Foundations of Proof Complexity: Bounded Arithmetic and Propositional Translations

Stephen Cook and Phuong Nguyen

October 9, 2006

Preface
(Preliminary Version)

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

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

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	$\mathbf{AC}^0$	$\mathbf{TC}^0$	$\mathbf{NC}^1$	$\mathbf{P}$	$\mathbf{PH}$	
theory	$\mathbf{V}^0$	$\mathbf{VTC}^0$	$\mathbf{VNC}^1$	$\mathbf{TV}^0$	$\mathbf{V}$	(2)
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.

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 $\langle P(i,j)\rangle$ $(0 \le i \le n,$ $0 \le j < n)$, 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

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

Ajtai 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 $P(i,j)$) 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 $\mathbf{TV}^0$ 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 $\mathbf{TV}^0$ 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 easy 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 $\mathbf{TV}^0$, 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 $\mathbf{TV}^0$, assuming that there is no probabilistic polynomial time factoring algorithm.

Propositional tautologies can be used to express universal theorems such as (3) 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.

Another advantage of our two-sorted approach is that the propositional translations of our theories are especially simple and elegant. These are done in the style of Paris and Wilkie, rather than the earlier and more cumbersome style introduced by Cook for the equational theory $\mathbf{PV}$.

Much of this book is based on course notes for a graduate course taught several times at the University of Toronto by the first author. The notes for the 2002 version are available online [**?**]. The prerequisites for the course and the book are some knowledge of both mathematical logic and complexity theory. There are exercises sprinkled throughout the text, which are intended both to supplement the material presented and to help the reader master the material.

The first two chapters provide a concise treatment of the required background in first-order logic, based on Gentzen's proof system $\mathbf{LK}$. An unusual feature is our treatment of anchored (or "free-cut-free") proofs. It is based on a completeness theorem for such proofs, as opposed to the usual syntactic cut-elimination theorem.

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 linear time hierarchy. The universal theory $\overline{\mathbf{I\Delta}}_0$ has function symbols for each of these functions, and forms a conservative extension of $\mathbf{I\Delta}_0$.

Chapter 4 introduces the syntax and intended semantics for the two-sorted theories, which will be used throughout the remaining chapters. Representation theorems are proved which state that formulas in the syntactic class $\mathbf{\Sigma}_0^B$ represent precisely the (two-sorted) $\mathbf{AC}^0$ relations, and for $i \geq 1$, formulas in $\mathbf{\Sigma}_i^B$ represent the relations in the i-th level of the polynomial hierarchy.

Chapter 5 introduces the two-sorted theory $\mathbf{V}^0$, 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 quantifiers in a $\mathbf{\Sigma}_1^1$-theorem of $\mathbf{V}^0$ can be witnessed by $\mathbf{AC}^0$-functions. Here the $\mathbf{\Sigma}_1^1$ theorems have all existential string quantifiers in front, which makes the proof 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 $\mathbf{P}$. $\mathbf{V}^1$ is the two-sorted version of Buss's theory $\mathbf{S}_2^1$. All polynomial time functions are definable in $\mathbf{V}^1$. This is shown two ways: by analyzing Turing machine computations and by using Cobham's characterization of these functions. The witnessing theorem for $\mathbf{V}^1$ is proved based on Cobham's theorem.

Chapter 7 gives a general definition of propositional proof system. The goal is to associate a proof system with each theory so that each $\mathbf{\Sigma}_0^B$ 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 for the quantified propositional calculus are defined, and it is shown how to translate bounded theorems of $\mathbf{V}^1$ to polynomial size families of proofs in the system $\mathbf{G}_1^\star$. 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 $\mathbf{TV}^0$ and its universal conservative extension $\mathbf{VPV}$ both appear to be weaker than $\mathbf{V}^1$, although they have the same $\mathbf{\Sigma}_1^B$ theorems as $\mathbf{V}^1$. $\mathbf{TV}^0$ is the base of the hierarchy of theories $\mathbf{TV}^0 \subseteq \mathbf{TV}^1 \subseteq \ldots$, 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. Other results on the hierarchies $\mathbf{V}^i$ and $\mathbf{TV}^i$ will be presented. This chapter also proves the RSUV isomorphism theorem between $\mathbf{S}_2^i$ and $\mathbf{V}^i$.

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 **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 presents further results on theories for the quantified propositional calculus.

Two sources have been invaluable for writing this book. The first is Krajíček's monograph [**?**], which is an essential possession for anyone working in this field. The second source is Buss's chapters [**?**, **?**] 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

The Propositional Calculus

In this chapter and the next we present the logical foundations for theories of bounded arithmetic. 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.

Propositional formulas (called simply *formulas* in this chapter) are built from the logical constants $\bot$, $\top$ (for False, True), propositional variables (or atoms) $P_1, P_2, ...$, connectives $\neg, \vee, \wedge$, and parentheses (,). We use $P, Q, R, ...$ to stand for propositional variables, $A, B, C, ...$ to stand for formulas, and $\Phi, \Psi, ...$ 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 1.1. *A truth assignment τ* satisfies *A iff $A^\tau = T$; τ* satisfies *a set Φ of formulas iff τ satisfies A for all $A \in \Phi$. Φ is* satisfiable *iff some τ satisfies Φ; otherwise Φ is* unsatisfiable*. Similarly for A. $\Phi \models A$ (i.e., A is a* logical consequence *of Φ) iff τ satisfies A for every τ such that τ satisfies Φ. A formula*

A is valid *iff* $\models A$ *(i.e., $A^\tau = T$ for all τ). A valid propositional formula is called a* tautology*. We say that A and B are* equivalent *(written $A \Longleftrightarrow B$) 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 $(P \vee Q) \neq_{syn} (Q \vee P)$.

1.1 Gentzen's Propositional Proof System PK

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

$$A_1, ..., A_k \longrightarrow B_1, ..., B_\ell \tag{1.1}$$

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 (1.1) iff either τ falsifies some A_i or τ satisfies some B_i. Thus the sequent is equivalent to the formula

$$\neg A_1 \vee \neg A_2 \vee ... \vee \neg A_k \vee B_1 \vee B_2 \vee ... \vee B_\ell \tag{1.2}$$

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

The *logical axioms* are of the form

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

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

weakening rules

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

exchange rules

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

contraction rules

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

$\neg$ introduction rules

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

$\wedge$ introduction rules

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

$\vee$ introduction rules

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

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*.

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

As an example, we give a **PK** proof of one of 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.)

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

Exercise 1.4. *Give* **PK** *proofs for each of the following valid sequents:*
a) $\neg P \vee \neg Q \longrightarrow \neg(P \wedge Q)$
b) $\neg(P \vee Q) \longrightarrow \neg P \wedge \neg Q$
c) $\neg P \wedge \neg Q \longrightarrow \neg(P \vee Q)$

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

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

1.2 Soundness and Completeness of PK

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

Theorem 1.7 (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). □

Theorem 1.8 (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 1.9 (Inversion Principle). *For each* **PK** *rule except for weakenings, if the bottom sequent is valid, then all top sequents are valid.*

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

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

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

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

1.3 PK Proofs from Assumptions

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

Theorem 1.10 (Derivational Soundness and Completeness Theorem). *A 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 1.15.

Exercise 1.11. *Let A_S be the formula giving the meaning of a sequent S, as in (1.2). 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 1.16, Form 2 (compactness), we obtain an easy proof of derivational completeness. Suppose that the sequent $\Gamma \longrightarrow \Delta$ is a logical consequence of sequents $S_1, ..., S_k$. 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

$$A_{S_1}, ..., A_{S_k}, \Gamma \longrightarrow \Delta \tag{1.3}$$

is valid, and hence has a **PK** proof by Theorem 1.8. Finally from (1.3) 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 1.12 (Anchored Proof). *An instance of the cut rule in a* **PK**-Φ *proof π is* anchored *if the cut formula A (also) occurs as a formula (rather than a subformula) in some nonlogical axiom of π. A* **PK**-Φ *proof π is* anchored *if every instance of cut in π is anchored.*

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

The derivational completeness theorem can be strengthened as follows.

Theorem 1.13 (Anchored Completeness Theorem). *If a 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 1.8 they have **PK** proofs π_1 and π_2, respectively. We can use these proofs to get a proof of $\Gamma \longrightarrow \Delta$ from $A \longrightarrow B$ as shown below, where the double line indicates several rules have been applied.

$$\dfrac{\begin{matrix}\vdots\ \pi_1\\ \Gamma \longrightarrow \Delta, A\end{matrix} \qquad \dfrac{\dfrac{A \longrightarrow B}{\overline{A, \Gamma \longrightarrow \Delta, B}}\text{(weakenings,exchanges)} \qquad \begin{matrix}\vdots\ \pi_2\\ B, A, \Gamma \longrightarrow \Delta\end{matrix}}{A, \Gamma \longrightarrow \Delta}\,(\mathbf{cut})}{\Gamma \longrightarrow \Delta}\,(\mathbf{cut})$$

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:

$$\dfrac{\dfrac{\longrightarrow A_1}{\overline{A_2, ..., A_k, \Gamma \longrightarrow \Delta, A_1}}\text{(weakenings,exchanges)} \qquad \begin{matrix}\vdots\ \pi\\ A_1, A_2, ..., A_k, \Gamma \longrightarrow \Delta\end{matrix}}{A_2, ..., A_k, \Gamma \longrightarrow \Delta}\,(\mathbf{cut})$$

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

A nice property of anchored proofs is the following.

