\mathbf{G}_0^\star$ is the subsystem of $\mathbf{G}^\star$ where all cut formulas are quantifier-free. Note also that $\mathbf{G}_0^\star$ p-equivalent to $\mathbf{G}_0$ with respect to prenex Σ_1^q formulas (Theorem 7.43). Recall also Definition 10.11. Our goal in this section is to prove the following theorem.

THEOREM 10.58 (Translation Theorem for $\mathbf{VNC}^1$). *Let $\varphi(\vec{x}, \vec{X})$ is a bounded theorem of $\mathbf{VNC}^1$. Then there is an $\mathbf{FTC}^0$ function $F_\varphi(\vec{m}, \vec{n})$ that provably in $\overline{\mathbf{VTC}}^0$ computes a $\mathbf{G}_0^\star$ proof of the propositional formulas $\|\varphi(\vec{x}, \vec{X})\|$ for all $\vec{m}$, $\vec{n}$ in $\mathbb{N}$.*

Our translation of an anchored $\mathbf{LK}^2$-$\mathbf{VNC}^1$ proof (in free variable normal form) of a bounded theorem of $\mathbf{VNC}^1$ follows the translation of $\mathbf{LK}^2$-$\mathbf{V}^0$ proofs discussed in Section 7E.1. Here the new type of cut formulas are instances of the formula $\exists Y \delta_{MFV}(a, G, I, Y)$ (see (359)). Note that the length $|Y|$ in MFV is bounded by $2a$. To make the translation easier we will fix $|Y|$. Thus we will use another axiom

$\delta'_{MFV}(a, G, I, Y)$ defined below, where now the length $|Y|$ of Y is required to be exactly $2a+1$. Informally, this is easily obtained by adding a fixed leading bit (bit $2a$) to the string Y. Note also that in MFV the bit $Y(0)$ of Y is not fixed. In $\delta'_{MFV}(a, G, I, Y)$ we will simply fix it to $\top$.

The fact that the new axiom is equivalent to MFV over $\mathbf{V}^0$ is easy and is left as an exercise.

EXERCISE 10.59. *Let* $\delta'_{MFV}(a, G, I, Y)$ *denote*

$$(360)\quad |Y| = 2a + 1 \wedge Y(0) \wedge \\ \forall x < a\big((Y(x + a) \leftrightarrow I(x)) \wedge 0 < x \supset \big(Y(x) \leftrightarrow \\ \big((G(x) \wedge Y(2x) \wedge Y(2x + 1)) \vee (\neg G(x) \wedge (Y(2x) \vee Y(2x + 1)))\big)\big)\big)$$

Then $\mathbf{VNC}^1$ *can be axiomatized by* $\mathbf{V}^0$ *together with*

$$MFV' \equiv \forall a \forall G \forall I \exists Y\, \delta'_{MFV}(a, G, I, Y)$$

As a result, every theorem of $\mathbf{VNC}^1$ has an anchored $\mathbf{LK}^2$ proof where cut formulas are instances of the axioms of $\mathbf{V}^0$ or the axiom MFV' above. To prove the Theorem 10.58 we will first translate $\mathbf{LK}^2$ proofs of this type, and then argue that the translation is indeed provably computable in $\overline{\mathbf{VTC}}^0$.

Recall the proof of Theorem 7.61. Below we will also use the notions such as *comprehension variables* from there. The idea of the translation is to extend the translation of anchored $\mathbf{LK}^2$-$\mathbf{V}^0$ proofs as described in Section 7E.1. Here we have to consider in addition instances of the axiom MFV' above. Like the translations of the Σ_0^B-**COMP** cut formulas, here the translations of the cut MFV' formulas will be tautologies that have **PK** proofs which are provably in $\overline{\mathbf{VTC}}^0$ computable.

PROOF OF THEOREM 10.58. By Exercise 10.59 above, there is an anchored $\mathbf{LK}^2$-proof π of φ where all cut formulas are instances of the axioms of $\mathbf{V}^0$ or instances of the axiom MFV'. In addition, we can assume that π is in free variable normal form.

The translations of cut Σ_0^B-**COMP** formulas from the proof of Theorem 7.61 can be extended here easily. So we will focus on the instances of the cut MFV' axiom. Similar to the notion of comprehension variables, we have:

Notation. A free string variable γ in π is called an *MFV variable* if it is used as the eigenvariable for the string $\exists$ left rule whose principal formula is an ancestor of a cut formula of the form $\exists Y\, \delta'_{MFV}(t, \alpha, \beta, Y)$. In this case, we also say that (α, β, t) is the *defining triple* of γ.

For example, consider an instance of the string $\exists$ left rule:

$$(361)\qquad \frac{\mathcal{S}_1}{\mathcal{S}_2} = \frac{\delta'_{MFV}(t, \alpha, \beta, \gamma), \Gamma \longrightarrow \Delta}{\exists Y \delta'_{MFV}(t, \alpha, \beta, Y), \Gamma \longrightarrow \Delta}$$

Suppose that the formula $\exists Y \delta'_{MFV}(t, \alpha, \beta, Y)$ in $\mathcal{S}_2$ is an ancestor of a cut formula. Then γ is an MFV variable.

In our translation below, if two MFV variables have the same defining triple, they will have identical translations.

Next, we extend the definition of the dependence relation defined in the proof of Theorem 7.61 to include MFV variables.

Notation. We say that an MFV variable γ depends on a variable β (or b) if β (or b) occurs in the subproof of π that ends in a string $\exists$-left as (361) above.

The *dependence degree* of a variable is defined as before by taking into account that now there are also MFV variables. Formally, all non-comprehension variables have dependence degree 0. The dependence degree of a comprehension variable (resp. an MFV variable) γ is one plus the maximum dependence degree of all variables occurring in its defining pair (resp. defining triple).

The formulas in π are translated in stages just as described in the proof of Theorem 7.61. Here the only new cases to be handled are (i) bits $\gamma(s)$ of an MFV variable γ, and (ii) instances of $\exists Y \delta'_{MFV}(a, \alpha, \beta, Y)$ which are cut (and all their descendants).

For (i), let $\vec{n}$ be the list of lengths/values of all free variables that γ depends on, and let (α, β, t) be the defining triple of γ. We will be interested only in the translations where the length of γ is exactly $(2t + 1)$. So we will denote the translation simply by $\gamma(s)[\vec{n}]$; it is defined by (reverse) induction on $val(s)$ as follows.

Let $m = val(t)$. First, suppose that $m \le val(s) < 2m$, then

$$\gamma(s)[\vec{n}] =_{\mathrm{def}} \beta(r)[\vec{n}]$$

where r is the numeral $\underline{val(s) - m}$. Next, suppose that $1 \le val(s) < m$, then

$$\gamma(s)[\vec{n}] =_{\mathrm{def}} (A \wedge (B_0 \wedge B_1)) \vee (A \wedge (B_0 \vee B_1)) \tag{362}$$

where $A \equiv \alpha(s)[\vec{n}]$, $B_0 \equiv \gamma(2s)[\vec{n}]$, and $B_1 \equiv \gamma(2s+1)[\vec{n}]$. Finally,

$$\gamma(s)[\vec{n}] =_{\mathrm{def}} \begin{cases} \bot & \text{if } val(s) > 2m \\ \top & \text{if } val(s) = 2m \vee val(s) = 0 \end{cases}$$

For (ii), we will show that the translation of instances of MFV' are tautologies. Therefore we do not need to translate sequents that are on the right branches of a cut MFV' instance (these sequents are ancestors of the top right sequent in an MFV' cuts, i.e., the sequent that contains MFV' in the succedent). For the remaining sequents in the proof, all MFV' instances that are ancestors of a cut MFV' formula are translated into the empty formula.

We show that the translations above are provably in $\overline{\mathbf{VTC}}^0$ computable.

LEMMA 10.60. *For each $\mathcal{L}_A^2$ formula $\varphi(\vec{x}, \vec{X})$ in π there is an* $\mathbf{FTC}^0$ *function $F_{\varphi,\pi}(\vec{k}, \vec{n})$ that provably in $\overline{\mathbf{VTC}}^0$ computes the translation $\varphi(\vec{x}, \vec{X})[\vec{k}; \vec{n}]$ as described above.*

PROOF SKETCH. Following our translation of formulas in π, $F_{\varphi,\pi}$ will be defined in stages: for $i \geq 0$, in stage i we define the functions for all formulas φ that contains some variables of dependence degree i but none of higher degree.

In each stage the construction is by structural induction on the formula φ. Stage 0 is exactly the same as in Exercise 10.17; and in general, for each stage i (where $i \geq 1$) except for the base case of the formulas $\gamma(s)$ for MFV variables γ, the arguments are the same as in Exercise 10.17.

So now consider stage i where $i \geq 1$. Let γ be an MFV variable of dependence degree i, and let (α, β, t) be the defining triple for γ. We need to define the function $F_{\gamma(s),\pi}$. As before, let $m = \mathit{val}(t)$. Then note that the length of γ is understood to be $2m + 1$.

Let $b = \mathit{val}(s)$. If $b = 0$ or $b \geq m$ then by definition $\gamma(s)[\vec{n}]$ is a constant $\top$ or $\bot$ or is a function that has been defined in the previous stage. So consider the case where $1 \leq b < m$.

By the definition (362), $\gamma(s)[\vec{n}]$ can be seen as a binary tree whose leaves are labeled with $\alpha(s)[\vec{n}]$, $\beta(s)[\vec{n}]$ and $\gamma(2s)[\vec{n}]$, $\gamma(2s+1)[\vec{n}]$. Intuitively we will expand this tree at the leaves $\gamma(2s)[\vec{n}]$, $\gamma(2s+1)[\vec{n}]$ until all leaves are labeled with either $\alpha(r)[\vec{n}]$ or $\beta(r)[\vec{n}]$ for some r.

The depth of the final tree can be shown to be computable by some $\mathbf{AC}^0$ functions (recall Section 3C.3, for example, that the function $\log(x)$ is in $\mathbf{AC}^0$, where $\log(x)$ is the length of the binary representation of x). Additionally, the labels of the nodes on all paths from the root to the leaves of the tree can be identified by $\mathbf{AC}^0$ functions. From these facts, using the counting gates we can compute the string that concatenates all labels on the leaves with parentheses properly inserted. $\dashv$

Now we verify that for every sequent $\mathcal{S}$ in π that are not on the right branches of the cut Σ_0^B-**COMP** or cut MFV' instances, there are provably in $\mathbf{VTC}^0$ computable $\mathbf{G}_0^\star$ proofs of the translation of $\mathcal{S}$. The proof is by induction on the sequent $\mathcal{S}$. Except for the case of the string $\exists$-left that introduces a cut MFV' instance, all cases are the same as in the proof of Theorem 10.16.

So consider an instance of the string $\exists$-left that introduces a cut MFV' as in (361). Consider the interesting case where the translation of $\mathcal{S}_1$ is not simplified to an axiom, and thus has the form

$$\mathcal{S}_1[\vec{n}] = \qquad \delta'_{MFV}(t, \alpha, \beta, \gamma)[\vec{n}], \Gamma' \longrightarrow \Delta'$$

Then $\mathcal{S}$ translates into

$$\mathcal{S}[\vec{n}] = \qquad \Gamma' \longrightarrow \Delta'$$

In order to obtain $\mathcal{S}[\vec{n}]$ from $\mathcal{S}_1[\vec{n}]$, we need to derive the tautology $\delta'_{MFV}(t,\alpha,\beta,\gamma)[\vec{n}]$ and then apply the cut rule. Here note that $\delta'_{MFV}(t,\alpha,\beta,\gamma)[\vec{n}]$ is just a conjunction of the form

$$\bigwedge(B_i \leftrightarrow B_i)$$

where B_i is the translation of $\gamma(s)$ when $val(s) = i$, for $1 \le i < 2val(t)$. Hence $\delta'_{MFV}(t,\alpha,\beta,\gamma)[\vec{n}]$ can be easily derived from the axioms

$$B_i \longrightarrow B_i$$

Given the $\mathbf{FTC}^0$ functions that compute the translations of formulas in $\mathcal{S}_1$, it is straightforward to obtain an $\mathbf{FTC}^0$ function that computes the above derivation of $\mathcal{S}[\vec{n}]$ from $\mathcal{S}_1[\vec{n}]$. By the same arguments as in the proof of Theorem 10.16, it follows that the $\mathbf{G}_0^\star$ proofs can be provably in $\overline{\mathbf{VTC}}^0$ computed by some $\mathbf{FTC}^0$ functions. ⊣

The next corollary follows easily.

COROLLARY 10.61. *For any Σ_0^B theorem φ of $\mathbf{VNC}^1$, there is an $\mathbf{FTC}^0$ function that provably in $\overline{\mathbf{VTC}}^0$ computes $\mathbf{PK}^\star$ proofs of the family of tautologies $\|\varphi\|$.*

10C.2. The Boolean Sentence Value Problem. Recall (page 298) that the Boolean Sentence Value Problem (BSVP) is to determine the truth value of a Boolean sentence. Here the sentence is given as a string over the alphabet:

$$\{\top, \bot, (,), \wedge, \vee, \neg\} \tag{363}$$

The sentence is viewed as a tree whose leaves are labeled with constants $\top, \bot$ and whose inner nodes are labeled with connectives. Note that when the tree representing a sentence A is a balanced binary tree then it is straightforward to show that there is an **ALogTime** algorithm that computes the value of A. In fact, by using the axiom MFV (Definition 9.92) we can easily formalize in $\mathbf{VNC}^1$ such an algorithm. However, designing an **ALogTime** algorithm for general tree structure is more difficult. The algorithm given in the proof of Theorem 10.62 below is due to Buss [17].

THEOREM 10.62 (Buss). *The Boolean Sentence Value Problem is in* **ALogTime**.

PROOF. We give an algorithm in terms of a game between two players: one is called the Pebbler and the other is called the Challenger. The game is defined so that the Pebbler has a winning strategy if and only if the given Boolean sentence is true. The actual algorithm works by first playing the game and then determining the winner.

By using DeMorgan's laws we can remove all occurrences of $\neg$ from the sentence. This transformation requires counting the number of occurrences of the $\neg$ connective along each path in the tree and therefore can be done in $\mathbf{TC}^0$. Thus we can assume that the underlying tree is a binary tree whose inner nodes are labeled with $\vee$ or $\wedge$ and whose leaves are labeled with $\top$ and $\bot$. By padding the sentence with $\wedge\top$ we can assume that the tree has exactly $(2^{d+1}-1)$ leaves, for some $d \geq 1$. We number the leaves of the tree from left to right with

$$1, 2, \ldots, 2^{d+1}-1$$

(We do not number inner nodes of the tree.)

The pebbling game: The game will be played in at most d rounds; each round consists of a move by the Pebbler followed by a move by the Challenger. In each round the Pebbler will assert the values of some nodes in the tree (by pebbling them with Boolean values) and the Challenger must deny the Pebbler's assertion by challenging one of the pebbled nodes. The Challenger is required not to challenge any node that has been pebbled but unchallenged in a previous round. In effect, the Challenger implicitly agrees with the Pebbler on all pebbled nodes in the descendants of the currently challenged node. The idea is that at the end of at most d rounds the value of the challenged node is easily computed from an agreed node and some leaves, thus revealing the winner of the game. Intuitively, a winning strategy for the Pebbler is to pebble the nodes with their correct values, and if the Pebbler fails to do so, the Challenger can win by challenging some incorrectly pebbled node. So by having the Pebbler start with pebbling the root with $\top$, the sentence is true iff the Pebbler has a winning strategy.