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

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

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

We conclude this chapter with a fundamental result which plays an important role in both the propositional and predicate calculus.

Theorem 1.16 (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 1.17. *Prove the equivalence of the three forms. (Note that Form 3 is the contrapositive of Form 1.)*

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

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

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

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

(a) Prove Form 3 using Zorn's Lemma as follows: Call a set Ψ of formulas *finitely satisfiable* if every finite subset of Ψ is satisfiable. Assume that Φ is finitely satisfiable. Let $\mathcal{C}$ be the class of all finitely satisfiable sets $\Psi \supseteq \Phi$ of propositional formulas using atoms in Φ. Order these sets Ψ by inclusion. Show that the union of any chain of sets in $\mathcal{C}$ is again in the class $\mathcal{C}$. Hence by Zorn's Lemma, $\mathcal{C}$ has a maximal element Ψ_0. Show that Ψ_0 has a unique satisfying assignment, and hence Φ is satisfiable.
(b) Show that Form 1 follows from Tychonoff's Theorem: The product of compact topological spaces is compact. The set of all truth assignments to the atom set $\mathcal{A}$ can be given the product topology, when viewed as the product for all atoms P in $\mathcal{A}$ of the two-point space $\{T, F\}$ of assignments to P, with the discrete topology. By Tychonoff's Theorem, this space of assignments is compact. Show that for each formula A, the set of assignments falsifying A is open. Thus Form 1 follows from the definition of compact: every open cover has a finite subcover.

1.4 Notes

Our treatment of **PK** in sections 1.1 and 1.2 is adapted from Section 1.2 of [**?**].

Chapter 2

The Predicate Calculus

In this chapter 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, described in Chapter 1, 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. We present major corollaries of the completeness theorem, and finally present a form of the Herbrand Theorem.

2.1 Syntax

A *first-order language* (or just *language*, or *vocabulary*) $\mathcal{L}$ is specified by the following:

1) For each $n \in \mathbb{N}$ 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 2.1 ($\mathcal{L}$-Terms). *Let $\mathcal{L}$ be a first-order vocabulary:*

1) Every variable is an $\mathcal{L}$-term.
2) If f is an n-ary function symbol of $\mathcal{L}$ and $t_1, \ldots, t_n$ are $\mathcal{L}$-terms, then $ft_1 \ldots t_n$ 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 2.2 ($\mathcal{L}$-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 2.1. A similar remark applies to formulas, and Definition 2.2.

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

Definition 2.3 (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 2.4 (Free and Bound Variables). *An occurrence of x in A is* bound *iff it is in a subformula of A of the form $\forall x B$ 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 2.5. *A formula A or a term t is* closed *if it contains no free occurrence of a variable. A closed formula is called a* sentence*.*

2.2 Semantics of Predicate Calculus

Definition 2.6 ($\mathcal{L}$-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 2.7 (Object Assignment). *An* object assignment *σ for a structure $\mathcal{M}$ is a mapping from variables to the universe M.*

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

Definition 2.8 (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) *$(ft_1 \cdots t_n)^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma])$*

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

a) *$\mathcal{M} \models \top$ and $\mathcal{M} \not\models \bot$*

b) $\mathcal{M} \models (Pt_1 \cdots t_n)[\sigma]$ *iff* $\langle t_1^{\mathcal{M}}[\sigma], \ldots, t_n^{\mathcal{M}}[\sigma] \rangle \in P^{\mathcal{M}}$

c) If $\mathcal{L}$ *contains* $=$*, then* $\mathcal{M} \models (s = t)[\sigma]$ *iff* $s^{\mathcal{M}}[\sigma] = t^{\mathcal{M}}[\sigma]$

d) $\mathcal{M} \models \neg A[\sigma]$ *iff* $\mathcal{M} \not\models A[\sigma]$.

e) $\mathcal{M} \models (A \vee B)[\sigma]$ *iff* $\mathcal{M} \models A[\sigma]$ *or* $\mathcal{M} \models B[\sigma]$.

f) $\mathcal{M} \models (A \wedge B)[\sigma]$ *iff* $\mathcal{M} \models A[\sigma]$ *and* $\mathcal{M} \models B[\sigma]$.

g) $\mathcal{M} \models (\forall x A)[\sigma]$ *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 2.10 (Standard Model). *The* standard model $\underline{\mathbb{N}}$ *for the language* $\mathcal{L}_A$ *is a structure with universe* $M = \mathbb{N} = \{0, 1, 2, \ldots\}$*, where* $0, 1, +, \cdot, =, \leq$ *get their usual meanings on the natural numbers.*

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

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

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

Definition 2.11. *a)* $\mathcal{M} \models \Phi[\sigma]$ *iff* $\mathcal{M} \models A[\sigma]$ *for all* $A \in \Phi$.

b) $\mathcal{M} \models \Phi$ *iff* $\mathcal{M} \models \Phi[\sigma]$ *for all* σ.

c) $\Phi \models A$ *iff for all* $\mathcal{M}$ *and all* σ*, if* $\mathcal{M} \models \Phi[\sigma]$ *then* $\mathcal{M} \models A[\sigma]$.

d) $\models A$ *(*A *is* valid*) iff* $\mathcal{M} \models A[\sigma]$ *for all* $\mathcal{M}$ *and* σ.

e) $A \Longleftrightarrow B$ *(*A *and* B *are* logically equivalent*, or just* equivalent*) iff for all* $\mathcal{M}$ *and all* σ*,* $\mathcal{M} \models A[\sigma]$ *iff* $\mathcal{M} \models B[\sigma]$.

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

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

Definition 2.12 (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 2.13. *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 the length of s. □

Definition 2.14. *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 2.15 (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 2.16 (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. □

2.3 The First-Order Proof System LK

We now extend the propositional proof system **PK** to the first-order sequent proof system **LK**. For this it is convenient to introduce two kinds of variables: *free variables* denoted by $a, b, c, \ldots$ and *bound variables* denoted by $x, y, z, \ldots$. A first-order sequent has the form $A_1, \ldots, A_k \longrightarrow B_1, \ldots, B_\ell$, where now the A_i'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 $\forall xA(x)$ or $\exists xA(x)$ satisfy the free/bound variable restrictions described above.

$\forall$ introduction rules

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

$\exists$ introduction rules

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

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 $x, y, z, \ldots$.

Definition 2.17 (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*

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

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

Theorem 2.18 (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. □

Exercise 2.19. *Give examples to show that the restriction given on the quantifier rules, that b must not occur in the conclusion in $\forall$-**right** and $\exists$-**left**, is necessary to ensure that these rules preserve validity.*

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

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

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

Exercise 2.20. *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) $\forall x \exists y Pxy \longrightarrow \exists y \forall x Pxy$

2.3.1 Free Variable Normal Form

In future chapters it will be useful to assume that **LK** proofs satisfy certain restrictions on free variables.

Definition 2.21 (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* $\forall$-**right** *and* $\exists$-**left**, *and in these cases the eigenvariable which is eliminated is not a parameter variable, and (2) every nonparameter free variable appearing in π is used exactly once as an eigenvariable.*

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

Transform π by repeatedly performing the following operation until the resulting proof is in free variable normal form. Select some upper-most rule in π which eliminates a free variable from a sequent. If the rule is $\forall$-**right** or $\exists$-**left**, and the eignevariable 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**,

∃-right, or **∀-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).

2.3.2 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 2.22. *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 **∀-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 2.23. *Suppose that A is a formula whose free variables comprise the list $a_1, \ldots, a_n$. Then the* universal closure *of A, written $\forall A$, is the sentence $\forall x_1 \ldots \forall x_n A(x_1/a_1, \ldots, x_n/a_n)$, where $x_1, \ldots, x_n$ is a list of new (bound) variables. If Φ is a set of formulas, then $\forall\Phi$ is the set of all sentences $\forall A$, for A in Φ.*

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

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

Theorem 2.24 (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 2.17). Note that if $S =_{\text{syn}} \Gamma \longrightarrow \Delta$, then $\forall S$ is *not* equivalent to $\forall\Gamma \longrightarrow \forall\Delta$ in general.

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

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

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

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

Lemma 2.25 (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 $\{C_1, \ldots, C_n\}$ of Φ such that the sequent*

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

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

Note that a form of the Compactness Theorem for predicate calculus sentences without equality follows from the above lemma. See Theorem 2.43 for a more general form of compactness.

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

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

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

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

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

Proof of the Completeness Lemma. We loosely follow the proof of the Cut-free Completeness Theorem, pp 33-36 of Buss [**?**]. (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 1.8.) 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

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

Here the double line represents a derivation involving the rule $\vee$-**left**, together with **exchange**s to move the principle formulas to the left end of the antecedent and back. 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

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

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

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

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

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

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

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

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

Definition 2.27 (Construction of the "Term Model" $\mathcal{M}$). *The universe M of $\mathcal{M}$ is the set of all $\mathcal{L}$-terms t (which contain only "free" variables $a, b, c, \ldots$). The object assignment σ just maps every variable a to itself.*

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

Exercise 2.28. *Prove by structural induction that for every term t, $t^{\mathcal{M}}[\sigma] = t$.*

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

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

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

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

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

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

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

Theorem 2.29 (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 **cut**s can be restricted to the formulas C.

Exercise 2.30. *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 20 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 21 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 22 (Definition 2.27), satisfy $\forall\Phi$. This should follow from the Claim on page 22, 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).*

2.4 Equality Axioms

Definition 2.31. *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 2.32 (Equality Axioms of $\mathcal{L}$ ($\mathcal{E}_{\mathcal{L}}$)).

EA1 $\forall x(x = x)$ *(reflexivity)*
EA2 $\forall x\forall y(x = y \supset y = x)$ *(symmetry)*
EA3 $\forall x\forall y\forall z((x = y \wedge y = z) \supset x = z)$ *(transitivity)*
EA4 $\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_n (x_1 = y_1 \wedge \ldots \wedge x_n = y_n) \supset f x_1 \ldots x_n = f y_1 \ldots y_n$ *for each $n \geq 1$ and each n-ary function symbol f in $\mathcal{L}$.*

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

Axioms **EA1, EA2, EA3** assert that $=$ is an equivalence relation. Axiom **EA4** asserts that functions respect the equivalence classes, and Axiom **EA5** 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 2.33 (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 2.34. *$\Phi \models A$ 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 2.35. *$\forall \Phi \models A$ iff A has an* **LK**-*Ψ proof, where $\Psi = \Phi \cup \mathcal{E}_{\mathcal{L}}$.*

Corollary 2.34 follows immediately from the Equality Theorem and the fact that $\Phi \models A$ iff $\Phi \cup \{\neg A\}$ is unsatisfiable. Corollary 2.35 follows from Corollary 2.34 and the derivational soundness and completeness of **LK** (page 18), 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 **EA4** and **EA5** 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 $\hat{\mathcal{M}} \models A[\hat{\sigma}]$. □

2.4.1 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**, $\ldots$, **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 $x, y, z, \ldots$).

Definition 2.36 (Equality Axioms for LK).

E1 $\longrightarrow t = t$

E2 $t = u \longrightarrow u = t$

E3 $t = u, u = v \longrightarrow t = v$

E4 $t_1 = u_1, \ldots, t_n = u_n \longrightarrow ft_1 \ldots t_n = fu_1 \ldots u_n$, *for each* f *in* $\mathcal{L}$

E5 $t_1 = u_1, \ldots, t_n = u_n, Pt_1 \ldots t_n \longrightarrow Pu_1 \ldots u_n$, *for each* P *in* $\mathcal{L}$ *(Here* P *is not* $=$*)*

Note that the universal closures of **E1,...,E5** are semantically equivalent to **EA1,...,EA5**, and in fact using the **LK** rule $\forall$-**right** repeatedly, $\longrightarrow$ **EA**i is easily derived in **LK** from **E**i (with terms t, u, etc., taken to be distinct variables), $i = 1, \ldots, 5$. Thus Corollary 2.35 above still holds when $\Psi = \Phi \cup \{\mathbf{E1}, \ldots, \mathbf{E5}\}$.

Definition 2.37 (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 18, where* Ψ *is* Φ *together with all instances of the equality axioms* $\mathbf{E1}, \ldots, \mathbf{E5}$*. If* Φ *is empty, we simply refer to an* **LK**-*proof (but allow* $\mathbf{E1}, \ldots, \mathbf{E5}$ *as axioms).*

2.4.2 Revised Soundness and Completeness of LK

Theorem 2.38 (Revised Soundness and Completeness of LK). *For any set* Φ *of formulas and sequent* S*,*

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

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 $\mathbf{E1}, \ldots, \mathbf{E5}$ have no universal quantifiers, but instead have instances for all terms $t, u, \ldots$. Recall that in an anchored **LK** proof, **cut**s are restricted so that cut formulas must occur in the nonlogical axioms. In the presence of equality, the nonlogical axioms must include $\mathbf{E1}, \ldots, \mathbf{E5}$, but the only formulas occurring here are equations of the form $t = u$. Since the Anchored **LK** Completeness Theorem (page 23) still holds when Φ is a set of sequents rather than a set of formulas, and since $\mathbf{E1}, \ldots, \mathbf{E5}$ are closed under substitution of terms for variables, we can extend this theorem so that it works in the presence of equality.

Definition 2.39 (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* $\mathbf{E1}, \ldots, \mathbf{E5}$*).*

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

Theorem 2.40 (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 23) and the above discussion about axioms **E1**, ..., **E5**.

We are interested in anchored proofs because of their subformula property. The following result generalizes Proposition 1.15.

Theorem 2.41 (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* **E1**, ..., **E5***).*

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 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. $\square$

2.5 Major Corollaries of Completeness

Theorem 2.42 (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 2.25), 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 22). 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). $\square$

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 2.43 (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 1.16.)

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 2.35. $\square$

Theorem 2.44. *Suppose $\mathcal{L}$ has only finitely many 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 2.45 (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.)*

2.6 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 2.46. *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 2.47 ($\mathcal{L}$-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 2.48. *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. □

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 2.49 (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 2.50 (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

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

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 2.51. *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 2.52. *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 Herbrand Theorem, Form 1. If Φ_0 is propositionally unsatisfiable, then Φ is unsatisfiable. This follows easily from Lemma 2.48, since Φ_0 is a logical consequence of Φ. □

Proof of the Completeness direction of Herbrand Theorem, Form 1. This follows from the Anchored **LK** Completeness Theorem (see Exercise 2.54 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 2.34, 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 1.16, 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 2.27).

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 2.8) and the Substitution Theorem (Theorem 2.15). □

Exercise 2.53. *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 2.54. *Show that the completeness direction of the Herbrand Theorem (Form 1) follows from the Anchored* **LK** *Completeness Theorem (with equality, Definition 2.39 and Theorem 2.40) and the following syntactic lemma.*

Lemma 2.55. *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 2.56 (Prenex Form). *We say that a formula A is in* prenex form *if A has the form $\mathsf{Q}_1 x_1 \dots \mathsf{Q}_n x_n B$, where each Q_i is either $\forall$ or $\exists$, and B is a quantifier-free formula.*

Theorem 2.57 (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 15). 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 2.16), 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.

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

□

2.7 Notes

Sections 2.1 to 2.3 roughly follow Sections 2.1 and 2.2 of [**?**]. However an important difference is that the definition of $\Phi \models A$ in [**?**] 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 2.30 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 $\mathbf{S}^1_2 \subseteq \mathbf{T}^1_2 \subseteq \mathbf{S}^2_2 \ldots$. 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 $\mathbb{N} = \{0, 1, 2, ...\}$.

3.1 Peano Arithmetic

Definition 3.1. *A* theory *over a language $\mathcal{L}$ 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 2.23).

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 language (Definition 2.3). Recall that the *standard structure* $\underline{\mathbb{N}}$ 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 structure.

Definition 3.2. **TA** *(True Arithmetic) is the theory consisting of all formulas valid in the standard structure:*

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

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

Note that by Definition 2.6, $=$ 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 2.37 and Corollaries 2.34, 2.35).

We start by listing nine "basic" quantifier-free formulas **B1**, ..., **B8** and **C**, which comprise the axioms for our basic theory. See Figure 3.1 below.

B1. $x + 1 \neq 0$	**B5.** $x \cdot 0 = 0$
B2. $x + 1 = y + 1 \supset x = y$	**B6.** $x \cdot (y + 1) = (x \cdot y) + x$
B3. $x + 0 = x$	**B7.** $(x \leq y \wedge y \leq x) \supset x = y$
B4. $x + (y + 1) = (x + y) + 1$	**B8.** $x \leq x + y$
C. $0 + 1 = 1$	

Figure 3.1: 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, **O2**). However we put it in so that $\forall\mathbf{B1}, \ldots, \forall\mathbf{B8}, \forall\mathbf{C}$ 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 **B1** and **C**, and the induction step follows from **B2** and **C**.

Next we use **B3**, **B4** 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 **B5**, **B6** 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 **B8**, 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 **B7**, 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. □

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

$$[\varphi(0) \wedge \forall x,\ \varphi(x) \supset \varphi(x+1)] \supset \forall z \varphi(z) \tag{3.1}$$

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* **B1**, ..., **B8**, *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, including apparently Andrew Wiles' proof of Fermat's Last Theorem. 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, I$\mathbf{\Delta}_0$, I$\mathbf{\Sigma}_1$). **OPEN** *is the set of open (i.e., quantifier-free) formulas;* $\mathbf{\Delta}_0$ *is the set of bounded formulas; and* $\mathbf{\Sigma}_1$ *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**, **I**$\mathbf{\Delta}_0$*, and* **I**$\mathbf{\Sigma}_1$ *are the subsystems of* **PA** *obtained by restricting the induction scheme so that* Φ *is* **OPEN**, $\mathbf{\Delta}_0$*, and* $\mathbf{\Sigma}_1$*, 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 ·)*
O5. $x\cdot y = y\cdot x$ *(Commutativity of ·)*
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. **O1:** 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 **O3**
O5: induction on y, after establishing $(y+1)\cdot x = y\cdot x + x$ by induction on x
O6: induction on z
O7: B8, **O2**, **B3**
O8: O7, **B7**
O9: B8, **B3**
O10: induction on x and **B2**. □

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* **I**$\mathbf{\Delta}_0$*:*
D1. $x \ne 0 \supset \exists y \le x(x = y+1)$ *(Predecessor)*
D2. $\exists z(x+z = y \vee y+z = x)$
D3. $x \le y \leftrightarrow \exists z(x+z = y)$
D4. $(x \le y \wedge y \le z) \supset x \le z$ *(Transitivity)*
D5. $x \le y \vee y \le x$ *(Total order)*
D6. $x \le y \leftrightarrow x+z \le y+z$
D7. $x \le y \supset x\cdot z \le y\cdot z$
D8. $x \le y+1 \leftrightarrow (x \le y \vee x = y+1)$ *(Discreteness 1)*
D9. $x < y \leftrightarrow x+1 \le y$ *(Discreteness 2)*
D10. $x\cdot z = y\cdot z \wedge z \ne 0 \supset x = y$ *(Cancellation law for ·)*

Proof. **D1:** Induction on x
D2: Induction on x. Base case: **B2**, **O2**. Induction step: **B3**, **B4**, **D1**.
D3: $\Longrightarrow$: **D2**, **B3** and **B7**; $\Longleftarrow$: **B8**.
D4: **D3**, **O1**.
D5: **D2**, **B8**.
D6: $\Longrightarrow$: **D3**, **O1**, **O2**; $\Longleftarrow$: **D3**, **O6**.
D7: **D3** and algebra (**O1**,...,**O8**).
D8: $\Longrightarrow$: **D3**, **D1**, and algebra; $\Longleftarrow$: **O9**, **B8**, **D4**.
D9: $\Longrightarrow$: **D3**, **D1**, and algebra; $\Longleftarrow$: **D3** and algebra.
D10: Exercise. □

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 $\mathbf{I\Delta}_0 \nvdash \forall x \varphi(x)$. 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 structure.