Each move by a player will be specified by a constant number of bits $0, 1$. Essentially, the moves by the Pebbler (resp. the Challenger) can be interpreted as the existential (resp. universal) states, and playing the game can therefore be seen as running an alternating Turing machine in logtime.

The i-th round of the game involves the following nodes: c_i (c for *challenged*), a_i (a for *agreed*), u_i, v_i, u_i^1, u_i^2, v_i^1, v_i^2 (u_i^j and v_i^j are children of u_i and v_i, respectively) and leaves ℓ_i (ℓ for *left*) and r_i (r for *right*). In general, $\ell_i < r_i$, and ℓ_i (resp. r_i) is never to the right (resp. left) of the subtree rooted at a_i. Moreover, a_i is a descendant of c_i, u_i as well as v_i. Also, all leaf descendants of c_i that are not descendants of a_i are numbered in the range

$$\{\ell_i - 2^{d-i}+1, \ldots, \ell_i, \ldots, \ell_i + 2^{d-i}-1\} \cup \{r_i - 2^{d-i}+1, \ldots, r_i, \ldots, r_i + 2^{d-i}-1\}$$

Therefore after d rounds the Pebbler's asserted value of the challenged node c_d can be compared with the appropriate combination of a_d and the leaves ℓ_d, r_d, allowing us to determine the winner of the game. A possible configuration of the nodes is given in Figure 14. Here we orient the tree so that the leaves are at the bottom of the diagram.

In the i-th round the Pebbler pebbles nodes

$$u_i, v_i, u_i^1, u_i^2, v_i^1, v_i^2$$

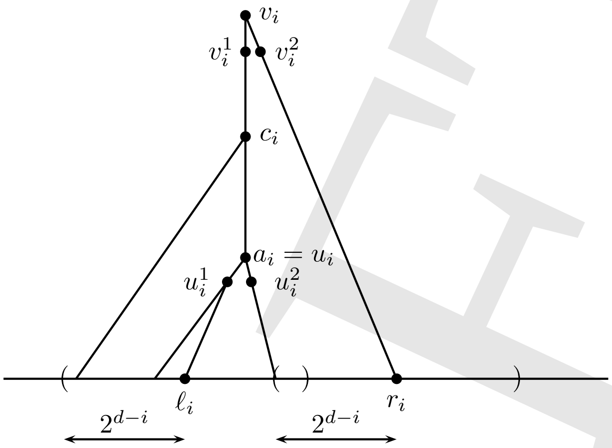

FIGURE 14. One possible configuration.

with some Boolean values, and the Challenger must either challenge one of these nodes or rechallenge a node it has challenged in the previous round. The Pebbler needs six bits for this task, and the Challenger needs three. Later we will summarize the conditions for ending the game in less than d rounds. For instance, the game will end if the Challenger challenges either u_i or v_i. This is because, for example, the asserted value of u_i can be compared with the asserted values of u_i^1 and u_i^2, or if u_i is a leaf its true value is readily available.

The nodes of the i-th round are determined as follows. First, c_i is the challenged node from the previous round (c_1 is understood to be the root). Also,

$$a_1 = 2^d, \qquad \ell_1 = 2^{d-1}, \qquad r_1 = 2^d + 2^{d-1}$$

and for $i \geq 1$:

$$u_i = lca(\ell_i, a_i), \qquad v_i = lca(a_i, r_i)$$

where $lca(n_1, n_2)$ denotes the least common ancestor of nodes n_1 and n_2. The nodes u_i^1 and u_i^2 are the left and right children of u_i, respectively. (If u_i is a leaf, then $u_i^1 = u_i^2 = u_i$.) Similarly for v_i^1 and v_i^2.

Next, for the $(i+1)$-st round (where $1 \leq i < d$) a_{i+1}, ℓ_{i+1}, r_{i+1} are determined based on the relative positions of c_i, u_i and v_i. For this purpose we need the following notation. Let

$$n_1 \rhd n_2$$

denote the fact that node n_1 is a proper ancestor of n_2, and

$$n_1 \trianglerighteq n_2$$

stand for $n_1 \rhd n_2$ or $n_1 = n_2$. It will be true in general that

$$c_i \trianglerighteq a_i, \qquad u_i \trianglerighteq a_i, \qquad v_i \trianglerighteq a_i$$

As a result, the only possible relative positions for c_i, u_i, v_i are listed in Table 2 below. Will refer to the cases by their number later.

Case 1	Case 2	Case 3	Case 4	Case 5	Case 6	Case 7
$u_i = v_i$	$c_i \trianglerighteq u_i \triangleright v_i$	$c_i \trianglerighteq v_i \triangleright u_i$	$u_i \triangleright c_i \trianglerighteq v_i$	$v_i \triangleright c_i \trianglerighteq u_i$	$u_i \triangleright v_i \triangleright c_i$	$v_i \triangleright u_i \triangleright c_i$

Table 2. Possible relative positions of c_i, u_i, v_i.

Note that exactly one of these cases will hold, and the Pebbler will use three bits to specify which one holds. Also, $u_i = v_i$ only if $u_i = v_i = a_i$.

The game ends in round i if the Challenger challenges u_i or v_i. So suppose first that in round i the Challenger challenges u_i^1 (i.e., $c_{i+1} = u_i^1$). The game ends in round i if $u_i \triangleright c_i$ (Cases 4,6,7) or $u_i = v_i$ (Case 1). For other cases,

$$a_{i+1} = \ell_i, \qquad \ell_{i+1} = \ell_i - 2^{d-i-1}, \qquad r_{i+1} = \ell_i + 2^{d-i-1} \tag{364}$$

See an illustration in Figure 15.

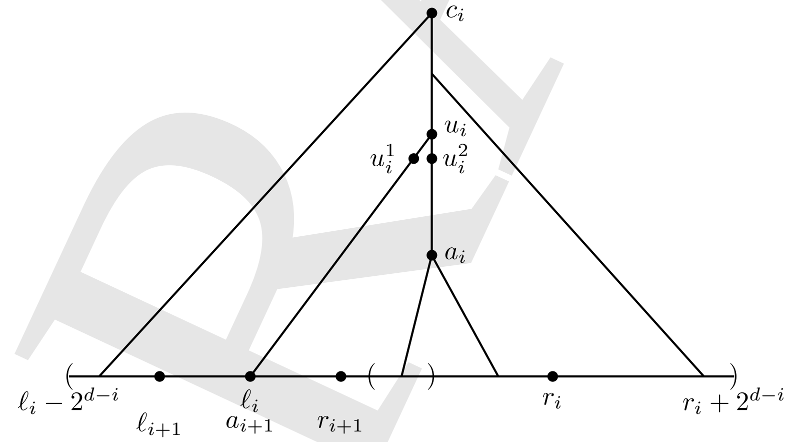

Figure 15. $c_{i+1} = u_i^1$ (v_i, v_i^1, v_i^2 are not shown).

Now suppose that the Challenger challenges u_i^2 in the i-th round. The game ends if $u_i = v_i$ (Case 1). If $u_i \triangleright v_i$ (Cases 2,4,6) then (see Figure 16 for an illustration):

$$a_{i+1} = v_i, \qquad \ell_{i+1} = \ell_i + 2^{d-i-1}, \qquad r_{i+1} = r_i + 2^{d-i-1} \tag{365}$$

Otherwise, $v_i \triangleright u_i$ (Cases 3,5,7), and

$$a_{i+1} = a_i, \qquad \ell_{i+1} = \ell_i + 2^{d-i-1}, \qquad r_{i+1} = r_i - 2^{d-i-1}$$

Note that if c_i is a proper descendant of u_i^2 then the Challenger will loose (see below).

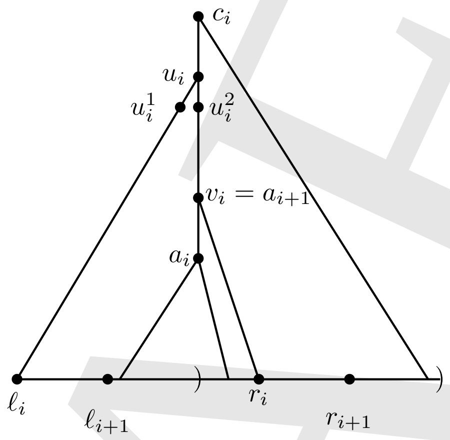

FIGURE 16. $c_{i+1} = u_i^2$ and $c_i \trianglerighteq u_i \triangleright v_i$.

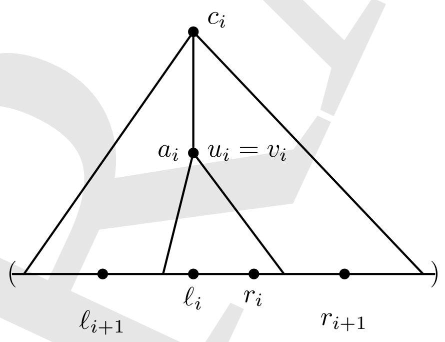

FIGURE 17. $c_{i+1} = c_i$ and $u_i = v_i$.

The cases where the Challenger challenges v_i^1 or v_i^2 are similar. So suppose now that in the i-th round the Challenger rechallenges c_i. The nodes a_{i+1}, ℓ_{i+1} and r_{i+1} are set as as specified in Table 3. Figure 17 illustrates Case 1.

In summary, in each round the Pebbler gives nine bits specifying the truth values of u_i, v_i, u_i^1, u_i^2, v_i^1, v_i^2 and the relative positions of c_i, u_i, v_i given in Table 2. (It is understood that the Pebbler also pebbles the

Case	$u_i = v_i$ (1)	$c_i \trianglerighteq u_i \triangleright v_i$ (2)	$c_i \trianglerighteq v_i \triangleright u_i$ (3)	$u_i \triangleright c_i \trianglerighteq v_i$ (4)	$v_i \triangleright c_i \trianglerighteq u_i$ (5)	$u_i, v_i \triangleright c_i$ (6,7)
a_{i+1}	a_i	u_i	v_i	v_i	u_i	a_i
ℓ_{i+1}	$\ell_i - t$	$\ell_i - t$	$\ell_i - t$	$\ell_i + t$	$\ell_i - t$	$\ell_i + t$
r_{i+1}	$r_i + t$	$r_i + t$	$r_i + t$	$r_i + t$	$r_i - t$	$r_i - t$

TABLE 3. The Challenger rechallenges c_i (i.e., $c_{i+1} = c_i$). Here $t = 2^{d-i-1}$.

root with $\top$ in the first round.) Each move by the Challenger consists of giving three bits specifying the challenging node.

The following moves cause the Pebbler to loose the game:

1) Pebble a leaf with the wrong value, or pebble incompatible values for u_i, u_i^1, u_i^2, v_i, v_i^1, v_i^2. For example, u_1 is an $\wedge$ node and u_1 is pebbled with $\bot$ while both u_i^1 and u_i^2 are pebbled with $\top$.
2) Pebble a node with both $\top$ and $\bot$.
3) Make wrong assertion about the relative positions of c_i, u_i and v_i.

The Challenger looses if he

1) challenges a correctly pebbled leaf;
2) challenges u_i or v_i when they are pebbled compatibly with u_i^1, u_i^2, v_i^1, v_i^2;
3) in round i challenges an non-descendant of the currently challenged node c_i;
4) in round i challenges a descendant of the currently agreed node a_i.

The game is played in at most d rounds; it may end in less than d rounds if a player obviously makes a mistake listed above and therefore looses the game. Thus the game ends as soon as

1) The Challenger challenges either u_i or v_i,
2) The Challenger challenges u_i^1 when the Pebbler says $u_i \triangleright c_i$ (Cases 4,6,7),
3) The Challenger challenges v_i^2 when the Pebbler says $v_i \triangleright c_i$ (Cases 5,6,7),
4) The Challenger challenges u_i^j or v_i^j when the Pebbler says $u_i = v_i$ (Case 1).

Claim. The Pebbler has a winning strategy iff the given sentence is true.

The Claim is straightforward: if the sentence is true, the Pebbler can always win by pebbling the nodes with their correct values and stating the correct relative positions of c_i, u_i, v_i; otherwise, if the sentence is false then the Challenger can win by always challenging the lowest node that is incorrectly pebbled.

Determining the winner: We finish the proof of Theorem 10.62 by showing that the winner of the Pebbling game above can be determined from the plays in **ALogTime**. The task is, given a sequence of moves of the players (represented as a binary string), to determine which player is the first to violate the conditions above. We will indeed show that this can be done in $\mathbf{TC}^0$.

We will first compute all ℓ_i, r_i, then all a_i. From these we can get u_i, v_i, u_i^1, u_i^2, v_i^1, v_i^2 easily. Then it is straightforward to find out the winner. Below we briefly show how to compute ℓ_i, r_i and a_i, for $1 \le i \le d$.

For simplicity, assume that the game lasts in exactly d rounds. Notice that ℓ_i and r_i have the form

$$\ell_i = x_d^i 2^d + x_{d-1}^i 2^{d-1} + \dots + x_{d-i}^i 2^{d-i}$$
$$r_i = y_d^i 2^d + y_{d-1}^i 2^{d-1} + \dots + y_{d-i}^i 2^{d-i}$$

where $x_j^i, y_j^i \in \{-1, 0, 1\}$. For example,

$$\ell_1 = 0 \times 2^d + 1 \times 2^{d-1}, \qquad r_1 = 1 \times 2^d + 1 \times 2^{d-1}$$

Also, if the Challenger challenges u_i^1 as in (364) then $x_{d-i-1}^{i+1} = -1$, $y_{d-i-1}^{i+1} = 1$, and for $d - i \le j \le d$:

$$x_j^{i+1} = y_j^{i+1} = x_j^i$$

On the other hand, if the Challenger challenges u_i^2 as in (365) then $x_{d-i-1}^{i+1} = y_{d-i-1}^{i+1} = 1$, and for $d - i \le j \le d$:

$$x_j^{i+1} = x_j^i, \qquad y_j^{i+1} = y_j^i$$

Generally, x_{d-i}^i and y_{d-i}^i can be easily extracted from the moves in round i; and for $0 \le j < i$, x_{d-j}^i and x_{d-j}^i can be computed from x_{d-j}^j and x_{d-j}^j by counting the number of "jumps" as in (364) where both ℓ_{j+1} and r_{j+1} are computed from only ℓ_j (or only r_j). From this we can conclude that ℓ_i and r_i can be computed in $\mathbf{TC}^0$ from the moves of both players.

Next, notice that we can compute simultaneously all i such that a_i is a leaf. For example a_1 is a leaf; a_{i+1} as in (364) is also a leaf. For each other value of i, let $j < i$ be largest so that a_j is a leaf and hence has been determined. Then a_i is the least common ancestor of a_j and a certain subset S_i of

$$\{\ell_j, r_j, \ell_{j+1}, r_{j+1}, \dots, \ell_{i-1}, r_{i-1}\}$$

For example, for $j \le k < i$, $r_k \in S_i$ if $a_{k+1} = v_k$ (e.g., as in (365)). The set S_i can be computed by an $\mathbf{AC}^0$ function from the moves of the players. Hence all a_i can be computed in $\mathbf{TC}^0$. $\dashv$

Recall that $(Z \models_0 X)$ holds iff the truth assignment Z satisfies the quantifier-free formula X, and $(Z \models_0^\Sigma X)$ and $(Z \models_0^\Pi X)$ are Σ_1^B and Π_1^B formulas that represent $(Z \models_0 X)$, respectively (see Lemma 10.26).

By formalizing the algorithm given in the proof of Theorem 10.62 above we can strengthen Lemma 10.26.

Corollary 10.63. $\mathbf{VNC}^1 \vdash (Z \models_0^\Sigma X) \leftrightarrow (Z \models_0^\Pi X)$.

Proof Sketch. The direction

$$(Z \models_0^\Sigma X) \supset (Z \models_0^\Pi X)$$

can in fact be proved in $\mathbf{V}^0$, so we focus on proving that

$$(Z \models_0^\Pi X) \supset (Z \models_0^\Sigma X)$$

In essence, we have to prove the existence of the array E' where $E'(i,j)$ is the truth value of the subformula encoded by $X[i,j]$, for all $1 \leq i \leq j \leq n$, where n is the length of X (as a string over the alphabet (363) on page 385). Thus we will evaluate the subformulas $X[i,j]$ in parallel. Using the function *Fval* (Definition 9.93), the idea is that each subformula $X[i,j]$ will be evaluated by constructing a suitable balanced tree encoded by some (a,G,I) (as in Sections 9E.1 and 9E.2) such that $\mathit{Fval}(a,G,I)(1)$ is the value of $X[i,j]$. The fact that all subformulas $X[i,j]$ can be evaluated simultaneously in $\mathbf{VNC}^1$ will follows from the fact that $\mathit{Fval}^\star$ is provably total in $\mathbf{VNC}^1$ (Exercise 9.102).

We will construct a tuple (a,G,I) so that $\mathit{Fval}(a,G,I)(1)$ is the value of the sentence X; the constructions for other subformulas of X are similar. Since $\overline{\mathbf{VNC}}^1$ is a conservative extension of $\mathbf{VNC}^1$, we will actually work in $\overline{\mathbf{VNC}}^1$. Recall that the **ALogTime** algorithm from the proof of Theorem 10.62 is obtained by first playing the pebbling game and then determining the winner of the game. The balanced tree (a,G) will encode the game playing part of the algorithm: each path from the root of the tree to a leaf corresponds to a possible play of the game. Each input bit $I(x)$ specifies the winner of the play corresponding to the path ending with that leaf.

We will in fact specify a balanced bounded fan-in tree T. Conversion from this tree to a balanced binary tree (a,G) as required for the arguments of *Fval* is straightforward and will be omitted. Let n be the number of constants in X (i.e., the number of leaves in the tree representing X). Let d be such that $2^d \leq n < 2^{d+1}$. As in the **ALogTime** algorithm for BSVP, we will pad the formula X with necessary $\wedge\top$ in order to make X a sentence with exactly $2^{d+1}-1$ constants $\top,\bot$ (i.e., the underlying tree for X has exactly $2^{d+1}-1$ leaves).

The tree T has $2d$ alternating layers of nodes corresponding to d rounds of the game. We number the layers starting at the root with number 1. The root is an $\vee$ node; generally, $\vee$ nodes are on layers $2j-1$ (for $1 \leq j \leq d$) and correspond to the Pebber's moves. They all have fan-in

$$7 \times 2^6$$

that represents 7×2^6 possibilities for a move by the Pebbler (2^6 different choices of the values for u_i, v_i, u_i^1, u_i^2, v_i^1, v_i^2, and 7 possible relative positions of c_i, u_i, v_i as in Table 2). Each child of an $\vee$ node is a $\wedge$ node that corresponds to a move by the Challenger. Thus all $\wedge$ nodes are on layers $2j$ (for $1 \le j \le d$) and have branching factor of 7 which encodes 7 possible choices for the Challenger. The children of an $\wedge$ node on layer $2j$ where $j < d$ correspond to the Pebbler's responses in round $(j+1)$, and the children of the $\wedge$ nodes on layer $2d$ are inputs that are specified below.

Note that here we make T a balanced tree by having each play of the game end in exactly d rounds. (If some play ends in less than d rounds, simply add arbitrary moves to it.) Using the fact that the relation $BIT(i,x)$ (Section 3C.3) is Δ_0-definable, the (binary form of the) tree T can be defined in $\mathbf{V}^0$.

Now, the "determining the winner" part in the proof of Theorem 10.62 has been shown to be computable in $\mathbf{TC}^0$; it is straightforward to formalize this part in $\mathbf{VTC}^0$ (and hence in $\mathbf{VNC}^1$). This implies that in $\overline{\mathbf{VNC}}^1$ we can define a string of inputs I to the tree T so that $I(x)$ is true iff the path from the root of T to the leaf x corresponds to a play of the pebbling game where the Pebbler wins.

We finish the proof by showing the correctness of our formalization. Simply write $Fval(T,I)$ for $Fval(a,G,I)$, where (a,G) is a balanced binary formula equivalent to T. Then to prove (in $\overline{\mathbf{VNC}}^1$) that our formalization is correct, we need to prove:

Claim. Suppose that $(A \odot B)$ is a subformula of X, where $\odot \in \{\wedge, \vee\}$, and suppose that $(T_{A\odot B}, I_{A\odot B})$, (T_A, I_A) and (T_B, I_B) are the result of our constructions for the sentences $A \odot B$, A and B, respectively. Then $\overline{\mathbf{VNC}}^1$ proves

$$Fval(T_{A\odot B}, I_{A\odot B})(1) = Fval(T_A, I_A)(1) \odot Fval(T_B, I_B)(1) \tag{366}$$

We prove the claim by structural induction on the subformula $(A \odot B)$. The base case (where A, B are both constants) is obvious. For the induction step consider the following cases:

Case I: $(A \odot B) = (A \wedge B)$.

Case Ia: First suppose that

$$Fval(T_A, I_A)(1) = Fval(T_B, I_B)(1) = \top$$

We show that $Fval(T_{A\wedge B}, I_{A\wedge B})(1) = \top$ (i.e., that the Pebbler has a winning strategy for the game played on $(A \wedge B)$).

Consider playing the game on $(A \wedge B)$. Intuitively, the Pebbler win by always giving the right values for the nodes u_i, v_i, u_i^1, u_i^2, v_i^1 and v_i^2. Formally, we will show that all nodes on the "winning paths" of the tree $T_{A\wedge B}$ are true. Here winning paths are defined in favor of the Pebbler: they are the paths from the root of $T_{A\wedge B}$ that follow the Pebbler's right move at every $\vee$ node (or any branch of the $\vee$ node if the game has ended earlier with the Pebbler being the winner). To find the winning paths we use the induction hypothesis, i.e., the value of a subformula C of $(A \wedge B)$ is $Fval(T_C, I_C)(1)$. Thus, a path from the root of $T_{A\wedge B}$ to a leaf x is a winning path if at every $\vee$ node the path follow the edge that is specified by (i) the right relative positions of c_i, u_i and v_i (as in Table 2) and (ii) the values of u_i, v_i, u_i^1, u_i^2, v_i^1 and v_i^2 as given by

$$Fval(T_{U_i}, I_{U_i})(1), \qquad Fval(T_{V_i}, I_{V_i})(1), \qquad \text{etc.}$$

(here U_i denotes the subformula whose root is u_i, etc.).

We will prove by reverse induction on the j, $0 \le j \le d$, that all nodes on layer $(2j+1)$ of all winning paths are true. For $j = 0$ we will have that the root of $T_{A\wedge B}$ is true, i.e., $Fval(T_{A\wedge B}, I_{A\wedge B})(1) = \top$ and we will be done with Case Ia.

For the base case, $j = d$, and all nodes on layer $(2d+1)$ are leaves of $T_{A\wedge B}$. Using the induction hypothesis (of the claim that (366) holds for all subformulas of $(A\wedge B)$) and using the fact that both $Fval(T_A, I_A)(1)$ and $Fval(T_B, I_B)(1)$ are true, by definition the inputs to $T_{A\wedge B}$ at the end of all winning paths are true. The induction step is straightforward: suppose that node w is on a winning path and w is on layer $(2j+1)$. Let t be the child of w the corresponds to a right move by the Pebbler (or any child of w if the Pebbler has won before round $(j+1)$). Then by the induction hypothesis all children of t are true. Hence both t and w are true, because w is an $\vee$ node and t is an $\wedge$ node.

Case Ib: At least one of $Fval(T_A, I_A)(1)$ and $Fval(T_B, I_B)(1)$ is $\perp$. The proof in this case is similar to Case Ia. Define "loosing paths" to be paths from the root of $T_{A\wedge B}$ where the Challenger always challenges the lowest node that is wrongly pebbled (recall that the trees are oriented with the roots at the top). Then by similar arguments as in Case Ia, it can be shown that all nodes on the loosing paths are false. In particular, the root of $T_{A\wedge B}$ is false.

Case II: $(A \odot B) = (A \vee B)$. This case can be handled similarly to Case I. $\dashv$

EXERCISE 10.64. *Show that* $\mathbf{VNC}^1$ *proves* $RFN_{\mathbf{PK}}$.

10D. Threshold Logic

In this section we will associate the theory $\mathbf{VTC}^0$ with proof systems called *bounded depth* **PTK** in the same way that $\mathbf{V}^1$ is associated with $\mathbf{G}_1^\star$. The system **PTK** extends **PK** by having a new kind of connectives called *threshold connectives* that informally correspond to the counting function *numones*. (the Boolean connectives $\vee$, $\wedge$ become superfluous in **PTK**.) The full version of **PTK** is p-equivalent to **PK**. So the systems that we are interested in are obtained from **PTK** by limiting the depth of the cut formulas to some constants in $\mathbb{N}$.

We introduce the sequent calculus **PTK** in Section 10D.1. We will associate the theory $\mathbf{VTC}^0$ with *bounded depth* **PTK** by showing that the families of tautologies translated from Σ_0^B theorems of $\mathbf{VTC}^0$ have polynomial-size bounded-depth **PTK** proofs (in fact, the bounded depth proofs are computable in $\mathbf{TC}^0$). This is done in Section 10D.2. The translation can be extended to quantified tautologies that correspond to any bounded theorem of $\mathbf{VTC}^0$ by using a quantified version of **PTK**. We introduce this extension of **PTK** in Section 10D.3.

10D.1. The Sequent Calculus PTK. The sequent calculus **PTK** is defined similarly to **PK**, but instead of the binary connectives $\wedge$ and $\vee$ **PTK** contains *threshold connectives* Th_k (for $1 \leq k \in N$) that have unbounded arity. The semantic of Th_k is that

$$\mathsf{Th}_k(A_1, A_2, \ldots, A_n)$$

is true if and only if at least k formulas A_i's are true. For example,

$$\mathsf{Th}_2(p, q, r) \Leftrightarrow (p \wedge q) \vee (q \wedge r) \vee (r \wedge p)$$

Also,

(367)

$$\mathsf{Th}_1(A_1, A_2, \ldots, A_n) \Leftrightarrow \bigvee_{i=1}^{n} A_i; \qquad \mathsf{Th}_n(A_1, A_2, \ldots, A_n) \Leftrightarrow \bigwedge_{i=1}^{n} A_i$$

For readability we will sometimes use $\bigwedge$ and $\bigvee$ in **PTK** formulas in place of Th_1 and Th_k.

Formally, **PTK** formulas (or threshold formulas, or just formula) are built from

- propositional constants $\top, \bot$;
- propositional variables $p, q, r, \ldots$;
- connectives $\neg, \mathsf{Th}_k$;
- parenthesis (,);

using the rules:

(a) $\top$, $\bot$, and p are *atomic* formulas, for any propositional variable p;
(b) if A is a formula, so is $\neg A$;
(c) for $n \geq 2, 1 \leq k \leq n$, if $A_1, A_2, \ldots, A_n$ are formulas, so is $\mathsf{Th}_k(A_1, A_2, \ldots, A_n)$.

Moreover,

$$\mathsf{Th}_0(A_1, A_2, \ldots, A_n) =_{\text{def}} \top, \qquad \mathsf{Th}_k(A_1, A_2, \ldots, A_n) =_{\text{def}} \bot \text{ for } k > n$$

The sequent calculus **PTK** is defined similarly to **PK** (Definition 2.2). Here the logical axioms are of the form

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

where A is any **PTK** formula. The weakening, exchange, contraction, cut and $\neg$ introduction rules are the same as on page 9. The other rules of **PTK** are listed below:

The left and right all-introduction rules (all-left and all-right) are as follows:

$$\frac{A_1, \ldots, A_n, \Lambda \longrightarrow \Gamma}{\mathsf{Th}_n(A_1, \ldots, A_n), \Lambda \longrightarrow \Gamma} \qquad \frac{\Lambda \longrightarrow A_1, \Gamma \quad \ldots \quad \Lambda \longrightarrow A_n, \Gamma}{\Lambda \longrightarrow \mathsf{Th}_n(A_1, \ldots, A_n), \Gamma}$$

Left and right one-introduction rules (one-left and one-right) are:

$$\frac{A_1, \Lambda \longrightarrow \Gamma \quad \ldots \quad A_n, \Lambda \longrightarrow \Gamma}{\mathsf{Th}_1(A_1, \ldots, A_n), \Lambda \longrightarrow \Gamma} \qquad \frac{\Lambda \longrightarrow A_1, \ldots, A_n, \Gamma}{\Lambda \longrightarrow \mathsf{Th}_1(A_1, \ldots, A_n), \Gamma}$$

Th_k-introduction rules (for $2 \leq k \leq n-1$):

$$\frac{\mathsf{Th}_k(A_2, \ldots, A_n), \Lambda \longrightarrow \Gamma \qquad A_1, \mathsf{Th}_{k-1}(A_2, \ldots, A_n), \Lambda \longrightarrow \Gamma}{\mathsf{Th}_k(A_1, \ldots, A_n), \Lambda \longrightarrow \Gamma} \ \mathsf{Th}_k\text{−left}$$

$$\frac{\Lambda \longrightarrow A_1, \mathsf{Th}_k(A_2, \ldots, A_n), \Gamma \qquad \Lambda \longrightarrow \mathsf{Th}_{k-1}(A_2, \ldots, A_n), \Gamma}{\Lambda \longrightarrow \mathsf{Th}_k(A_1, \ldots, A_n), \Gamma} \ \mathsf{Th}_k\text{−right}$$

Similar to **PK**, **PTK** is sound and complete and has cut elimination. Proving these properties is left as an exercise.

EXERCISE 10.65 (Soundness and Completeness of **PTK**). *Show that a sequent provable in* **PTK** *is valid, and valid sequents have cut-free* **PTK** *proofs.*

We will prove in Theorem 10.69 that **PTK** and **PK** are p-equivalent. So we are mainly interested in subsystems of **PTK** where the cut formulas have bounded depths.

DEFINITION 10.66 (Depth of a **PTK** Formula). *The depth of a* **PTK** *formula A is the nesting depth of the connectives in A.*

So, for example, the atomic **PTK** formulas have depth 0.