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 $\forall x \leq t \psi(x)$ and φ is $\exists x \leq t \psi(x)$, 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 $\mathbf{I\Delta}_0 \vdash \varphi$. □

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

$$\mathbf{I\Delta}_0 \vdash x \leq \underline{n} \leftrightarrow (x = \underline{0} \vee x = \underline{1} \vee ... \vee x = \underline{n})$$

Proof. Induction on n. The base case $n = 0$ follows from **O7** and **O8**, and the induction step follows from **D8**. □

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*

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

where φ *is a formula in* Φ*.*

Theorem 3.14. $\mathbf{I\Delta}_0$ *proves* $\mathbf{\Delta}_0$-**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* **B1**, …, **B8**, **O10** *(Example 3.8),* **D1** *(Example 3.9), and the axiom scheme* $\mathbf{\Delta}_0$-**MIN**.

Bounded Induction Scheme

The $\mathbf{\Delta}_0$-**IND** scheme of $\mathbf{I\Delta}_0$ can be replaced by the following *bounded induction scheme* for $\mathbf{\Delta}_0$ formulas, i.e.,

$$[\varphi(0) \wedge \forall x < z(\varphi(x) \supset \varphi(x+1))] \supset \varphi(z) \tag{3.2}$$

where $\varphi(x)$ is any $\mathbf{\Delta}_0$-formula. (Note that the **IND** formula (3.1) 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* $\mathbf{\Delta}_0$-**IND** *scheme is replaced by the above bounded induction scheme for* $\mathbf{\Delta}_0$ *formulas. (It suffices to show that the new scheme is provable in* $\mathbf{I\Delta}_0$*.)*

Strong Induction Scheme

The *strong induction axiom* for a formula $\varphi(x)$ is the following formula:

$$\forall x[(\forall y < x\varphi(y)) \supset \varphi(x)] \supset \forall z\varphi(z) \tag{3.3}$$

Exercise 3.17. *Show that* $\mathbf{I\Delta}_0$ *proves the strong induction axiom scheme for* $\mathbf{\Delta}_0$ *formulas.*

3.2 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

$$\mathcal{T} \vdash \forall \vec{x}\ f(\vec{x}) \leq t(\vec{x}) \tag{3.4}$$

We say that f is *polynomially 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 *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 3.3.3 we will show how to represent the relation $y = 2^x$ by a bounded formula φ_{exp}. 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 **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 **PA**.

Our proof of Parikh's Theorem will be based on the Anchored **LK** Completeness Theorem with Equality (2.40). 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 **B1**–**B8** (page 34) and the $\mathbf{\Delta}_0$-**IND** scheme, which can be replaced by the Bounded Induction Scheme (3.2).

Definition 3.23 ($\mathbf{ID}_0$). $\mathbf{ID}_0$ *is the set of all term substitution instances of* **B1**–**B8** *and the Bounded Induction Scheme, where now the terms contain only "free" variables* $a, b, c, \dots$.

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

For example $(c \cdot b) + 1 \neq 0$ is an instance of **B1**, 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 (3.2) 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 (2.40) and Derivational Soundness of **LK** (2.24).

Theorem 3.24 (LK-$\mathbf{ID}_0$ Adequacy Theorem). *Let A be an $\mathcal{L}_A$-formula satisfying the* **LK** *constraint that only variables $a, b, c, \dots$ occur free and only $x, y, z, \dots$ 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 2.3.1). By the sub-formula property of anchored proofs (2.41), 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 **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 $\exists y \leq t\varphi(y)$ 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 $\exists y \leq t\varphi$, where the term t depends on $\mathcal{S}$ and the placement of $\mathcal{S}$ in π. Further t satisfies the condition

$$\text{Every variable in } t \text{ occurs free in the original sequent } \mathcal{S}. \tag{3.5}$$

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

In order to maintain the condition (3.5) we use our assumption that π is in free variable normal form. Thus if the variable b occurs in t in the formula $\exists y \leq t\varphi$, 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

$$\frac{\Gamma \longrightarrow \Delta, \varphi(s)}{\Gamma \longrightarrow \Delta, \exists y\varphi(y)} \tag{3.6}$$

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

$$\Gamma \longrightarrow \Delta, \exists y \leq s\varphi(y) \tag{3.7}$$

from the top sequent. Let $\mathbf{D}(\mathcal{S})$ be that derivation and let $\mathcal{S}'$ be the sequent (3.7).

If Δ has one or more occurrence of $\exists y\varphi$, then by the induction hypothesis the top sequent $\mathcal{S}_1$ of (3.6) 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

$$\Gamma \longrightarrow \Delta', \exists y \leq t\varphi, \exists y \leq s\varphi \tag{3.8}$$

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 (3.8) with two **cut**s and a **contraction** to obtain a derivation of

$$\Gamma \longrightarrow \Delta', \exists y \leq (s+t)\varphi(y) \tag{3.9}$$

Let $\mathbf{D}(\mathcal{S})$ be this derivation and let $\mathcal{S}'$ be the resulting sequent (3.9).

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

$$\frac{\Gamma \longrightarrow \Delta}{\Gamma \longrightarrow \Delta, \exists y\varphi} \tag{3.10}$$

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 (3.10).

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

$$\frac{b \leq r \wedge \psi(b), \Gamma \longrightarrow \Delta}{\exists x \leq r\psi(x), \Gamma \longrightarrow \Delta} \tag{3.11}$$

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 (3.11). Otherwise, by the induction hypothesis, the top sequent was converted to a sequent of the form

$$b \leq r \wedge \psi(b), \Gamma \longrightarrow \Delta', \exists y \leq s(b)\varphi(y) \tag{3.12}$$

Note that b may appear on the succedent and thus violate the Restriction of the $\exists$**-left** rule (page 16).

In order to apply the $\exists$**-left** rule (and continue to satisfy the condition (3.5)), 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 (3.12) 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

$$\exists x \leq r\psi(x), \Gamma \longrightarrow \Delta', \exists y \leq t\varphi(y) \tag{3.13}$$

Let $\mathbf{D}(\mathcal{S})$ be this derivation and let $\mathcal{S}'$ be the resulting sequent (3.13).

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 $\mathcal{S}' = \mathcal{S}_1'$. □

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

3.3 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 $\mathcal{L}_1$.*

3.3.1 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 $\varphi(x)$.

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*

$$P(\vec{x}) \leftrightarrow \varphi(\vec{x}) \tag{3.14}$$

(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*

$$\mathcal{T} \vdash \forall\vec{x}\exists! y\varphi(\vec{x}, y), \tag{3.15}$$

and that

$$y = f(\vec{x}) \leftrightarrow \varphi(\vec{x}, y) \tag{3.16}$$

We say that (3.14) *is a* defining axiom *for $P(\vec{x})$ and* (3.16) *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 $\mathit{Prime}(x)$ (in any theory whose vocabulary contains $\mathcal{L}_A$) might be

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

Note that $\mathbf{\Delta}_0$ and $\mathbf{\Sigma}_1$ (Definition 3.7) are sets of $\mathcal{L}_A$-formulas. In general, given a language $\mathcal{L}$ the sets $\mathbf{\Delta}_0(\mathcal{L})$ and $\mathbf{\Sigma}_1(\mathcal{L})$ are defined as in Definition 3.7 but the formulas are from $\mathcal{L}$.

Notation In Definition 3.27, if $\Phi = \mathbf{\Delta}_0(\mathcal{L})$ (resp. $\Phi = \mathbf{\Sigma}_1(\mathcal{L})$) then we sometimes omit mention of $\mathcal{L}$ and simply say that the symbols P, f are $\mathbf{\Delta}_0$-definable (resp. $\mathbf{\Sigma}_1$-definable) in $\mathcal{T}$.

In the case of functions, the choice $\Phi = \mathbf{\Sigma}_1(\mathcal{L})$ plays a special role. A $\mathbf{\Sigma}_1$-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 $\mathbf{S}^1_2$ (see Section 3.5) the polytime functions. In Section 3.4 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 $f(x_1, \ldots, x_m)$ and $h_i(x_1, \ldots, x_n)$ (for $1 \le i \le m$) are $\mathbf{\Sigma}_1$-definable in a theory $\mathcal{T}$. Show that the function $f(h_1(\vec{x}), \ldots, h_m(\vec{x}))$ (where $\vec{x}$ stands for $x_1, \ldots, x_n$) is also $\mathbf{\Sigma}_1$-definable in $\mathcal{T}$. (In other words, show that $\mathbf{\Sigma}_1$-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 (3.14) or (3.16) is satisfied. Notice that this interpretation is uniquely determined by the defining axiom, and in the case of a function symbol the provability condition (3.15) is needed (both existence and uniqueness of y) in order to ensure that both directions of the equivalence (3.16) 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 $\mathcal{T}_1 \vdash A$. □

Corollary 3.31. *Let $\mathcal{T}$ be a theory and $\mathcal{T}_0 = \mathcal{T} \subset \mathcal{T}_1 \subset \ldots$ 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 $\mathcal{T}_\infty = \bigcup_{n\geq 0} \mathcal{T}_n$. 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 $\mathbf{\Delta}_0$-definable symbols, and in fact we may add both $\mathbf{\Delta}_0$-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 $\mathbf{\Sigma}_1$-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 $\forall \vec{x} \exists ! y \leq t \varphi(\vec{x}, y)$.

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

Corollary 3.34. *If $\mathcal{T}$ is a polynomial-bounded theory, then a function f is $\mathbf{\Sigma}_1$-definable in $\mathcal{T}$ iff f is $\mathbf{\Delta}_0$-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 3.3.3 that the *relation* $(y = 2^x)$ is $\mathbf{\Delta}_0$-definable in $\mathbf{I\Delta}_0$. Since the function 2^x is Σ_1-definable in $\mathbf{PA}$, 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 $\mathbf{\Delta}_0$-definable predicate or a $\mathbf{\Sigma}_1$-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 $\mathbf{\Delta}_0(\mathcal{L})$-**IND** axiom scheme. Then $\mathcal{T}^+$ proves the $\mathbf{\Delta}_0(\mathcal{L}^+)$-**IND** axiom scheme.*

Proof of the Conservative Extension Lemma. First, suppose that $\mathcal{T}^+$ is obtained from $\mathcal{T}$ by adding to it a $\mathbf{\Delta}_0$-definable predicate symbol P and its defining axiom (3.14). 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, 2.16). Note that the defining axiom (3.14) 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 $\mathbf{\Sigma}_1$-definable function symbol f and its defining axiom (3.16). That $\mathcal{T}^+$ is polynomial bounded follows from Theorem 3.33.

Start translating φ^+ by replacing every bounded quantifier $\forall x \leq u\psi$ by $\forall x \leq u'(x \leq u \supset \psi)$, 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 $\exists x \leq u\psi$ by $\exists x \leq u'(x \leq u \wedge \psi)$.

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

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

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 $\mathbf{\Delta}_0(\mathcal{L}_0)$-**IND** axioms. Let $\mathcal{T}_0 \subset \mathcal{T}_1 \subset \mathcal{T}_2 \subset ...$ be a sequence of extensions of $\mathcal{T}_0$ where each $\mathcal{T}_{i+1}$ is obtained from $\mathcal{T}_i$ by adding a $\mathbf{\Sigma}_1$-definable function symbol f_{i+1} (or a $\mathbf{\Delta}_0$-definable predicate symbol P_{i+1}) and its defining axiom. Let*

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

*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 $\mathbf{\Delta}_0(\mathcal{L})$ formula with some $\mathbf{\Delta}_0(\mathcal{L}_0)$ formula, and $\mathcal{T} \vdash \mathbf{\Delta}_0(\mathcal{L})$-**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 $\mathbf{\Delta}_0(\mathcal{L}_i)$ formula is equivalent to some $\mathbf{\Delta}_0(\mathcal{L}_0)$ 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 $\mathbf{\Delta}_0(\mathcal{L})$ formula with some $\mathbf{\Delta}_0(\mathcal{L}_0)$ formula, and $\mathcal{T} \vdash \mathbf{\Delta}_0(\mathcal{L})$-**IND**. It follows from Corollary 3.31 that $\mathcal{T}$ is a conservative extension of $\mathcal{T}_0$. □

3.3.2 $\overline{\mathbf{I\Delta}}_0$: A Universal Conservative Extension of $\mathbf{I\Delta}_0$

Note 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 2.6).

A *universal formula* is a formula in prenex form (Definition 2.56) 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 2.46). We can now restate Form 2 of the Herbrand Theorem (2.50) 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*

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_m \exists y \varphi(\vec{x}, y). \tag{3.17}$$

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

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

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 (3.17), 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 (3.17) it follows that

$$\Gamma \cup \{\exists x_1 \ldots \exists x_m \forall y \neg \varphi(\vec{x}, y)\} \tag{3.18}$$

is unsatisfiable. Let $c_1, \ldots, c_m$ be new constant symbols. Then it is easy to check that (3.18) 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 2.49), 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

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

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 2.42) 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 $\mathbf{\Delta}_0$-**IND** axiom scheme. Let $\varphi(z)$ be a $\mathbf{\Delta}_0$ formula (possibly with other free variables $\vec{x}$). Then the induction scheme for $\varphi(z)$ can be written as

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

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 a 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 φ (thus $\vec{x}$ may contain z if t does). Let $f_{\varphi,t}(\vec{x})$ be the least $y < t$ such that $\varphi(y)$ holds, or t if no such y exists. Then $f_{\varphi,t}$ is total and can be defined as follows (we assume that y, v do not appear in $\vec{x}$):

$$y = f_{\varphi,t}(\vec{x}) \leftrightarrow [y \leq t \ \wedge\ (y < t \supset \varphi(y)) \ \wedge\ \forall v < y \neg\varphi(v)] \tag{3.19}$$

Note that (3.19) contains an implicit existential quantifier $\exists v$ (consider the direction $\leftarrow$). Our universal theory will contain the following equivalent axiom instead:

$$f(\vec{x}) \leq t \ \wedge\ [f(\vec{x}) < t \supset \varphi(f(\vec{x}))] \ \wedge\ [v < f(\vec{x}) \supset \neg\varphi(v)] \tag{3.20}$$

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

Although the predecessor function $pd(x)$ can be defined by a formula of the form (3.20), we will use the following two recursive defining axioms instead.

D1′. $pd(0) = 0$ **D1″**. $x \neq 0 \supset pd(x) + 1 = x$

Note that **D1″** implies **D1** (see Example 3.9), and **D1′** is needed to define $pd(0)$.

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

Definition 3.41 ($\mathcal{L}_{\mathbf{\Delta}_0}$). *Let $\mathcal{L}_{\mathbf{\Delta}_0}$ be the smallest set that satisfies*

1) $\mathcal{L}_{\mathbf{\Delta}_0}$ includes $\mathcal{L}_A \cup \{pd\}$;

2) For each open $\mathcal{L}_{\mathbf{\Delta}_0}$-formula $\varphi(\vec{x}, z)$ and $\mathcal{L}_A$-term $t(\vec{x})$ there is a function $f_{\varphi,t}$ in $\mathcal{L}_{\mathbf{\Delta}_0}$.

Note that $\mathcal{L}_{\mathbf{\Delta}_0}$ can be alternatively defined as follows. Let

$$\mathcal{L}_0 = \mathcal{L}_A \cup \{pd\}$$
$$\text{for } n \geq 0\text{: } \mathcal{L}_{n+1} = \mathcal{L}_n \cup \{f_{\varphi,t} : \varphi \text{ is an open } \mathcal{L}_n\text{-formula, } t \text{ is an } \mathcal{L}_A\text{-term}\}$$

Then

$$\mathcal{L}_{\mathbf{\Delta}_0} = \bigcup_{n\geq 0} \mathcal{L}_n$$

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 $\mathcal{L}_{\mathbf{\Delta}_0}$ with the following set of axioms: $\mathbf{B1}, \ldots, \mathbf{B8}, \mathbf{B8'}, \mathbf{B8''}, \mathbf{D1'}, \mathbf{D1''}$ and (3.20) for each function $f_{\varphi,t}$ of $\mathcal{L}_{\mathbf{\Delta}_0}$.*

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

Lemma 3.44. *For each $\mathbf{\Delta}_0(\mathcal{L}_A)$ formula φ, there is an open $\mathcal{L}_{\mathbf{\Delta}_0}$-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 $\exists y \leq t\psi(y)$. 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 (3.20). No properties of $\leq$ and $<$ are needed for this implication except the definition $y < f(\vec{x})$ stands for $(y \leq f(\vec{x}) \wedge y \neq f(\vec{x}))$. □

Proof of Theorem 3.43. First we show that $\overline{\mathbf{I\Delta}}_0$ is an extension of $\mathbf{I\Delta}_0$, i.e., $\mathbf{\Delta}_0$-**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 $\mathcal{L}_{\mathbf{\Delta}_0}$-formulas. Let $\varphi(\vec{x}, z)$ be any open $\mathcal{L}_{\mathbf{\Delta}_0}$-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 (3.20). We need to be careful when arguing about $\leq$, because the properties **O1**–**O9** and **D1**–**D10** which we have been using for reasoning in $\mathbf{I\Delta}_0$ require induction to prove.