DEFINITION 10.67 (Bounded Depth **PTK**). *For each constant $d \in \mathbb{N}$, a d-***PTK** *proof is a* **PTK** *proof in which all cut formulas have depth at most d. A* bounded depth **PTK** *system (or just* **bPTK***) is any system d-***PTK** *for $d \in \mathbb{N}$.*

The treelike versions of **PTK**, d-**PTK** and **bPTK** are denoted by **PTK***, d-**PTK*** and b-**PTK***, respectively.

THEOREM 10.68. *For any $d \in \mathbb{N}$, d-***PTK** *p-simulates d-***PK** *with respect to formulas of depth d.*

PROOF. Let π be a d-**PK** derivation whose end sequent has depth at most d. Note that all formulas in π have depth at most d. We translate each **PK** formula into a **PTK** formula using (367). The results is a **PTK** formula of the same depth. Thus each formula A in

π is translated into a **PTK** formula A' of depth at most d. For each sequent $\mathcal{S}$ in π, let $\mathcal{S}'$ be the translation of $\mathcal{S}$. We prove by induction on the length of π that there is a d-**PTK** proof π' of size polynomial in the size of π that contains all translations $\mathcal{S}'$ of sequents $\mathcal{S}$ in π.

The base case is obvious because axioms of **PK** are translated into axioms of **PTK** of the same depth. For the induction step, suppose that $\pi = (\pi_1, S)$ where S is the end sequent of π. Consider, for example, the case where $\mathcal{S}$ is derived from two sequents $\mathcal{S}_1$ and $\mathcal{S}_2$ in π_1 as follows:

$$\frac{\mathcal{S}_1 \qquad \mathcal{S}_2}{\mathcal{S}} = \frac{\bigvee A_i, \Gamma \longrightarrow \Delta \qquad \bigvee B_j, \Gamma \longrightarrow \Delta}{\bigvee A_i \vee \bigvee B_j, \Gamma \longrightarrow \Delta}$$

Here $\bigvee A_i$ is any parenthesizing of $A_1 \vee A_2 \vee \cdots \vee A_n$, and similarly for $\bigvee B_j$. Note that

$$\mathcal{S}_1' = \mathsf{Th}_1(\vec{A}'), \Gamma' \longrightarrow \Delta', \qquad \mathcal{S}_2' = \mathsf{Th}_1(\vec{B}'), \Gamma' \longrightarrow \Delta',$$
$$\mathcal{S}' = \mathsf{Th}_1(\vec{A}', \vec{B}'), \Gamma' \longrightarrow \Delta'$$

Using the one-left rule we can derive

$$\mathsf{Th}_1(\vec{A}', \vec{B}') \longrightarrow \mathsf{Th}_1(\vec{A}'), \mathsf{Th}_1(\vec{B}')$$

From this and S_1', S_2', using the cut rule (with cut formulas $\mathsf{Th}_1(\vec{A}')$ and $\mathsf{Th}_1(\vec{B}')$) we derive $\mathcal{S}'$. The derivation π' is obtained from π_1' and the above derivation.

It is easy to see that π' as described above has size bounded by a polynomial in the size of π. ⊣

Theorem 10.69. **PK** *is p-equivalent to* **PTK**.

Proof Sketch. The fact that **PTK** p-simulates **PK** follows from Theorem 10.68 above. It remains to shows that **PK** p-simulates **PTK**.

In Section 9E.3 we show that the function *numones* is provably total in $\mathbf{VNC}^1$. In essence, we construct a uniform family of $\mathbf{NC}^1$ circuits (i.e., formulas) that compute *numones*. In other words, there are **PK** formulas $F_{n,k}(p_1, p_2, \ldots, p_n)$ so that

$$F_{n,k}(p_1, p_2, \ldots, p_n) \Leftrightarrow \text{the number of } \top \text{ in } p_1, p_2, \ldots, p_n \text{ is } k$$

Moreover, similar to the fact that $\mathbf{VNC}^1 \vdash$ *NUMONES* (Theorem 9.104) we can prove:

Proposition 10.70. *There are polynomial-size* **PK***-proofs of the following sequents:*

1) $F_{n,n}(A_1, A_2, \ldots, A_n) \longrightarrow A_i$ *(for* $1 \le i \le n$*);*
2) $A_1, A_2, \ldots, A_n \longrightarrow F_{n,n}(A_1, A_2, \ldots, A_n)$*;*
3) $A_i \longrightarrow F_{n,1}(A_1, A_2, \ldots, A_n)$ *(for* $1 \le i \le n$*);*
4) $F_{n,1}(A_1, A_2, \ldots, A_n) \longrightarrow A_1, A_2, \ldots, A_n$*;*
5) $F_{n,\ell}(A_1, A_2, \ldots, A_n) \longrightarrow A_1, F_{n-1,\ell}(A_2, \ldots, A_n)$ *(for* $1 \le \ell \le n-1$*);*

6) $A_1, F_{n,\ell}(A_1, A_2, \dots, A_n) \longrightarrow F_{n-1,\ell-1}(A_2, \dots, A_n)$ *(for* $2 \le \ell \le n$*);*
7) $F_{n-1,\ell}(A_2, \dots, A_n) \longrightarrow A_1, F_{n,\ell}(A_1, A_2, \dots, A_n)$ *(for* $1 \le \ell \le n-1$*);*
8) $A_1, F_{n-1,\ell-1}(A_2, \dots, A_n) \longrightarrow F_{n,\ell}(A_1, A_2, \dots, A_n)$ *(for* $2 \le \ell \le n$*).*

Now **PTK** formulas are translated into **PK** formulas (of unbounded depth) inductively using the formulas $F_{n,k}$ as follows. No translation is required for atomic formulas, because they are the same in **PK** and **PTK**. For the inductive step, suppose that A_i has been translated to A'_i, for $1 \le i \le n$. Then $\mathsf{Th}_k(A_1, A_2, \dots, A_n)$ is translated into

$$\bigvee_{\ell=k}^{n} F_{n,\ell}(A'_1, A'_2, \dots, A'_n)$$

To show that **PK** p-simulates **PTK** it suffices to show that the translations of rules of **PTK** have polynomial-size **PK** proofs.

First, consider the rule all-left. We need to show that there is a polynomial-size **PK** derivation of the form

$$\frac{A_1, A_2, \dots, A_n, \Lambda \longrightarrow \Gamma}{\overline{F_{n,n}(A_1, A_2, \dots, A_n), \Lambda \longrightarrow \Gamma}}$$

For this we can use Proposition 10.70 (1) with successive cuts on the formulas A_i.

Similarly, the rule one-left can be simulated using Proposition 10.70 (4).

Consider now the rule Th_k-left. Suppose that $2 \le k \le n-1$, we need to give polynomial-size **PK**-derivations of the form

$$\frac{\vee_{\ell=k}^{n-1} F_{n-1,\ell}(A_2, \dots, A_n), \Lambda \longrightarrow \Gamma \quad A_1, \vee_{\ell=k-1}^{n-1} F_{n-1,\ell}(A_2, \dots, A_n), \Lambda \longrightarrow \Gamma}{\overline{\vee_{\ell=k}^{n} F_{n,\ell}(A_1, A_2, \dots, A_n), \Lambda \longrightarrow \Gamma}}$$

It suffices to derive for each ℓ, $k \le \ell \le n$, the sequent

$$F_{n,\ell}(A_1, A_2, \dots, A_n), \Lambda \longrightarrow \Gamma \tag{368}$$

From Proposition 10.70 (5) we derive (by weakening and $\vee$-right):

$$F_{n,\ell}(A_1, A_2, \dots, A_n) \longrightarrow A_1, \bigvee_{\ell=k}^{n-1} F_{n-1,\ell}(A_2, \dots, A_n)$$

Using the cut rule for this sequent and

$$\bigvee_{\ell=k}^{n-1} F_{n-1,\ell}(A_2, \dots, A_n), \Lambda \longrightarrow \Gamma$$

we obtain

$$F_{n,\ell}(A_1, A_2, \dots, A_n), \Lambda \longrightarrow A_1, \Gamma \tag{369}$$

From Proposition 10.70 (6) for $\ell = k$ we obtain (using weakening and $\vee$-right):

$$A_1, F_{n,k}(A_1, A_2, \ldots, A_n) \longrightarrow \bigvee_{\ell=k-1}^{n-1} F_{n-1,\ell}(A_2, \ldots, A_n)$$

From this and

$$A_1, \bigvee_{\ell=k-1}^{n-1} F_{n-1,\ell}(A_2, \ldots, A_n), \Lambda \longrightarrow \Gamma$$

we derive

$$A_1, F_{n,k}(A_1, A_2, \ldots, A_n), \Lambda \longrightarrow \Gamma \tag{370}$$

Combined (369) with (370) we obtain the desired derivation.

It is easy to verify that the above derivations have size polynomial in the size of the end sequents. Simulating the other rules of **PTK** is left as an exercise. $\dashv$

EXERCISE 10.71. *Complete the proof of Theorem 10.69 by showing that the translations of the rules all-right, one-right and* Th*-right have polynomial-size* **PK***-proofs.*

10D.2. Propositional translation for $\mathbf{VTC}^0$. Our goal of this section is to translate $\mathbf{VTC}^0$-proofs into families of polynomial-size $\mathbf{bGTC}_0$-proofs. One way would be to translate directly all instances of the axiom *NUMONES* (224) into **PTK** formulas (the bits of the "counting sequence" Y in *NUMONES* are translated using the Th_k connectives). Here we take another approach, based on Lemma 10.73 below.

Recall that $numones'(z, X)$ has the same value as $numones(z, X)$ (which is the number of elements in X that are less than z). For convenience, we list below the defining axioms of $numones'$ ((228), (229) and (230), page 273):

$$numones'(0, X) = 0 \tag{371}$$

$$X(z) \supset numones'(z+1, X) = numones'(z, X) + 1 \tag{372}$$

$$\neg X(z) \supset numones'(z+1, X) = numones'(z, X). \tag{373}$$

DEFINITION 10.72 ($\mathbf{V}^0(numones')$). *The theory* $\mathbf{V}^0(numones')$ *has vocabulary* $\mathcal{L}_A^2 \cup \{numones'\}$ *and is axiomatized by* 2-**BASIC***, the axioms* (371)*,* (372)*,* (373) *and the* $\Sigma_0^B(numones')$-**COMP** *axiom scheme.*

LEMMA 10.73. $\mathbf{V}^0(numones')$ *is a conservative extension of* $\mathbf{VTC}^0$.

PROOF. First, *NUMONES* is provable in $\mathbf{V}^0(numones')$ because the counting sequence Y in *NUMONES* can be defined by $\Sigma_0^B(numones')$-**COMP** as follows:

$$(Y)^z = y \leftrightarrow numones'(z, X) = y$$

Hence $\mathbf{V}^0(\mathit{numones}')$ extends $\mathbf{VTC}^0$.

Also, $\overline{\mathbf{VTC}}^0$ is an extension of $\mathbf{V}^0(\mathit{numones}')$, so the fact that $\overline{\mathbf{VTC}}^0$ is conservative over $\mathbf{VTC}^0$ (Theorem 9.34) implies that $\mathbf{V}^0(\mathit{numones}')$ is conservative over $\mathbf{VTC}^0$. $\dashv$

Suppose that φ is a Σ_0^B theorem of $\mathbf{VTC}^0$. It follows from Lemma 10.73 that φ has a $\mathbf{V}^0(\mathit{numones}')$-proof π. All formulas in π are $\Sigma_0^B(\mathit{numones}')$, and we will show that π can be translated into a family of polynomial-size bounded-depth **PTK**-proofs for the translation of φ.

We will describe the translation of atomic formulas. The translations of other $\Sigma_0^B(\mathit{numones}')$ formulas build up inductively as in Section 7B.1 using appropriate connectives Th_k for $\wedge$ and $\vee$.

Thus let $\varphi(\vec{x}, \vec{X})$ be an atomic formula. If φ does not contain *numones*$'$ then the translation $\varphi[\vec{m}, \vec{n}]$ is defined as in Section 7B.1 (using Th_k instead of $\wedge, \vee$). So suppose that φ contains *numones*$'$. Now if φ is of the form $X(t)$, where t contains *numones*$'$, then we can use the equivalence

$$X(t) \leftrightarrow \exists z < |X|(z = t \wedge X(z))$$

to translate φ using the translations of other atomic formulas $z = t$ and $X(z)$ (the latter does not contain *numones*$'$). Thus we only need to focus on atomic formulas φ of the form $s = t$ or $s \leq t$.

Let

$$\mathit{numones}'(t_1, X_1), \mathit{numones}'(t_2, X_2), \ldots, \mathit{numones}'(t_\ell, X_\ell)$$

be all occurrences of *numones*$'$ in φ (some t_i may contain terms of the form $\mathit{numones}'(t_j, X_j)$). Note that the truth value of $\varphi(\vec{x}, \vec{X})$ depends on the values $\vec{m}$ of $\vec{x}$, the length $\vec{n}$ of $\vec{X}$ and the values of $\mathit{numones}'(t_i, X_i)$. So for a fixed sequences $\vec{m}, \vec{n}$, let $S = S_{\varphi,\vec{m},\vec{n}}$ be the following set (recall *val* on page 158)

$$\{(k_1, k_2, \ldots, k_\ell) \;:\; k_i \leq \mathit{val}(t_i(\vec{m}, \vec{n})), \;\text{ and }\; \varphi \text{ is true when } \mathit{numones}'(t_i, X_i) = k_i, \text{ for } 1 \leq i \leq \ell\}$$

Recall that for each string variables X_i and a length $n_i \geq 2$ we introduce the propositional variables

$$\vec{p}^{X_i} = p_0^{X_i}, p_1^{X_i}, \ldots, p_{n_i-2}^{X_i}$$

If the set S is empty, then define $\varphi[\vec{m}; \vec{n}] = \bot$; otherwise $\varphi[\vec{m}; \vec{n}]$ is defined to be the simplification (explained below) of (374) below. Note that for readability we here use $\bigwedge$ for Th_k in $\mathsf{Th}_k(p_1, p_2, \ldots, p_k)$, and $\bigvee$ for Th_1 in $\mathsf{Th}_k(p_1, p_2, \ldots, p_k)$. The translation of φ is obtained by simplifying the following formula:

(374)

$$\bigvee_{\vec{k} \in S} \bigwedge_{i=1}^{\ell} (\mathsf{Th}_{k_i}(p_0^{X_i}, p_1^{X_i}, \ldots, p_{s_i-1}^{X_i}) \wedge \neg\mathsf{Th}_{k_i+1}(p_0^{X_i}, p_1^{X_i}, \ldots, p_{s_i-1}^{X_i}))$$

where $s_i = val(t_i(\vec{m}, \vec{n}))$ and $p^{X_i}_{n_i-1} = \top$, $p^{X_i}_j = \bot$ for $j \geq n_i$.