First we rewrite (3.20) for the case f is $f_{\neg\varphi,z}$.

$$f(\vec{x}, z) \leq z \ \wedge\ [f(\vec{x}, z) < z \supset \neg\varphi(f(\vec{x}, z))] \ \wedge\ [v < f(\vec{x}, z) \supset \varphi(v)] \qquad (3.21)$$

Now $0 < z$ by $\mathbf{B8}'$ and our assumptions $\varphi(0)$ and $\neg\varphi(z)$, so $f(\vec{x}, z) \neq 0$ by (3.21). Hence $y + 1 = pd(f(\vec{x}, z)) + 1 = f(\vec{x}, z)$ by $\mathbf{D1}''$. Therefore $\neg\varphi(y + 1)$ by (3.21) and the assumption $\neg\varphi(z)$.

To establish $\varphi(y)$ it suffices by (3.21) to show $y < f(\vec{x}, z)$. This holds because $f(\vec{x}, z) = y + 1$ as shown above, and $y < y + 1$ by $\mathbf{B8}''$.

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 $f_1 = pd, f_2, f_3, \ldots$ be an enumeration of $\mathcal{L}_{\mathbf{\Delta}_0} \setminus \mathcal{L}_A$ such that for $n \geq 1$, f_{n+1} is defined using some $\mathcal{L}_A$-term t and $(\mathcal{L}_A \cup \{f_1, \ldots, f_n\})$-formula φ as in (3.20).

For $n \geq 0$ let $\mathcal{L}_n$ denote $\mathcal{L}_A \cup \{f_1, \ldots, f_n\}$. Let $\mathcal{T}_0 = \mathbf{I\Delta}_0$, and for $n \geq 0$ 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 $\mathbf{D1}', \mathbf{D1}''$). Then

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

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 $\mathbf{\Delta}_0(\mathcal{L}_n)$-**IND** axiom scheme;
2) f_{n+1} is $\mathbf{\Delta}_0(\mathcal{L}_n)$-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 $\mathbf{\Delta}_0(\mathcal{L}_n)$-**IND** axiom scheme and $\mathbf{\Delta}_0(\mathcal{L}_n)$-defines f_{n+1}. Therefore by Corollary 3.36, $\mathcal{T}_{n+1}$ proves the $\mathbf{\Delta}_0(\mathcal{L}_{n+1})$-**IND** axiom scheme. Consequently, $\mathcal{T}_{n+1}$ also proves the $\mathbf{\Delta}_0(\mathcal{L}_{n+1})$-**MIN** axiom scheme. The defining equation for f_{n+2} has the form (3.20), and hence $\mathcal{T}_{n+1}$ proves (3.19) where f is f_{n+2}. Thus (3.19) is a defining axiom which shows that f_{n+2} is $\mathbf{\Delta}_0(\mathcal{L}_{n+1})$-definable in $\mathcal{T}_{n+1}$. Here we use the $\mathbf{\Delta}_0(\mathcal{L}_{n+1})$-**MIN** axiom scheme to prove $\exists y$ in (3.15). □

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 $\mathbf{\Sigma}_1$-definable function) to show that $\mathcal{T}_n$ proves the $\mathbf{\Delta}_0(\mathcal{L}_n)$-**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 $\overline{\mathbf{I\Delta}}_0$, f_{n+1} is already $\mathbf{\Delta}_0$-definable in $\mathcal{T}_n$ (the induction step in the proof of Theorem 3.43). Therefore we have actually used only a simple case of Corollary 3.36 (i.e., adding $\mathbf{\Delta}_0$-definable functions). 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

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

By Lemma 3.44, there is an open $\mathcal{L}_{\mathbf{\Delta}_0}$-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 $\mathcal{L}_{\mathbf{\Delta}_0}$-terms $t_1,\dots,t_n$ such that

$$\overline{\mathbf{I\Delta}}_0 \vdash \forall\vec{x}[\varphi'(\vec{x},t_1(\vec{x})) \vee \ldots \vee \varphi'(\vec{x},t_n(\vec{x}))] \tag{3.22}$$

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

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

Note that we have proved more than a bound on the existential quantifier $\exists y$. In fact, (3.22) 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.

3.3.3 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 $\mathbf{\Delta}_0$-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 3.4 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 $\mathbf{\Delta}_0$-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 $\mathbf{\Sigma}_1$-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 $\mathbf{\Delta}_0$-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 (3.25) defines $(y = 2^x)$, and the bounded formula $BIT(i, x)$ given in (3.26) 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 $\mathbf{\Delta}_0$-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 $\mathbf{\Delta}_0$-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 $\mathbf{O1}, \ldots, \mathbf{O9}$, $\mathbf{D1}, \ldots, \mathbf{D10}$ 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

$$z = x \dot{-} y \leftrightarrow [(y + z = x) \vee (x \le y \wedge z = 0)]$$

In order to show that $\mathbf{I\Delta}_0$ can $\mathbf{\Delta}_0$-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 $\mathbf{D2}$ 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 $\mathbf{B8}$, $x \le x + z'$, hence $x \le y$, and thus we can take $z = 0$.

For the uniqueness of z, first suppose that $x \le y$. Then we have to show that $y + z = x \supset z = 0$. Assume $y + z = x$. By $\mathbf{B8}$, $y \le y + z$, hence $y \le x$. Therefore $x = y$ by $\mathbf{B7}$. Now from $x + 0 = x$ ($\mathbf{B3}$) and $x + z = x$ we have $z = 0$, by $\mathbf{O2}$ (Commutativity of $+$) and $\mathbf{O6}$ (Cancellation law for $+$).

Next, suppose that $\neg(x \le y)$. Then $y + z = x$, and by $\mathbf{O2}$ and $\mathbf{O6}$, $y + z = x \wedge y + z' = x \supset z = z'$.

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

$$z = \lfloor x/y \rfloor \leftrightarrow [(y \cdot z \leq x \land x < y(z+1)) \lor (y = 0 \land z = 0)]$$

The existence of z is proved by induction on x. The uniqueness of z follows from transitivity of $\leq$ (**D4**), Total Order (**D5**), and **O5**, **D7**.

3) **Remainder:** The function x mod y can be defined by

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

Since x mod y is a composition of $\mathbf{\Sigma}_1$-definable functions, it is $\mathbf{\Sigma}_1$-definable by Exercise 3.28. Hence it is $\mathbf{\Delta}_0$-definable by Corollary 3.34.

4) **Square root:**

$$y = \lfloor \sqrt{x} \rfloor \leftrightarrow [y \cdot y \leq x \land x < (y+1)(y+1)]$$

The existence of y follows from the least number principle. The uniqueness of y follows from Transitivity of $\leq$ (**D4**), Total Order (**D5**), and **O5**, **D7**.

Exercise 3.45. *Show carefully that the functions x/y, x mod y and $\lfloor \sqrt{x} \rfloor$ are $\mathbf{\Delta}_0$-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

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

6) **Powers of 2 and 4:**

x is a power of 2 :	$Pow2(x) \leftrightarrow [x \neq 0 \land \forall y \leq x((1 < y \land y|x) \supset 2|y)]$
x is a power of 4 :	$Pow4(x) \leftrightarrow Pow2(x) \land x \bmod 3 = 1$

7) **LenBit:** We want the relation $LenBit(2^i, x)$ 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

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

Note that we intend to use $LenBit(y, x)$ 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.,

$$\begin{array}{rcl} (\forall 2^i)\ A(2^i) & \text{stands for} & \forall x\ (Pow2(x) \supset A(x)) \\ (\forall 2^i \leq t)\ A(2^i) & \text{stands for} & \forall x\ ((Pow2(x) \land x \leq t) \supset A(x)) \end{array}$$

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) $\forall x Pow2(x) \leftrightarrow Pow2(2x)$.
b) $(\forall 2^i)(\forall 2^j)(2^i < 2^j \supset 2^i | 2^j)$. *(Hint: using strong induction* (3.3).*)*
c) $(\forall 2^i)(\forall 2^j \leq 2^i)\ Pow2(2^i/2^j))$.
d) $(\forall 2^i)(\forall 2^j)(2^i < 2^j \supset 2 \cdot 2^i \leq 2^j)$.
e) $(\forall 2^i)(\forall 2^j)\, Pow2(2^i \cdot 2^j)$.
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**:**

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

Exercise 3.47. *Show that* $\mathbf{I\Delta}_0$ *can* $\mathbf{\Delta}_0$*-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) $x > 0 \supset LenBit(gp(x), x)$.
c) $y = x \dot{-} gp(x) \supset (\forall 2^i \leq y)\ (LenBit(2^i, y) \leftrightarrow LenBit(2^i, x))$.

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 (3.3) 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.*

Defining the Relation $y = 2^x$

This is much more difficult to $\mathbf{\Delta}_0$-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 $(x_{k-1} \ldots x_0)_2$ is the binary representation of x (where $x_{k-1} = 1$), i.e.,

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

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 $x_{k-1} = 1$):

$$\begin{aligned} &a_1 = 1, \qquad b_1 = 2 \\ &\text{For } 1 \le i < k\text{: } a_{i+1} = x_{k-i-1} + 2a_i \qquad b_{i+1} = 2^{x_{k-i-1}} b_i^2 \end{aligned} \tag{3.23}$$

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

We will code the sequences $\langle a_1, \ldots, a_k \rangle$ and $\langle b_1, \ldots, b_k \rangle$ 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

$$ext(u, z) = \lfloor z/u \rfloor \bmod u \tag{3.24}$$

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 $\mathbf{\Delta}_0$-definable in $\mathbf{I\Delta}_0$.

Note that $a, b < 2^{2^{k+1}}$, and $y \ge 2^{k-1}$. 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

$$\begin{aligned} &y = 2^x \leftrightarrow \varphi_{exp}(x, y) \\ \text{where} \qquad &\varphi_{exp} \equiv (x = 0 \wedge y = 1) \vee \exists a, b < y^4 \psi_{exp}(x, y, a, b) \end{aligned} \tag{3.25}$$

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, $2^{2^1} \le u \le y$ 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 $u \le y^2$ 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 $\mathbf{\Delta}_0$-defined by the formula

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

From this we can $\mathbf{\Delta}_0$-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) $\forall x \forall y,\ \varphi_{exp}(x, y) \supset Pow2(y)$.

b) $\mathbf{I\Delta}_0 \vdash Pow2(y) \supset \exists x < y\ \varphi_{exp}(x, y)$. *(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 $\mathbf{\Delta}_0$-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 $\mathbf{\Delta}_0$-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* (3.23).

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 $\mathbf{\Delta}_0$-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.*

The *BIT* Relation

Finally 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:

$$BIT(i, x) \leftrightarrow \exists z \leq x(z = 2^i \wedge LenBit(z, x)) \tag{3.26}$$

Exercise 3.55. *Show that the* **Length function**, *$|x| = \lceil \log_2(x + 1) \rceil$, is $\mathbf{\Delta}_0$-definable in $\mathbf{I\Delta}_0$.*

3.4 $\mathbf{I\Delta}_0$ and the Linear Time Hierarchy

3.4.1 The Polynomial and Linear Time Hierarchies

An element of a complexity class such as **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 **P** to be subsets of $\mathbb{N}$, or more generally sets of 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 $\mathbf{\Sigma}_0^p$ is sometimes used for polynomial time. Thus $\mathbf{\Sigma}_0^p$ 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 $\mathbf{\Sigma}_i^p$ is the i-th level of the polynomial-time hierarchy. This can be defined inductively by the recurrence

$$\mathbf{\Sigma}_{i+1}^p = \mathbf{NP}(\mathbf{\Sigma}_i^p)$$

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

Alternatively, $\mathbf{\Sigma}_i^p$ is 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,

$$\mathbf{\Sigma}_1^p = \mathbf{NP}$$

We define the polynomial time hierarchy by

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

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 **LinTime** and **NLinTime** to denote time $O(n)$ on a deterministic and nondeterministic multi-tape Turing machine, respectively. Then

$$\mathbf{\Sigma}_0^{lin} = \mathbf{LinTime}$$

and for $i \geq 0$

$$\mathbf{\Sigma}_{i+1}^{lin} = \mathbf{NLinTime}(\mathbf{\Sigma}_i^{lin}) \qquad (3.27)$$

Alternatively, we can define $\mathbf{\Sigma}_i^{lin}$ to be the relations accepted in linear time on an ATM with i alternations, beginning with an existential state. In either case,[1]

$$\mathbf{LTH} = \bigcup_{i=0}^{\infty} \mathbf{\Sigma}_i^{lin}$$

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.

3.4.2 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 $(n_1, \ldots, n_k) \in \mathbb{N}^k$,

$$(n_1, \ldots, n_k) \in R \qquad \text{iff} \qquad \underline{\mathbb{N}} \models \varphi(n_1, \ldots, n_k) \qquad (3.28)$$

[1] **LTH** is different from **LH**, the *logtime-hierarchy* discussed in Section 4.1

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 $\vec{n} \in \mathbb{N}$,

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

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

For example, the class of $\mathbf{\Sigma}_1$-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 3.5), **NP** relations are precisely the $\mathbf{\Sigma}_1^b$-representable relations. ($\mathbf{\Sigma}_1^b$ is defined for the language $\mathcal{L}_{\mathbf{S}_2}$ of $\mathbf{S}_2$.) Here we show that the **LTH** relations are exactly the $\mathbf{\Delta}_0$-representable relations.

Definition 3.56. $\mathbf{\Delta}_0^{\mathbb{N}}$ *is the class of* $\mathbf{\Delta}_0$*-representable relations.*

For instance, we have shown that the relation BIT is in $\mathbf{\Delta}_0^{\mathbb{N}}$; so is the relation $Prime(x)$ (x is a prime number), because

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

Lemma 3.57. *The relation* $Numones(x, y)$*, asserting that* y *is the number of one-bits in the binary notation for* x*, is in* $\mathbf{\Delta}_0^{\mathbb{N}}$*.*

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 $n = |x|$ and s_i is the number of ones in the first i bits of 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" [**?**], 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

$$\begin{aligned} Numones(x, y) \leftrightarrow \exists \langle t_0, \ldots, t_{\sqrt{n}}\rangle,\ t_0 = 0 \wedge t_{\sqrt{n}} = y \wedge \forall i < \sqrt{n}\ \exists \langle u_0, \ldots, u_{\sqrt{n}}\rangle \\ [u_0 = t_i \wedge u_{\sqrt{n}} = t_{i+1} \wedge \forall j < \sqrt{n}\ (u_{j+1} = u_j + FBIT(i\sqrt{n} + j, x))] \end{aligned}$$

where the function $FBIT(i, x)$ is bit i of x. □

Theorem 3.58 (LTH Theorem). $\quad$ $\mathbf{LTH} = \mathbf{\Delta}_0^{\mathbb{N}}$

Proof Sketch. First consider the inclusion $\mathbf{LTH} \subseteq \mathbf{\Delta}_0^{\mathbb{N}}$. The hard part here is to show $\mathbf{NLinTime} \subseteq \mathbf{\Delta}_0^{\mathbb{N}}$. Once this is done we can easily show $\mathbf{\Sigma}_i^{lin} \subseteq \mathbf{\Delta}_0^{\mathbb{N}}$ either by using the recurrence in (3.27), or considering an ATM with i alternations. (Note that on an input $x \in \mathbb{N}$ of length n, a linear time ATM can guess a binary number y of length cn, and the $\mathbf{\Delta}_0$ formula can use the bounded quantifier $\exists y < x^{c+1}$.)

To show $\mathbf{NLinTime} \subseteq \mathbf{\Delta}_0^{\mathbb{N}}$ 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 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 $\mathbf{\Delta}_0^{\mathbb{N}} \subseteq \mathbf{LTH}$ by structural induction on $\mathbf{\Delta}_0$ 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 $\mathbf{LTH}$ we use Corollary 3.60 below which shows that $\mathbf{L} \subseteq \mathbf{LTH}$. ($\mathbf{L}$ is the class of relations computable in logarithmic space using Turing machines. See Appendix A.1.1.) 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 $\mathbf{L}$. □

Theorem 3.59 (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 3.57. 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 to 7) will get the lengths of the quantified strings below linear. □

For the following corollary, **NL** is the class of relations computable by *non-deterministic* Turing machines in logarithmic space. See Appendix A.2.

Corollary 3.60. NL $\subseteq$ LTH.

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

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 **L** $\subset$ **PSPACE** by a simple diagonal argument.

Also **LTH** $\subseteq$ **LinSPACE** $\subset$ **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**.

3.4.3 Characterizing the LTH by $\mathbf{I\Delta}_0$

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

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

Definition 3.61 (FLTH). *A function $f : \mathbb{N}^k \to \mathbb{N}$ is in* **FLTH** *precisely if its graph $G_f(\vec{x}, y)$ is in* **LTH** *and its length has at most linear growth, i.e.,*

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

Theorem 3.62 ($\mathbf{I\Delta}_0$-Definability Theorem). *A function is $\mathbf{\Sigma}_1$-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

$$f(\vec{x}) \leq t(\vec{x}) \tag{3.29}$$

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

$$\mathbf{I\Delta}_0 \vdash \exists y A(y) \supset \exists! y Min_y[A(y)](y) \tag{3.30}$$

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 (3.29). 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 (3.30)

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

3.5 Buss's S_2^i Hierarchy: The Road Not Taken

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

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

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 $\mathbf{\Sigma}_i^b$ (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 $\mathbf{\Sigma}_1^b$ represent precisely the **NP** relations, and more generally formulas in $\mathbf{\Sigma}_i^b$ represent precisely the relations in the level $\mathbf{\Sigma}_i^p$ 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 $\mathbf{\Sigma}_i^b$-**IND** scheme. The axioms for $\mathbf{S}_2^i$ are the same as those of $\mathbf{T}_2^i$, except for $\mathbf{\Sigma}_i^b$-**IND** is replaced by the $\mathbf{\Sigma}_i^b$-**PIND** scheme:

$$[\varphi(0) \wedge \forall x(\varphi(\lfloor \frac{1}{2}x \rfloor) \supset \varphi(x))] \supset \forall x \varphi(x)$$

where $\varphi(x)$ is any $\mathbf{\Sigma}_i^b$ formula. Note that this axiom scheme is true in $\underline{\mathbb{N}}$. Also for $i \geq 1$, $\mathbf{T}_2^i$ proves the $\mathbf{\Sigma}_i^b$-**PIND** axiom scheme, and $\mathbf{S}_2^{i+1}$ proves the $\mathbf{\Sigma}_i^b$-**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 $\mathbf{\Sigma}_i^b$-definable in $\mathbf{S}_2^i$ are precisely those polytime reducible to relations in $\mathbf{\Sigma}_{i-1}^p$ (level $i-1$ of the polynomial hierarchy). In particular, the functions $\mathbf{\Sigma}_1^b$-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 $\mathbf{\Sigma}_1$-definable functions in $\mathbf{S}_2$ are polynomial time reducible to **PH**. To show that the $\mathbf{\Sigma}_1$-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 the following chapters 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. Buss introduced versions of these second-order theories in his thesis, and Razborov and Zambella [**?**] have contributed to their presentation and development.