The simplification of (374) is performed inductively, starting with the atomic formulas $\mathsf{Th}_{k_i}(p^{X_i}_0, p^{X_i}_1, \ldots, p^{X_i}_{s_i-1})$ and $\mathsf{Th}_{k_i+1}(p^{X_i}_0, p^{X_i}_1, \ldots, p^{X_i}_{s_i-1})$. Each formula is simplified by applying the following procedure repeatedly (recall that

$$\mathsf{Th}_0(A_1, A_2, \ldots, A_n) =_{\text{def}} \top$$

and

$$\mathsf{Th}_k(A_1, A_2, \ldots, A_n) =_{\text{def}} \bot$$

Simplification Procedure: Whenever possible

- $\mathsf{Th}_1(A)$ is simplified to A,
- $\neg\bot$ is simplified to $\top$,
- $\neg\top$ is simplified to $\bot$,
- $\mathsf{Th}_1(A, A, A_1, A_2, \ldots, A_n)$ is simplified to $\mathsf{Th}_1(A, A_1, A_2, \ldots, A_n)$,
- $\mathsf{Th}_{n+1}(A, A, A_1, \ldots, A_{n-1})$ is simplified to $\mathsf{Th}_n(A, A_1, \ldots, A_{n-1})$,
- $\mathsf{Th}_k(\bot, A_1, A_2, \ldots, A_n)$ is simplified to $\mathsf{Th}_k(A_1, A_2, \ldots, A_n)$,
- $\mathsf{Th}_k(\top, A_1, A_2, \ldots, A_n)$ is simplified to $\mathsf{Th}_{k-1}(A_1, A_2, \ldots, A_n)$.

Example 10.74. *Recall the defining axioms* (371), (372) *and* (373) *for numones$'$. They are translated as follows.*

(a) (371) *is translated into* $\top$.

(b) *To translate* (372), *first we translate the atomic formula*

$$\varphi(z, X) \equiv \mathit{numones}'(z+1, X) = \mathit{numones}'(z, X) + 1$$

Here $\ell = 2$, $t_1 = z+1, t_2 = z, X_1 = X_2 = X$. *For* $m, n \in \mathbb{N}$, $n \geq 2$, *we have*

$$S_{\varphi,m,n} = \{(k+1, k) \;:\; k \leq m\}$$

We omit the superscript X *for the variables* p^X_i, *and let* $\vec{p}$ *denote* $p_0, p_1, \ldots, p_{m-1}$. *Then* $\varphi[m; n]$ *is (the simplification of)*

$$\bigvee_{k=0}^{m} \big((\mathsf{Th}_{k+1}(\vec{p}, p_m) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)) \wedge (\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+1}(\vec{p}))\big)$$

As a result, (372) *translates into*

$$\begin{cases} \neg p_m \vee \varphi[m; n] \quad \textit{(see (375) below)} & \textit{if } m \leq n-2 \\ \varphi[n-1; n] \quad \textit{(see (376) below)} & \textit{if } m = n-1 \\ \top & \textit{if } m \geq n \end{cases}$$

Note that for $m \leq n-2$,

$$\text{(375)} \quad \varphi[m; n] \equiv (\mathsf{Th}_1(\vec{p}, p_m) \wedge \neg\mathsf{Th}_2(\vec{p}, p_m) \wedge \neg\mathsf{Th}_1(\vec{p})) \vee \left(\bigvee_{k=1}^{m-1} \big(\mathsf{Th}_{k+1}(\vec{p}, p_m) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m) \wedge \mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+1}(\vec{p})\big)\right) \vee \big(\mathsf{Th}_{m+1}(\vec{p}, p_m) \wedge \mathsf{Th}_m(\vec{p})\big)$$

where $\vec{p}$ stands for $p_0, p_1, \ldots, p_{m-1}$. Also,

(376)

$$\varphi[n-1;n] \equiv \neg \mathsf{Th}_1(\vec{p}) \vee \left(\bigvee_{k=1}^{n-2} (\mathsf{Th}_k(\vec{p}) \wedge \neg \mathsf{Th}_{k+1}(\vec{p})) \right) \vee \mathsf{Th}_{n-1}(\vec{p})$$

where $\vec{p} = p_0, p_1, \ldots, p_{n-2}$.

(c) *For (373), consider the atomic formula*

$$\psi(z, X) \equiv \mathit{numones}'(z+1, X) = \mathit{numones}'(z, X)$$

Here ℓ, t_1, t_2, X_1, X_2 are as in (b) and

$$S_{\psi,m,n} = \{(k,k) \ : \ k \leq m\}$$

Again, drop mentions of the superscript X, and let $\vec{p} = p_0, p_1, \ldots, p_{m-1}$. The formula $\psi[m;n]$ is (the simplification of)

$$\bigvee_{k=0}^{m} \big(\mathsf{Th}_k(\vec{p}, p_m) \wedge \neg \mathsf{Th}_{k+1}(\vec{p}, p_m)) \wedge (\mathsf{Th}_k(\vec{p}) \wedge \neg \mathsf{Th}_{k+1}(\vec{p})\big) \tag{377}$$

For $m \geq n$, the simplification of (377) is just $\varphi[n-1;n]$ in (376). Hence, (373) translates into

$$\begin{cases} p_m \vee \psi[m;n] \ \textit{(see (375))} & \textit{if } m \leq n-2 \\ \top & \textit{if } m = n-1 \\ \varphi[n-1;n] \ \textit{see (376)} & \textit{if } m \geq n \end{cases}$$

We will show that the translations of (371), (372) and (373) described above d-$\mathbf{GTC}_0^\star$ proofs of size polynomial in m, n, for some constant $d \in \mathbb{N}$. We need the following lemma.

LEMMA 10.75. (a) *The sequents (376) have polynomial size (in n) cut-free* **PTK** *proofs.*

(b) *Let $\vec{p}$ denote $p_0, \ldots, p_{m-1}$. The following sequents have polynomial-size (in m) cut-free* **PTK** *proofs:*

(378)

$$p_m \longrightarrow \neg \mathsf{Th}_2(\vec{p}, p_m) \vee \left(\bigvee_{k=1}^{m-1} \big(\mathsf{Th}_k(\vec{p}) \wedge \neg \mathsf{Th}_{k+2}(\vec{p}, p_m)\big) \right) \vee \mathsf{Th}_m(\vec{p})$$

Proof. (a) The cut-free **PTK** proof is as follows, recall that for readability we use $A \wedge B$ for $\mathsf{Th}_2(A, B)$ and $\vee$ for Th_1:

$$
\begin{array}{cl}
\mathsf{Th}_{n-1}(\vec{p}) \longrightarrow \mathsf{Th}_{n-1}(\vec{p}) & \\ \hline
\longrightarrow \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & (7) \\ \hline\hline
\mathsf{Th}_{n-2}(\vec{p}) \longrightarrow \mathsf{Th}_{n-2}(\vec{p}) \wedge \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & (6) \\
\vdots & \\
\longrightarrow \neg\mathsf{Th}_3(\vec{p}), \mathsf{Th}_3(\vec{p}) \wedge \neg\mathsf{Th}_4(\vec{p}), \dots, \mathsf{Th}_{n-2}(\vec{p}) \wedge \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & \\ \hline\hline
\mathsf{Th}_2(\vec{p}) \longrightarrow \mathsf{Th}_2(\vec{p}) \wedge \neg\mathsf{Th}_3(\vec{p}), \dots, \mathsf{Th}_{n-2}(\vec{p}) \wedge \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & (5) \\ \hline
\longrightarrow \neg\mathsf{Th}_2(\vec{p}), \mathsf{Th}_2(\vec{p}) \wedge \neg\mathsf{Th}_3(\vec{p}), \dots, \mathsf{Th}_{n-2}(\vec{p}) \wedge \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & (4) \\ \hline\hline
\mathsf{Th}_1(\vec{p}) \longrightarrow \mathsf{Th}_1(\vec{p}) \wedge \neg\mathsf{Th}_2(\vec{p}), \dots, \mathsf{Th}_{n-2}(\vec{p}) \wedge \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & (3) \\ \hline
\longrightarrow \neg\mathsf{Th}_1(\vec{p}), \mathsf{Th}_1(\vec{p}) \wedge \neg\mathsf{Th}_2(\vec{p}), \dots, \mathsf{Th}_{n-2}(\vec{p}) \wedge \neg\mathsf{Th}_{n-1}(\vec{p}), \mathsf{Th}_{n-1}(\vec{p}) & (2) \\ \hline
\longrightarrow \neg\mathsf{Th}_1(\vec{p}) \vee \left(\bigvee_{k=1}^{n-2} (\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+1}(\vec{p})) \right) \vee \mathsf{Th}_{n-1}(\vec{p}) & (1)
\end{array}
$$

Here the top sequent is an axiom, (1) is by the rule one-right, (2, 4, 7) are $\neg$-right, and the derivations (3, 5, 6) consist of the rule all-right and a derivation from the axiom of the form

$$\mathsf{Th}_i(\vec{p}) \longrightarrow \mathsf{Th}_i(\vec{p})$$

(b) The **PTK** proof is presented below. Our convention is to read the proof from bottom up (starting from the inference (1) below). Because of the space limit, we will give one fragment of the proof at a time. There are $(m+1)$ fragments. The top fragment is:

$$
\begin{array}{cl}
p_m, \mathsf{Th}_{m+1}(\vec{p}, p_m) \longrightarrow \mathsf{Th}_m(\vec{p}) & \\ \hline
p_m, \longrightarrow \neg\mathsf{Th}_{m+1}(\vec{p}, p_m), \mathsf{Th}_m(\vec{p}) & (10) \\ \hline\hline
p_m, \mathsf{Th}_{m-1}(\vec{p}) \longrightarrow \mathsf{Th}_{m-1}(\vec{p}) \wedge \neg\mathsf{Th}_{m+1}(\vec{p}, p_m), \mathsf{Th}_m(\vec{p}) & (9)
\end{array}
$$

The two bottom fragments are:

$$
\begin{array}{cl}
\begin{array}{cc}
 & p_m, \mathsf{Th}_2(\vec{p}) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=2}^{m-1}, \mathsf{Th}_m(\vec{p}) \\ \cline{2-2}
\mathcal{S}_2 & p_m, p_m, \mathsf{Th}_2(\vec{p}) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=2}^{m-1}, \mathsf{Th}_m(\vec{p}) \;(8)
\end{array} & \\ \hline
p_m, \mathsf{Th}_3(\vec{p}, p_m) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=2}^{m-1}, \mathsf{Th}_m(\vec{p}) & (7) \\ \hline
p_m \longrightarrow \neg\mathsf{Th}_3(\vec{p}, p_m), \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=2}^{m-1}, \mathsf{Th}_m(\vec{p}) & (6) \\ \hline\hline
p_m, \mathsf{Th}_1(\vec{p}) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=1}^{m-1}, \mathsf{Th}_m(\vec{p}) & (5)
\end{array}
$$

and

$$
\dfrac{\dfrac{\dfrac{\mathcal{S}_1 \qquad \dfrac{p_m, \mathsf{Th}_1(\vec{p}) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=1}^{m-1}, \mathsf{Th}_m(\vec{p})}{p_m, p_m, \mathsf{Th}_1(\vec{p}) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=1}^{m-1}, \mathsf{Th}_m(\vec{p})}\,(4)}{p_m, \mathsf{Th}_2(\vec{p}, p_m) \longrightarrow \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=1}^{m-1}, \mathsf{Th}_m(\vec{p})}\,(3)}{p_m \longrightarrow \neg\mathsf{Th}_2(\vec{p}, p_m), \{\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\}_{k=1}^{m-1}, \mathsf{Th}_m(\vec{p})}\,(2)}{p_m \longrightarrow \neg\mathsf{Th}_2(\vec{p}, p_m) \vee \left(\bigvee_{k=1}^{m-1} \big(\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\big) \right) \vee \mathsf{Th}_m(\vec{p})}\,(1)
$$

From the bottom up: (1) is by the rule one-right, (2) is $\neg$-right. The sequent $\mathcal{S}_1$ is the top sequent in (8), and (3) is Th_2-left. The rule (4) is contraction left. The derivation (5) consists of an $\wedge$-right and a derivation by weakenings from the axiom

$$\mathsf{Th}_1(\vec{p}) \longrightarrow \mathsf{Th}_1(\vec{p})$$

The other steps (6, 7, etc.) are similar. Finally, the top sequent of (10) is obtained from some axioms by the rules all-left and all-right. $\dashv$

Lemma 10.76. *The translations of the defining axioms* (371), (372) *and* (373) *for numones′ (described in Example 10.74) have polynomial size* d-**PTK** *proofs, for some constant* d.

Proof. The translation of (371) is $\top$, so the conclusion is obvious. Consider the translations of the defining axiom (372) in part (b) of Example 10.74). Recall the formulas $\varphi[m;n]$ and $\varphi[n-1;n]$ from (375) and (376), respectively. We need to show that the following sequents have polynomial size d-**PTK** proofs, for some d:

$$\longrightarrow \neg p_m \vee \varphi[m;n] \qquad \text{and} \qquad \longrightarrow \varphi[n-1;n]$$

By Lemma 10.75 (a) the latter has a polynomial size cut-free **PTK** proof. To derive the former, by Lemma 10.75 (b) it suffices to derive

(379)

$$p_m, \neg\mathsf{Th}_2(\vec{p}, p_m) \vee \left(\bigvee_{k=1}^{m-1} \big(\mathsf{Th}_k(\vec{p}) \wedge \neg\mathsf{Th}_{k+2}(\vec{p}, p_m)\big) \right) \vee \mathsf{Th}_m(\vec{p}) \longrightarrow \varphi[m;n]$$

(where $\vec{p}$ denotes $p_0, p_1, \ldots, p_{m-1}$). This is left as an exercise (see below).

Finally consider the translation of axiom (373) described in Example 10.74 (c). As mentioned above, the sequents (376) have polynomial size cut-free **PTK** proofs. It remains to show that (recall $\psi[m;n]$ from (377)):

$$\longrightarrow p_m \vee \psi[m;n] \tag{380}$$

has polynomial size d-**PTK** proof, for some constant d. This is left as an exercise. $\dashv$

Exercise 10.77. *Complete the proof of Lemma 10.76 above by showing that the sequents* (379) *and* (380) *have polynomial size d-**PTK** proofs, for some constant d. Hint: first deriving the following sequents, then use Lemma 10.75:*

1) $p_m, \mathsf{Th}_k(\vec{p}) \longrightarrow \mathsf{Th}_{k+1}(\vec{p}, p_m)$ *(for $1 \leq k \leq m$).*
2) $p_m, \neg\mathsf{Th}_{k+2}(\vec{p}, p_m) \longrightarrow \neg\mathsf{Th}_{k+1}(\vec{p})$ *(for $0 \leq k \leq m-1$).*
3) $\mathsf{Th}_k(\vec{p}) \longrightarrow p_m, \mathsf{Th}_k(\vec{p}, p_m)$ *for $1 \leq k \leq m$.*
4) $\neg\mathsf{Th}_{k+1}(\vec{p}) \longrightarrow p_m, \neg\mathsf{Th}_{k+1}(\vec{p}, p_m)$ *for $0 \leq k \leq m-1$.*

As in Section 10A.1, it can be shown that formulas, sequents and proofs of **PTK** are Δ_1^B-definable in $\mathbf{FTC}^0$.

Lemma 10.78. *For every $\Sigma_0^B(numones')$ formula $\varphi(\vec{x}, \vec{X})$, there is a constant d and a polynomial $\mathbf{p}(\vec{m}, \vec{n})$ so that for all sequences $\vec{m}, \vec{n}$, the propositional formula $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ has depth d and size bounded by $\mathbf{p}(\vec{m}, \vec{n})$. Moreover, $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ is computable by an $\mathbf{FTC}^0$ function $G(\vec{m}, \vec{n})$.*

Proof. By structural induction on φ. ⊣

Theorem 10.79. *Suppose that $\varphi(\vec{x}, \vec{X})$ is a $\Sigma_0^B(numones')$ theorem of $\mathbf{V}^0(numones')$. Then there are a constant $d \in \mathbb{N}$ and an $\mathbf{FTC}^0$ function $F(\vec{m}, \vec{n})$ so that, provably in $\overline{\mathbf{VTC}}^0$, $F(\vec{m}, \vec{n})$ is a d-**PTK** proof of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$, for all $\vec{m}$, $\vec{n}$.*

Corollary 10.80. *For every Σ_0^B theorem $\varphi(\vec{x}, \vec{X})$ of $\mathbf{VTC}^0$, there are a constant $d \in \mathbb{N}$ and an $\mathbf{FTC}^0$ function $F(\vec{m}, \vec{n})$ so that, provably in $\overline{\mathbf{VTC}}^0$, $F(\vec{m}, \vec{n})$ is a d-**PTK** proof of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$, for all $\vec{m}$, $\vec{n}$.*

Corollary 10.80 follows from Theorem 10.79 because $\mathbf{V}^0(numones')$ is a conservative extension of $\mathbf{VTC}^0$. Theorem 10.79 can be proved in the same way that Theorem 7.20 is proved in Section 7B.3, i.e., by showing that every $\Sigma_0^B(numones')$ theorem of $\mathbf{V}^0(numones')$ has an $\mathbf{LK}^2$ proof where the inference rule $\Sigma_0^B(numones')$-**IND** (Definition 6.38) is allowed.

In the next section we introduce the quantified threshold formulas and a sequent calculus **GTC** for them. Theorem 10.79 will be generalized to show that the propositional translations of *bounded* theorems of $\mathbf{VTC}^0$ have proofs in **GTC** that are provably in $\mathbf{VTC}^0$ computable by some $\mathbf{FTC}^0$ functions (Theorem 10.82).

10D.3. Bounded Depth $\mathbf{GTC}_0$. Now we consider an extension of **PTK** which allows quantifiers over propositional variables. We require that the quantifiers do not occur inside the scope of the threshold connectives Th_k. So here will will use the binary connectives $\wedge, \vee$ that are used solely for quantified formulas. Formally, *quantified threshold formulas* (or QT formulas, or just formulas) are defined as follows:

(a) Any **PTK** formula is a QT formula;

(b) If $A(p)$ is a QT formula, then so are $\forall x A(x)$ and $\exists x A(x)$, for any free variable p and bound variable x.

(c) If A and B are non-**PTK** formulas, so are $(A \wedge B)$, $(A \vee B)$, $\neg A$;

The system **GTC** is the extension of **PTK** where the axioms now consist of

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

for all QT formulas A. The introduction rules for **PTK** given in Section 10D.1 are for **PTK** formulas only. The introduction rules for $\exists$ and $\forall$ can be applied for every formulas, but the rules for $\vee$ and $\wedge$ can be applied only to quantified formulas.

Theorem 10.69 can be extended to show that **GTC** and **G** are p-equivalent. In fact, for $i \geq 0$ define Σ_i^{qt} and Π_i^{qt} of QT formulas in the same way as Σ_i^q and Π_i^q, and let $\mathbf{GTC}_i$ be obtained from **GTC** by restricting the cut formulas to $\Sigma_i^{qt} \cup \Pi_i^{qt}$. Then it can be shown that $\mathbf{GTC}_i$ and $\mathbf{G}_i$ are p-equivalent for $i \geq 0$. Here we are interested in the following subsystems of $\mathbf{GTC}_0$.

Definition 10.81 (Bounded Depth $\mathbf{GTC}_0$). *For each $d \in \mathbb{N}$, d-$\mathbf{GTC}_0$ is the subsystem of* **GTC** *where all cut formulas are quantifier-free and have depth at most d. A* bounded depth $\mathbf{GTC}_0$ *(or just* $\mathbf{bGTC}_0$*) system is any system d-$\mathbf{GTC}_0$ for $d \in \mathbb{N}$. Treelike d-$\mathbf{GTC}_0$ (resp. treelike* $\mathbf{bGTC}_0$*) is denoted by d-$\mathbf{GTC}_0^\star$ (resp.* $\mathbf{bGTC}_0^\star$*).*

As in Section 10A.1, it can be shown that formulas, sequents and proofs in **GTC** are Δ_1^B-definable in $\mathbf{VTC}^0$. It is also straightforward to extend the translation given in Section 10D.2 so that $\Sigma_i^B(\mathit{numones}')$ and $\Pi_i^B(\mathit{numones}')$ formulas (for $i \geq 1$) are translated into quantified threshold formulas in Σ_i^{qt} and Π_i^{qt}, respectively.

Theorem 10.82 (Propositional Translation Theorem for $\mathbf{V}^0(\mathit{numones}')$). *Let $\varphi(\vec{x}, \vec{X})$ be a bounded theorem of $\mathbf{V}^0(\mathit{numones}')$. There is a constant $d \in \mathbb{N}$ and a function F in $\mathbf{FTC}^0$ so that $F(\vec{m}, \vec{n})$ is provably in $\overline{\mathbf{VTC}}^0$ a d-$\mathbf{GTC}_0^\star$ proof of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$, for all $\vec{m}$, $\vec{n}$.*

Corollary 10.83 (Propositional Translation Theorem for $\mathbf{VTC}^0$). *For every bounded theorem $\varphi(\vec{x}, \vec{X})$ of $\mathbf{VTC}^0$, there is a constant $d \in \mathbb{N}$ and an $\mathbf{FTC}^0$ function F such that $\overline{\mathbf{VTC}}^0$ proves that for all $\vec{m}$ and $\vec{n}$, $F(\vec{m}, \vec{n})$ is a d-$\mathbf{GTC}_0^\star$ proof of $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$.*

Proof. Since φ is a theorem of $\mathbf{VTC}^0$, by Lemma 10.73 it is also a theorem of $\mathbf{V}^0(\mathit{numones}')$. Now apply Theorem 10.82. $\dashv$

The proof of Theorem 10.82 is similar to the proof of Theorem 10.16. Here we translate cut $\Sigma_0^B(\mathit{numones})$-**COMP** formulas in the same way that cut Σ_0^B-**COMP** formulas are translated in Theorem 7.61. Then it can be shown that the translation of formulas in an $\mathbf{LK}^2$-$\mathbf{V}^0(\mathit{numones}')$ are provably in $\overline{\mathbf{VTC}}^0$ computable by some $\mathbf{FTC}^0$ functions. Furthermore, the **PTK** version of the sequent in Exercise 7.58 can be shown to have d-$\mathbf{PTK}^\star$ proofs that are provably in $\overline{\mathbf{VTC}}^0$ computable by some $\mathbf{FTC}^0$ function, for some constant d. Details are left as an exercise.

Exercise 10.84. *Proof Theorem 10.82.*

10E. Notes

The results in Section 10A.3 are from [56].

The idea of using the Reflection Principle for p-simulation is from [28] where a variant of Exercise 10.54 is proved. Theorem 10.38 and Corollary 10.41 are from [56]. Theorem 10.51 is a simple case of a result from [70]. Theorem 10.57 is from [61].

Several **ALogTime** algorithms for the Boolean Sentence Value Problem have been shown by Buss [14, 16, 17]. The algorithm presented in the proof of Theorem 10.62 is from [17]. The algorithm from [16] has been formalized in [5]. The algorithm from [17] has also been formalized in [71] using the string theory T^1.

Chapter 11

COMPUTATION MODELS

This version is preliminary and incomplete

In this Appendix, the functions f, g are used for functions from the natural numbers to $\mathbb{R}_{\geq 0} = \{x \in \mathbb{R} : x \geq 0\}$. We will use the following notations.

- $g \in \mathcal{O}(f)$ if there is a constant $c > 0$ so that $g(n) \leq cf(n)$ for all but finitely many n.
- $g \in \Omega(f)$ if there is a constant $c > 0$ so that $g(n) \geq cf(n)$ for all but finitely many n.
- $\log n$ stands for $\log_2 n$. When $\log n$ is required to be an integer, it is understood that it takes the value $\lceil \log_2 n \rceil$.

Appendix A. Deterministic Turing Machines

A k–tape deterministic Turing machine (DTM) consists of k two–way infinite tapes and a finite state control. Each tape is divided into squares, each of which holds a symbol from a finite alphabet Γ. Each tape also has a read/write head that is connected to the control and that scans the squares on the tape. Depending on the state of the control and the symbols scanned, the machine makes a move which consists of

1) printing a symbol on each tape;
2) moving each head left or right one square;
3) assuming a new state.

Definition 11.1. *For a natural number $k \geq 1$, a k–tape DTM M is specified by a tuple $\langle \mathsf{Q}, \Sigma, \Gamma, \sigma \rangle$ where*

1) Q *is the finite set of* states. *There are 3 distinct designated states $\mathsf{q}_{initial}$ (the initial state), q_{accept} and q_{reject} (the states in which M halts).*
2) Σ *is the finite, non-empty set of input symbols.*
3) Γ *is the finite set of working symbols, $\Sigma \subset \Gamma$. Γ contains a special symbol $\not b$ (read "blank"), and $\not b \in \Gamma \setminus \Sigma$.*

4) *σ is the transition funtion, i.e., a total function:*

$$\sigma : ((\mathsf{Q} \setminus \{\mathsf{q}_{accept}, \mathsf{q}_{reject}\}) \times \Gamma^k) \to (\mathsf{Q} \times (\Gamma \times \{\mathsf{L}, \mathsf{R}\})^k)$$

If the current state is q, the current symbols being scanned are $s_1, \ldots, s_k$, and $\sigma(q, \vec{s}) = (q', s'_1, h_1, \ldots, s'_k, h_k)$, then q' is the new state, $\vec{s'}$ are the symbols printed, and for $1 \leq i \leq k$, the head of the ith tape will move one square to the left or right depending on whether $h_i = \mathsf{L}$ or $h_i = \mathsf{R}$.

On an input x (a finite string of Σ symbols) the machine M works as follows. Initially, the input is given on tape 1, called the *input tape*, which is completely blank everywhere else. Other tapes (i.e., the *work tapes*) are blank, and their heads point to some squares. Also the input tape head is pointing to the leftmost symbol of the input (if the input is the empty string, then the input tape will be completely blank, and its head will point to some square). The control is initially in state $\mathsf{q}_{initial}$. Then M moves according to the transition function σ.

If M enters either q_{accept} or q_{reject} then it halts. If M halts in q_{accept} we say that it accepts the input x, if it halts in q_{reject} then we say that it rejects x. Note that it is possible that M never halts on some input. Let Σ^* denote the set of all finite strings of Σ symbols. We say that M accepts a language $L \subseteq \Sigma^*$ if M accepts input $x \in \Sigma^*$ iff $x \in L$. We let $L(\mathsf{M})$ denote the language accepted by M.

Unless specified otherwise, Turing machines are multi-tape (i.e., $k > 1$). In this case we require that the input tape head is read-only. Also, for a Turing machine M to compute a (partial) function, tape 2 is called the *output tape* and the content of the output tape when the machine halts in q_{accept} is the output of the machine. For machines that compute a function we require that the output tape is write-only.

A *configuration* of M is a tuple $\langle q, u_1, v_1, \ldots, u_k, v_k\rangle \in \mathsf{Q} \times (\Gamma^* \times \Gamma^*)^k$. The intuition is that q is the current state of the control, the string $u_i v_i$ is the content of the tape i, and the head of tape i is on the left-most symbol of v_i. If both u_i and v_i are the empty string, then the head points to a blank square. If only v_i is the empty string then the head points to the left-most blank symbol to the right of u_i. We require that for each i, u_i does not start with the blank symbol $\not{b}$, and v_i does not end with $\not{b}$.

The *computation* of M on an input x is the (possibly infinite) sequence of configurations of M, starting with the *initial configuration* $\langle \mathsf{q}_{initial}, \epsilon, x, \epsilon, \epsilon, \ldots, \epsilon, \epsilon\rangle$, where ϵ is the empty string, and each subsequent configuration is obtained from the previous one as specified by the transition function σ. Note that the sequence can contain at most one *final configuration*, i.e., a configuration of the form $\langle \mathsf{q}_{accept}, \ldots\rangle$ or $\langle \mathsf{q}_{reject}, \ldots\rangle$. The sequence contains a final configuration iff it is finite iff M halts on x. The *length* of the computation is the length of the sequence.

A.1. L, P, PSPACE and EXP. Suppose that a Turing machine $\mathsf{M} = \langle \mathsf{Q}, \Sigma, \Gamma, \sigma \rangle$ halts on input x. Then the running time of M on x, denoted by $time_{\mathsf{M}}(x)$, is the number of moves that M makes before halting (i.e., the number of configurations in the computation of M on x). Otherwise we let $time_{\mathsf{M}}(x) = \infty$.

Recall that $L(\mathsf{M})$ denotes the language accepted by M. We say that M *runs in time* $f(n)$ if for all but finitely many $x \in L(\mathsf{M})$, $time_{\mathsf{M}}(x) \leq f(|x|)$, where $|x|$ denotes the length of x. In this case we also say that M *accepts the language* $L(\mathsf{M})$ *in time* $f(n)$.

Definition 11.2 (**DTime**). *For a function* $f(n)$*, define*

$$\mathbf{DTime}(f) = \{L : \textit{there is a DTM accepting } L \textit{ in time } f(n)\}$$

In general, if f is at least linear, then the class $\mathbf{DTime}(f)$ is robust in the following sense.

Theorem 11.3 (Speed-up Theorem). *For any* $\epsilon > 0$,

$$\mathbf{DTime}(f) \subseteq \mathbf{DTime}((1+\epsilon)n + \epsilon f).$$

The classes of polynomial time and exponential time computable languages are defined as follows.

Definition 11.4 (**P** and **EXP**).

$$\mathbf{P} = \bigcup_{k \geq 1} \mathbf{DTime}(n^k), \qquad \mathbf{EXP} = \bigcup_{k \geq 1} \mathbf{DTime}(2^{n^k})$$

The working space of a (multi-tape) DTM M on input x, denoted by $space_{\mathsf{M}}(x)$, is the total number of squares on the *work tapes* (i.e., excluding the input and output tapes) that M visits at least once during the computation. Note that it is possible that $space_{\mathsf{M}}(x) = \infty$, and also that $space_{\mathsf{M}}(x)$ can be finite even if M does not halt on x.

We say that M *runs in space* $f(n)$ if for all but finitely many $x \in L(\mathsf{M})$, $space_{\mathsf{M}}(x) \leq f(|x|)$. In this case we also say that M *accepts the language* $L(\mathsf{M})$ *in space* $f(n)$.

Definition 11.5 (**DSpace**). *For a function* $f(n)$*, define*

$$\mathbf{DSpace}(f) = \{L : \textit{there is a DTM accepting } L \textit{ in space } f(n)\}$$

Theorem 11.6 (Tape Compression Theorem). *For any* $\epsilon > 0$ *and any function* f,

$$\mathbf{DSpace}(\epsilon f) = \mathbf{DSpace}(f)$$

The class of languages computable in logarithmic and polynomial space are defined as follows.

Definition 11.7 (**L** and **PSPACE**).

$$\mathbf{L} = \mathbf{DSpace}(\log n), \qquad \mathbf{PSPACE} = \bigcup_{k \geq 1} \mathbf{DSpace}(n^k)$$

For a single-tape Turing machine, the working space is the total number of squares visited by the tape head during the computation. The classes **P**, **PSPACE** and **EXP** remain the same even if we restrict to single-tape DTMs. This is due to the following theorem.

THEOREM 11.8 (Multi Tape Theorem). *For each multi-tape Turing machine* M *that runs in time* $t(n)$ *and space* $s(n)$, *there is a single–tape Turing machine* M' *that runs in time* $(t(n))^2$ *and space* $max\{n, s(n)\}$ *and accepts the same language as* M. *There exists also a 2–tape Turing machine* M'' *that works in space* $s(n)$ *and accepts* $L(\mathsf{M})$.

For the Time Hierarchy Theorem below we need the notion of *time constructible functions*. A function $f(n)$ is time constructible if there is a Turing machine M such that on all input x, the running time of M is *exactly* $f(|x|)$. We will be concerned only with time bounding functions that are constructible.

THEOREM 11.9 (Time Hierarchy Theorem). *Suppose that* $f(n)$ *is a function,* $f(n) \geq n$, *and* $g(n)$ *is a time constructible function so that*

$$\lim_{n\to\infty} \inf \frac{f(n)\log f(n)}{g(n)} = 0.$$

Then

$$\mathbf{DTime}(g) \setminus \mathbf{DTime}(f) \neq \emptyset$$

It is easy to see that

$$\mathbf{L} \subseteq \mathbf{P} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}. \tag{381}$$

The Time Hierarchy Theorem shows that

$$\mathbf{DTime}(n) \subsetneq \mathbf{DTime}(n^2) \subsetneq \dots \qquad \text{and} \qquad \mathbf{P} \subsetneq \mathbf{DTime}(2^{\epsilon n})$$

for any $\epsilon > 0$. Also, the Space Hierarchy Theorem (below) shows that $\mathbf{L} \subsetneq \mathbf{PSPACE}$. However, none of the immediate inclusions in (381) is known to be proper.

A function $f(n)$ is *space constructible* if there is Turing machine M such that on all input x, the working space of M is *exactly* $f(|x|)$. The space bounds that we are interested in are all constructible.

THEOREM 11.10 (Space Hierarchy Theorem). *Suppose that* $f(n)$ *is a function and* $g(n)$ *is a space constructible function so that*

$$g(n) = \Omega(\log n) \qquad \textit{and} \qquad \lim_{n\to\infty} \inf \frac{f(n)}{g(n)} = 0.$$

Then

$$\mathbf{DSpace}(g) \setminus \mathbf{DSpace}(f) \neq \emptyset$$

Appendix B. Nondeterministic Turing Machines

DEFINITION 11.11. *A k–tape nondeterministic Turing machine (NTM) is specified by a tuple $\langle \mathsf{Q}, \Sigma, \Gamma, \sigma\rangle$ as in Definition 11.1, but now the transition function σ is of the form*

$$\sigma : ((\mathsf{Q} \setminus \{\mathsf{q}_{accept}, \mathsf{q}_{reject}\}) \times \Gamma^k) \to \mathcal{P}(\mathsf{Q} \times (\Gamma \times \{\mathsf{L}, \mathsf{R}\})^k)$$

where $\mathcal{P}(S)$ denotes the power set *of the set S.*

Here $\sigma(q, s_1, \ldots, s_k)$ is the (possibly empty) set of possible moves of M, given that the current state is q and the symbols currently being scanned are $\vec{s}$.

A computation of M on an input x is a (possibly infinite) sequence of configurations of M, starting with the initial configuration

$$\langle \mathsf{q}_{initial}, \epsilon, x, \epsilon, \epsilon, \ldots, \epsilon, \epsilon\rangle$$

and each subsequent configuration is a configuration that can be obtained from the previous one by one of the possible moves specified by σ. By definition, each computation of M may contain at most one configuration of the form $\langle \mathsf{q}_{accept}, \ldots\rangle$ or $\langle \mathsf{q}_{reject}, \ldots\rangle$. In the former case we say that it is an *accepting computation*, and in the latter case we say that it is a *rejecting computation*.

We say that the NTM M accepts x is there is an accepting computation of M on x. We say that M accepts x *in time* $f(n)$ if there is an accepting computation of length $\leq f(|x|)$, and M accepts x *in space* $f(n)$ if there is an accepting computation such that the number of squares on the work tapes used by M during this computation is $\leq f(n)$.

As for DTMs, if for all but finitely many $x \in L(\mathsf{M})$ the NTM M accepts x in time/space $f(n)$, we also say that M accepts the language $L(\mathsf{M})$ in time/space $f(n)$.

DEFINITION 11.12 (**NTime** and **NSpace**). *For a function $f(n)$, define*

$$\mathbf{NTime}(f) = \{L : \textit{there is a NTM accepting } L \textit{ in time } f(n)\}$$
$$\mathbf{NSpace}(f) = \{L : \textit{there is a NTM accepting } L \textit{ in space } f(n)\}$$

The Speed-up Theorem (11.3) and Tape Compression Theorem (11.6) continue to hold for NTMs.

DEFINITION 11.13 (**NP** and **NL**).

$$\mathbf{NP} = \bigcup_{k \geq 1} \mathbf{NTime}(n^k), \qquad \mathbf{NL} = \mathbf{NSpace}(\log n)$$

The list in (381) is extended as follows:

$$\mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}$$

However, it is not known whether any of the immediate inclusions is proper.

For a class $\mathbf{C}$ of languages, we define *co*-$\mathbf{C}$ to be the class of the complements of the languages in $\mathbf{C}$. It is also easy to see that

$$\mathbf{P} \subseteq co\text{-}\mathbf{NP} \subseteq \mathbf{PSPACE}$$

But the questions $\mathbf{P} \stackrel{?}{=} co\text{-}\mathbf{NP}$, $\mathbf{NP} \stackrel{?}{=} co\text{-}\mathbf{NP}$ and $co\text{-}\mathbf{NP} \stackrel{?}{=} \mathbf{PSPACE}$ are open.

For $\mathbf{NL}$ vs. $co\text{-}\mathbf{NL}$ we have an affirmative answer, due to Immerman and Szelepcsényi:

Theorem 11.14 (Immerman–Szelepcsényi Theorem). *For any space constructible function $f(n) \geq \log(n)$, $\mathbf{NSpace}(f) = co\text{-}\mathbf{NSpace}(f)$.*

The class of languages computable by NTMs in polynomial space is defined similarly, but by Savitch's Theorem (below), this is the same as **PSPACE**.

Theorem 11.15 (Savitch's Theorem). *For any space constructible function $f(n) \geq \log n$,*

$$\mathbf{NSpace}(f) \subseteq \mathbf{DSpace}(f^2)$$

It follows that nondeterministic polynomial space is the same as **PSPACE**, and also that $\mathbf{NL} \subsetneq \mathbf{PSPACE}$.

Appendix C. Oracle Turing Machines

Let L be a language. An Oracle Turing machine (OTM) M with oracle L is a Turing machine augmented with the ability to ask questions of the form "is $y \in L$". Formally, M has a designated write-only tape for the queries, called the *query tape*. It also has 3 additional states, namely $\mathsf{q}_{query}, \mathsf{q}_{Yes}$ and q_{No}. In order to ask the question "is $y \in \mathbf{L}$", the machine writes the string y on the query tape, and enters the state q_{query}. The next state of M is then either q_{Yes} or q_{No}, depending on whether $y \in L$. Also the query tape is blanked out before M makes the next move.

In case the queries are witnessed (e.g., Definition 8.98) or we want a function oracle, i.e., oracles that answer queries of the form

$$F(W)?$$

for some function F, then the OTM will have an *answer tape* that contains oracle replies (with the tape head positioned to the left-most non-blank square) whenever the machine enters the state q_{query}.

The running time of M on an input x is defined as before. Note that the time it takes to write down the queries (and read the oracle answers/witnesses) are counted. Thus an OTM running in polynomial time cannot ask long (e.g., exponentially long) queries. Note also that it takes only 1 move to get the answer from the oracle.

A nondeterministic oracle Turing machine (NOTM) is a generalization of OTM where the transition function is a many-valued function. For a language L, we denote by $\mathbf{P}(L)$ the class of languages accepted by some OTM running in polynomial time with L as the oracle, and similarly $\mathbf{NP}(L)$ the class of languages accepted by some NOTM running in polynomial time with L as the oracle. For a class $\mathbf{C}$ of languages, define

$$\mathbf{P}(\mathbf{C}) = \bigcup_{L \in \mathbf{C}} \mathbf{P}(L), \qquad \text{and} \qquad \mathbf{NP}(\mathbf{C}) = \bigcup_{L \in \mathbf{C}} \mathbf{NP}(L)$$

Then the polynomial time hierarchy (**PH**) is defined as follows.

Definition 11.16 (**PH**). $\Delta_0^p = \Sigma_0^p = \Pi_0^p = \mathbf{P}$. *For* $i \geq 0$,

$$\Sigma_{i+1}^p = \mathbf{NP}(\Sigma_i^p), \qquad \Pi_{i+1}^p = co\text{-}\Sigma_{i+1}^p, \qquad \Delta_{i+1}^p = \mathbf{P}(\Sigma_i^p)$$

And

$$\mathbf{PH} = \bigcup_{i \geq 0} \Sigma_i^p$$

It can be shown that $\mathbf{PH} \subseteq \mathbf{PSPACE}$, but the inclusion is not known to be proper. Also, it is not known whether the polynomial time hierarchy collapses.

The Linear Time Hierarchy (**LTH**) is defined analogously to **PH**. Here **LinTime** and **NLinTime** are the classes of languages accepted in linear time by respectively multi-tape DTMs and NTMs.

Definition 11.17 (**LinTime** and **NLinTime**).

$$\mathbf{LinTime} = \mathbf{DTime}(n), \qquad \mathbf{NLinTime} = \mathbf{NTime}(n),$$

The class **LinTime** is not as robust a class as **P**; for example it is plausible that a $(k+1)$–tape linear time DTM can accept a language not accepted by any k–tape linear time DTM. However it is not hard to see that **NLinTime** is more robust, in the sense that every language in this class can be accepted by a 2–tape linear time NTM.

For a class $\mathbf{C}$ of languages, let $\mathbf{NLinTime}(\mathbf{C})$ be the class of languages accepted by a linear time Oracle TM with oracle from $\mathbf{C}$. Then the Linear Time Hierarchy is defined as follows.

Definition 11.18 (**LTH**).

$$\Sigma_0^{lin} = \mathbf{LinTime}, \qquad \Sigma_{i+1}^{lin} = \mathbf{NLinTime}(\Sigma_i^{lin}) \text{ for } i \geq 0,$$

and

$$\mathbf{LTH} = \bigcup_{i \geq 0} \Sigma_i^{lin}$$

Both **PH** and **LTH** can be alternatively defined using the notion of *alternating Turing machines*, which we will define in the next section.

Appendix D. Alternating Turing Machines

An *alternating Turing machine* (ATM) M is defined as in Definition 11.11 for a nondeterministic Turing machine, but now the finite set $\mathsf{Q} \setminus \{\mathsf{q}_{accept}, \mathsf{q}_{reject}\}$ is partitioned into 2 disjoint sets of states, namely the set of $\exists$ states and the set of $\forall$ states.

If a configuration c_2 of M can be obtained from c_1 as specified by the transition function σ, we say that it is *a successor configuration* of c_1. An *existential* (resp. *universal*) configuration is a configuration of the form $\langle q, \ldots \rangle$ where q is an $\exists$-state (resp. a $\forall$-state).

We define the set of *accepting configurations* to be the smallest set of configurations that satisfies:

- a final configuration of the form $\langle \mathsf{q}_{accept}, ... \rangle$ is an accepting configuration (*a final accepting configuration*);
- an existential configuration is accepting iff at least one of its successor configuration is accepting;
- a universal configuration is accepting iff all of its successor configurations are accepting.

We say that M accepts x iff the initial configuration $\langle \mathsf{q}_{initial}, \epsilon, x, \epsilon, \epsilon, \ldots, \epsilon, \epsilon \rangle$ is an accepting configuration of M.

A computation of M on an input x is viewed as a tree T with leaves labelled with the configurations as follows:

- the root of T is labeled with the initial configuration of M on x;
- if n is an inner node of T labeled with a universal configuration c which has k successor configurations, then n has k children each labeled uniquely by a successor configuration of c;
- if n is an inner node of T labeled with an existential configuration c which has k successor configurations, then n has k' children where $1 \leq k' \leq k$, and each of children of n is labeled uniquely by a successor configuration of c.

A finite computation of M is called an *accepting computation* if all its leaves are labeled with a final accepting configuration.

We say that an ATM M accepts x in time t if there is an accepting computation of M where the paths from the root to any leaf has length $\leq t$. Aslo, M accepts $L = L(M)$ in time $f(n)$ if for all $x \in L$, M accepts x in time $f(|x|)$.

The alternation depth of a computation is the maximum number of blocks of states of the same type (i.e., existential or universal) along any path from the root to a leaf.

It can be seen that for $i \geq 1$, Σ_i^p is the class of languages accepted by a polytime ATM whose initial state $\mathsf{q}_{initial}$ is an existential state.

D.1. $\mathbf{NC}^1$ and $\mathbf{AC}^0$.

Definition 11.19 ($\mathbf{AC}^0$). *DEFINE* $\mathbf{AC}^0$ *USING CIRCUIT CLASS, CHANGING THIS DEFINITION MAY CHANGE DISCUSSION IN THE PROOF OF* Σ_0^B *REPRESENTATION THEOREM.*

Theorem 11.20 (Alternative Definition of $\mathbf{AC}^0$). $\mathbf{AC}^0 = \mathbf{LTH} \ldots$

ALSO LANGUAGES VS. RELATIONS: CODING OR TUPLES INTO A SINGLE STRING

ALSO FUNCTION CLASSES

REFERENCES

[1] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena, *PRIMES is in P*, ***Annals of Mathematics***, vol. 160 (2004), pp. 781–793.

[2] Miklos Ajtai, *Σ_1^1-formulae on finite structures*, ***Annals of Pure and Applied Logic***, vol. 24 (1983), pp. 1–48.

[3] ———, *The complexity of the pigeonhole principle*, ***Proceedings of the IEEE 29th Annual Symposium on Foundations of Computing***, 1988, pp. 346–355.

[4] Aleksandar Ignjatovic, *Delineating Classes of Computational Complexity via Second Order Theories with Weak Set Existence Principles*, ***Journal of Symbolic Logic***, vol. 60 (1995), pp. 103–121.

[5] Toshiyasu Arai, *A bounded arithmetic AID for Frege systems*, ***Annals of Pure and Applied Logic***, vol. 103 (2000), pp. 155–199.

[6] David A. Barrington, *Bounded-Width Polynomial-Size Branching Programs Recognizes Exactly Those Languages in* $\mathbf{NC}^1$, ***Proceedings of the 18th Annual ACM Symposium on Theory of Computing***, 1986, pp. 1–5.

[7] David A. Mix Barrington, Neil Immerman, and Howard Straubing, *On Uniformity within NC^1*, ***Journal of Computer and System Sciences***, vol. 41 (1990), pp. 274–306.

[8] Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák, *Exponential lower bounds for the pigeonhole principle*, ***Proceedings of the 24th Annual ACM Symposium on Theory of Computing***, 1992, pp. 200–220.

[9] Paul Beame and Toniann Pitassi, *Propositional Proof Complexity: Past, Present and Future*, ***Current trends in computer science entering the 21st century*** (G. Rozenberg G. Paun and A. Salomaa, editors), World Scientific Publishing, 2001, pp. 42–70.

[10] James Bennett, ***On spectra***, PhD thesis, Princeton University, Departmentof Mathematics, 1962.

[11] Maria Luisa Bonet, Toniann Pitassi, and Ran Raz, *On Interpolation and Automatization for Frege Systems*, ***SIAM Journal on Computing***, vol. 29 (2000), no. 6, pp. 1939–1967.

[12] Samuel Buss, ***Bounded arithmetic***, Bibliopolis, 1986.

[13] ———, *Polynomial size proofs of the propositional pigeonhole principle*, ***Journal of Symbolic Logic***, vol. 52 (1987), pp. 916–927.

[14] ———, *The Boolean formula value problem is in ALOGTIME*, ***Proceedings of the 19th Annual ACM Symposium on Theory of Computing***, 1987, pp. 123–131.

[15] ———, *Axiomatizations and conservation results for fragments of bounded arithmetic*, ***Logic and computation, proceedings of a workshop held at carnegie mellon university***, AMS Contemporary Mathematics (106), 1990, pp. 57–84.

[16] ———, *Propositional Consistency Proofs*, ***Annals of Pure and Applied Logic***, vol. 52 (1991), pp. 3–29.

[17] ———, *Algorithms for Boolean formula evaluation and for tree-contraction*, ***Arithmetic, proof theory, and computational complexity*** (Peter Clote and Jan Krajíček, editors), Oxford, 1993, pp. 95–115.

[18] ———, *Relating the bounded arithmetic and polynomial time hierarchies*, ***Annals of Pure and Applied Logic***, vol. 75 (1995), pp. 67–77.

[19] ———, *An Introduction to Proof Theory*, ***Handbook of proof theory*** (S. Buss, editor), Elsevier, 1998, Available on line at `www.math.ucsd.edu/~sbuss/ResearchWeb/HandbookProofTheory/`, pp. 1–78.

[20] ———, *First–Order Proof Theory of Arithmetic*, ***Handbook of proof theory*** (S. Buss, editor), Elsevier, 1998, Available on line at `www.math.ucsd.edu/~sbuss/ResearchWeb/HandbookProofTheory/`, pp. 79–147.

[21] Samuel Buss and Jan Krajíček, *An application of boolean complexity to separation problems in bounded arithmetic*, ***Proc. London Math. Soc.***, vol. 69(3) (1994), pp. 1–21.

[22] Samuel Buss, Jan Krajíček, and Gaisi Takeuti, *On Provably Total Functions in Bounded Arithmetic Theories R^i_3, U^i_2, and V^i_2*, ***Arithmetic, proof theory and computational complexity*** (Peter Clote and Jan Krajíček, editors), Oxford, 1993, pp. 116–161.

[23] Ashok K. Chandra, Larry Stockmeyer, and Uzi Vishkin, *Constant Depth Reducibility*, ***SIAM Journal on Computing***, vol. 13(2) (1984), pp. 423–439.

[24] Mario Chiari and Jan Krajíček, *Witnessing functions in bounded arithmetic and search problems*, ***Journal of Symbolic Logic***, vol. 63 (1998), pp. 1095–1115.

[25] Peter Clote, *Sequential, Machine-Independent Characterizations of the Parallel Complexity Classes AlogTIME, AC^k, NC^k and NC*, ***Feasible mathematics*** (S. R. Buss and P.J. Scott, editors), Birkhauser, 1990.

[26] ———, *On Polynomial Size Frege Proofs of Certain Combinatorial Principles*, ***Arithmetic, proof theory, and computational***

complexity (Peter Clote and Jan Krajíček, editors), Oxford, 1993, pp. 162–184.

[27] PETER CLOTE and GAISI TAKEUTI, *First Order Bounded Arithmetic and Small Boolean Circuit Complexity Classes*, ***Feasible mathematics ii*** (P. Clote and J. B. Remmel, editors), Birkhäuser, 1995.

[28] STEPHEN COOK, *Feasibly constructive proofs and the propositional calculus*, ***Proceedings of the 7th Annual ACM Symposium on Theory of computing***, (1975), pp. 83–97.

[29] ———, *Proof Complexity and Bounded Arithmetic*, Course Notes for CSC 2429S. `http://www.cs.toronto.edu/~sacook/`, 2002.

[30] ———, *Theories for Complexity Classes and Their Propositional Translations*, ***Complexity of computations and proofs*** (Jan Krajíček, editor), Quaderni di Matematica, 2005, pp. 175–227.

[31] STEPHEN COOK and ANTONINA KOLOKOLOVA, *A second-order system for polytime reasoning based on Grädel's theorem*, ***Annals of Pure and Applied Logic***, vol. 124 (2003), pp. 193–231.

[32] ———, *A Second-order Theory for NL*, ***Logic in computer science (lics)***, 2004.

[33] STEPHEN COOK and TSUYOSHI MORIOKA, *Quantified Propositional Calculus and a Second-Order Theory for* NC^1, ***Archive for Mathematical Logic*** , (2005), pp. 1–37, (to appear).

[34] STEPHEN COOK and ROBERT RECKHOW, *The relative efficiency of propositional proof systems*, ***Journal of Symbolic Logic***, vol. 44(1) (1979).

[35] STEPHEN COOK and NEIL THAPEN, *The Strength of Replacement in Weak Arithmetic*, ***Proc. 19th ieee symposium on logic in computer science***, 2004.

[36] ———, *The strength of replacement in weak arithmetic*, ***ACM Transactions on Computational Logic***, vol. 7(4) (2006), pp. 749–764.

[37] MARTIN DOWD, ***Propositional representation of arithmetic proofs***, PhD thesis, University of Toronto, Department of Computer Science, 1979.

[38] ERICH GRÄDEL", *Capturing complexity classes by fragments of second order logic*, vol. 101 (1992), pp. 35–57.

[39] RONALD FAGIN, ***Contributions to the model theory of finite structures***, PhD thesis, U. C. Berkeley, Departmentof Mathematics, 1973.

[40] M. FURST, J. B. SAXE, and M. SIPSER, *Parity, circuits and the polynomial-time hierarchy*, ***Math. Systems Theory***, vol. 17 (1984), pp. 13–27.

[41] PETR HÁJEK and PAVEL PUDLÁK, ***Metamathematics of first-order arithmetic***, Springer–Verlag, 1993.

[42] William Hess, Eric Allender, and David A. Mix Barrington, *Uniform Constant-Depth Threshold Circuits for Division and Iterated Multiplication*, vol. 65 (2002), pp. 695–716.

[43] Aleksander Ignjatovic and Phuong Nguyen, *Characterizing Polynomial Time Computable Functions Using Theories with Weak Set Existence Principles*, ***Computing: The australasian theory symposium***, Electronic Notes in Theoretical Computer Science, Volume 78, 2003.

[44] Neil Immerman, *Nondeterministic space is closed under complementation*, ***SIAM J. Comput.***, vol. 17 (1988), no. 5, pp. 935–938.

[45] ———, ***Descriptive Complexity***, Springer, 1999.

[46] Emil Jerábek, ***Weak pigeonhole principle, and randomized computation***, PhD thesis, Charles University in Prague, Faculty of Mathematics and Physics, 2004.

[47] Jan Johannsen, *Satisfiability problems complete for deterministic logarithmic space*, ***Stacs 2004, 21st annual symposium on theoretical aspects of computer science, proceedings*** (Volker Diekert and Michel Habib, editor), 2004.

[48] Jan Johannsen and Chris Pollett, *On Proofs about Threshold Circuits and Counting Hierarchies*, ***Proc. 13th ieee symposium on logic in computer science***, 1998, pp. 444–452.

[49] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis, *How easy is local search?*, ***Journal of Computer and System Sciences***, vol. 37 (1988), pp. 79–100.

[50] Richard M. Karp and Richard J. Lipton, *Turing machines that take advice*, ***L'Enseignement Mathematique***, vol. 30 (1982), pp. 255–273.

[51] Antonina Kolokolova, ***Systems of Bounded Arithmetic from Descriptive Complexity***, Ph.D. thesis, University of Toronto, 2004.

[52] Jan Krajíček, *On the number of steps in proofs*, ***Annals of Pure and Applied Logic***, vol. 41 (1989), pp. 153–178.

[53] ———, *Exponentiation and second-order bounded arithmetic*, ***Annals of Pure and Applied Logic***, vol. 48 (1990), pp. 261–276.

[54] ———, *Fragments of bounded arithmetic and bounded query classes*, ***Trans. AMS***, vol. 338(2) (1993), pp. 587–98.

[55] ———, ***Bounded arithmetic, propositional logic and computational complexity***, Cambridge University Press, 1995.

[56] Jan Krajíček and Pudlák, *Quantified Propositional Calculi and Fragments of Bounded Arithmetic*, ***Zeitschrift f. Mathematickal Logik u. Grundlagen d. Mathematik***, vol. 36 (1990), pp. 29–46.

[57] Jan Krajíček, Pavel Pudlák, and Jiri Sgall, *Interactive computations of optimal solutions*, ***Mathematical foundations of computer science*** (B. Rovan, editor), Lecture Notes in Computer

Science, no. 452, Springer-Verlag, 1990, pp. 48–60.

[58] JAN KRAJÍČEK, PAVEL PUDLÁK, and GAISI TAKEUTI, *Bounded Arithmetic and the Polynommial Hierarchy*, ***Annals of Pure and Applied Logic***, vol. 52 (1991), pp. 143–153.

[59] JAN KRAJÍČEK, ALAN SKELLEY, and NEIL THAPEN, *Np search problems in low fragments of bounded arithmetic*, ***Journal of Symbolic Logic***, vol. 72(2) (2007), pp. 649–672.

[60] JOHN C. LIND, *Computing in logarithmic space*, Technical Report 52, MAC Technical Memorandom, 1974.

[61] TSUYOSHI MORIOKA, ***Logical approaches to the complexity of search problems: Proof complexity, quantified propositional calcuclus, and bounded arithmetic***, PhD thesis, Univeristy of Toronto, Department of Computer Science, 2005.

[62] V. A. NEPOMNJAŠČIJ, *Rudimentary predicates and turing calculations*, ***Soviet Math. Dokl.***, vol. 11 (1970), no. 6, pp. 1462–1465.

[63] PHUONG NGUYEN, ***Bounded Reverse Mathematics***, Ph.D. thesis, University of Toronto, 2008, `http://www.cs.toronto.edu/~pnguyen/`.

[64] PHUONG NGUYEN and STEPHEN COOK, *Theory for* $\mathbf{TC}^0$ *and Other Small Complexity Classes*, ***Logical Methods in Computer Science***, (2005).

[65] ———, *The Complexity of Proving Discrete Jordan Curve Theorem*, ***Proc. 22nd ieee symposium on logic in computer science***, 2007, pp. 245–254.

[66] NOAM NISAN and AMNON TA-SHMA, *Symmetric logspace is closed under complement*, 1995, pp. 140–146.

[67] R. PARIKH, *Existence and feasibility in arithmetic*, ***J. Symb. Logic***, vol. 36 (1971), pp. 494–508.

[68] J. PARIS and A. WILKIE, *Counting problems in bounded arithmetic*, ***Methods in mathematical logic***, Lecture Notes in Mathematics, no. 1130, Springer, 1985, pp. 317–340.

[69] J.B. PARIS, W.G. HANDLEY, and A.J. WILKIE, *Characterizing some low arithmetic classes*, ***THEORY OF ALGORITHMS*** (L. Lovász and E. Smerédi, editor), Colloquia Mathematica Societatis Janos Bolyai, no. 44, North-Holland, 1985, pp. 353–365.

[70] STEVEN PERRON, ***Power of Non-Uniformity in Proof Complexity***, Ph.D. thesis, University of Toronto, 2008.

[71] FRANCOIS PITT, ***A Quantifier-Free String Theory* Alogtime *Reasoning***, Ph.D. thesis, University of Toronto, 2000.

[72] CHRIS POLLETT, *Structure and Definability in General Bounded Arithmetic Theories*, ***Annals of Pure and Applied Logic***, vol. 100 (1999), pp. 189–245.

[73] MICHAEL RABIN, *Digitalized signatures and public-key functions as intractable as factorization*, Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[74] ALEXANDER A. RAZBOROV, *An Equivalence between Second Order Bounded Domain Bounded Arithmetic and First Order Bounded Arithmetic*, ***Arithmetic, proof theory and computational complexity*** (Peter Clote and Jan Krajíček, editors), Oxford, 1993, pp. 247–277.

[75] ALEXANDER. A. RAZBOROV, *Bounded arithmetic and lower bounds in boolean complexity*, ***Feasible mathematics ii*** (P. Clote and J. Remmel, editors), Birkhauser, 1995, pp. 344–386.

[76] OMER REINGOLD, *Undirected ST-Connectivity in Log-Space*, ***Proceedings of the 37th annual acm symposium on theory of computing***, 2005, pp. 376–385.

[77] STEPHEN SIMPSON, ***Subsystems of second order arithmetic***, Springer, 1999.

[78] RAYMOND SMULLYAN, ***Theory of formal systems***, Princeton University Press, 1961.

[79] L. J. STOCKMEYER, *The polynomial-time hierarchy*, ***Theoretical Computer Science***, vol. 3 (1976), pp. 1–21.

[80] R. SZELEPCSÉNYI, *The method of forced enumeration for nondeterministic automata*, ***Acta Informatica***, vol. 26 (1988), pp. 279–284.

[81] GAISI TAKEUTI, *S^i_3 and $V^i_2(BD)$*, ***Archive for Mathematical Logic***, vol. 29 (1990), pp. 149–169.

[82] ———, *RSUV Isomorphism*, ***Arithmetic, proof theory and computational complexity*** (Peter Clote and Jan Krajíček, editors), Oxford, 1993, pp. 364–386.

[83] G. S. TSEITIN, *On the complexity of derivation in propositional calculus*, ***Studies in constructive mathematics and mathematical logic, part 2*** (A. O. Slisenko (Translated from Russian), editor), Consultants Bureau, New Yor, London, 1970, pp. 115–125.

[84] CELIA WRATHALL, *Rudimentary predicates and relative computation*, ***SIAM J. Computing***, vol. 7 (1978), pp. 194–209.

[85] D. ZAMBELLA, *Notes on polynomially bounded arithmetic*, ***Journal of Symbolic Logic***, vol. 61 (1996), no. 3, pp. 942–966.

[86] DOMENICO ZAMBELLA, *End Extensions of Models of Linearly Bounded Arithmetic*, ***Annals of Pure and Applied Logic***, vol. 88 (1997), pp. 263–277.

INDEX