3.6 Notes

The main references for this chapter are [**?**, **?**] and [**?**, pp 277–293].

Parikh's Theorem originally appears in [**?**], and the proof there is based in the Herbrand Theorem, and resembles our "Alternative Proof" given at the end of Section 3.3.2. Buss [**?**] gives a proof based on cut elimination which is closer to our first proof.

James Bennett [**?**] was the first to show that the relation $y = z^x$ can be defined by $\mathbf{\Delta}_0$ formulas. Hájek and Pudlák [**?**] 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 3.3.3 follows that of Buss in [**?**], simplified with an idea from earlier proofs.

Bennett's Trick, described in the proof of Lemma 3.57, is due to Bennett [**?**] 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 $\mathbf{LTH} = \mathbf{\Delta}_0^{\mathbb{N}}$, is due to Wrathall [**?**]. Nepomnjaščij's Theorem 3.59 appears in [**?**].

Chapter 4

Two-Sorted First-Order Logic

In this chapter we introduce two-sorted first-order logic, an extension of the (single-sorted) first-order logic that we have seen in the previous chapters. Our motivation for this two-sorted logic comes from descriptive complexity theory, where each object (a language or a relation) in a complexity class is described by a logical formula of a certain kind. In fact each object corresponds to the set of all finite models of the formula. In the two-sorted logic setting, each object corresponds to an interpretation of a variable in the formula satisfying the formula in the standard model. Here we also study the corresponding *function classes*: Each class **C** is associated with a theory whose class of provably total functions is exactly **FC**, the function class corresponding to **C**. Our theories are also related to *propositional proof systems* by way of *propositional translation*, a topic to be covered later in the book.

In the first part of this chapter we present a brief introduction to descriptive complexity theory. (A comprehensive treatment can be found in [**?**].) Then we introduce the two-sorted first-order logic, describe the 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.

4.1 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 $L \subseteq \Sigma^*$, where $\Sigma = \{0, 1\}$, and the formula is a formula of the first-order predicate calculus. We assume that the underlying vocabulary consists of

$$\mathcal{L}_{FO} = [0, max; X, BIT, \leq, =], \tag{4.1}$$

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 $M = \{0, .., n-1\}$ 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 of nonempty binary strings $\{0,1\}^+$.

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:

$$L(\varphi) =_{def} \{X \in \{0,1\}^+ \mid \mathcal{M}[X] \models \varphi\}.$$

Definition 4.1 (The Class FO).

$$\mathbf{FO} = \{L \mid L = L(\varphi) \textit{ for some } \mathcal{L}_{FO}\textit{-sentence } \varphi\}$$

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 [**?**, 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

$$\forall x \forall y,\ x + y = max \supset (X(x) \leftrightarrow X(y)).$$

Thus $PAL \in \mathbf{FO}$.

Immerman showed that the class **FO** is the same as a uniform version of $\mathbf{AC}^0$ (see Appendix A.4.1). 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 [**?**, 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 (and independently Furst, Saxe, and Sipser) 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 $\mathbf{AC}^0$. Do not confuse this with the result of [**?**, 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 structures with universe $M = \{0, ..., n-1\}$.

4.2 Two-Sorted First-Order Logic

4.2.1 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 2.1), 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 2.3):

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 2.1 and 2.2). 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.

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 $\mathcal{L}$, $t_1, \ldots, t_n$ are $\mathcal{L}$-number terms, and $T_1, \ldots, T_m$ are $\mathcal{L}$-string terms, then $ft_1 \ldots t_n T_1 \ldots T_m$ is an $\mathcal{L}$-number term.*
4) *If F is an (n, m)-ary string function symbol of $\mathcal{L}$, and $t_1, \ldots, t_n$ and $T_1, \ldots, T_m$ are as above, then $Ft_1 \ldots t_n T_1 \ldots T_m$ is an $\mathcal{L}$-string term.*

Note that all constants in $\mathcal{L}$ are $\mathcal{L}$-terms.

We often denote number terms by $r, s, t, \ldots$, and string terms by $S, T, \ldots$.

The formulas over a two-sorted language $\mathcal{L}$ are defined as in the single-sorted case (Definition 2.2), 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 $\mathcal{L}$, $t_1,\ldots,t_n$ are $\mathcal{L}$-number terms and $T_1,\ldots,T_m$ are $\mathcal{L}$-string terms, then $Pt_1\ldots t_nT_1\ldots T_m$ 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 $\varphi,\psi,\ldots$.
Recall that in $\mathcal{L}^2_A$ we write $X(t)$ for $t\in X$.

Example 4.7 ($\mathcal{L}^2_A$-Terms and $\mathcal{L}^2_A$-Formulas).

1) *The only string terms of $\mathcal{L}^2_A$ are the string variables $X,Y,Z,\ldots$.*
2) *The number terms of $\mathcal{L}^2_A$ 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 $\mathcal{L}^2_A$ are $\bot$, $\top$ or those of the form $s=t$, $X=Y$, $s\le t$ and $X(t)$ for string variables X,Y and number terms s,t.*

4.2.2 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 2.6.

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 $\mathcal{L}^2_A$, an $\mathcal{L}^2_A$-structure with universe $\langle U_1,U_2\rangle$ contains the following interpretations of $\mathcal{L}^2_A$:

- 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 2.7):

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 (2.8) and the notion $\mathcal{M} \models \varphi[\sigma]$ (Definition 2.9) generalize 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 2.11), and we do not repeat them here. Also, the Substitution Theorem (2.15) generalizes to the current context, and the Formula Replacement Theorem (2.16) continues to hold, and we will not restate them.

4.3 Two-sorted Complexity Classes

4.3.1 Notation for Numbers and Finite Sets

In Section 3.4 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 $x_1, \ldots, x_k$ of a relation $R(x_1, \ldots, x_k)$ are presented in binary to the accepting machine. In the two-sorted context, however, the relations $R(x_1, \ldots, x_k, X_1, \ldots, X_m)$ 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 $\bot$ and 1 for $\top$, then we can use the binary string

$$w(S) = S(n)S(n-1)\ \dots\ S(1)S(0) \tag{4.2}$$

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 arguments X_i is represented as 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**.

4.3.2 Representation Theorems

Notation If $\vec{T} = T_1, \dots T_n$, is a sequence of string terms, then $|\vec{T}|$ denotes the sequence $|T_1|, \dots, |T_n|$ 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 $\exists \vec{x} \leq \vec{t}\varphi$ stands for $\exists x_1 \leq t_1 \ldots \exists x_k \leq t_k \varphi$ for some k, where no x_i occurs in any t_j (even if $i < j$). Similarly for $\forall \vec{x} \leq \vec{t}$, $\exists \vec{X} \leq \vec{t}$, and $\forall \vec{X} \leq \vec{t}$.

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 $\exists x_1 \leq t_1 \exists x_2 \leq t_2(x_1)\varphi$ is equivalent to

$$\exists x_1 \leq t_1 \exists x_2 \leq t_2(t_1)(x_2 \leq t_2(x_1) \wedge \varphi)$$

We will now define the following important classes of formulas.

Definition 4.14 (The $\mathbf{\Sigma}_1^1(\mathcal{L})$, $\mathbf{\Sigma}_i^B(\mathcal{L})$ and $\mathbf{\Pi}_i^B(\mathcal{L})$ Formulas). *Let $\mathcal{L} \supseteq \mathcal{L}_A^2$ be a two-sorted language. Then $\mathbf{\Sigma}_0^B(\mathcal{L}) = \mathbf{\Pi}_0^B(\mathcal{L})$ is the set of $\mathcal{L}$-formulas whose only quantifiers are bounded number quantifiers (there can be free string variables). For $i \geq 0$, $\mathbf{\Sigma}_{i+1}^B(\mathcal{L})$ (resp. $\mathbf{\Pi}_{i+1}^B(\mathcal{L})$) 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 $\mathbf{\Pi}_i^B(\mathcal{L})$ formula (resp. a $\mathbf{\Sigma}_i^B(\mathcal{L})$ formula), and $\vec{t}$ is a sequence of $\mathcal{L}_A^2$-terms not involving any variable in $\vec{X}$. Also, a $\mathbf{\Sigma}_1^1(\mathcal{L})$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L})$ formula.*

We will drop mention of $\mathcal{L}$ when it is clear from the context. Thus

$$\mathbf{\Sigma}_0^B \subseteq \mathbf{\Sigma}_1^B \subseteq \mathbf{\Sigma}_2^B \subseteq \cdots$$
$$\mathbf{\Sigma}_0^B \subseteq \mathbf{\Pi}_1^B \subseteq \mathbf{\Pi}_2^B \subseteq \cdots$$

and for $i \geq 0$

$$\mathbf{\Sigma}_i^B \subseteq \mathbf{\Pi}_{i+1}^B \quad \text{and} \quad \mathbf{\Pi}_i^B \subseteq \mathbf{\Sigma}_{i+1}^B$$

Notice the "strict" requirements on $\mathbf{\Sigma}_i^B(\mathcal{L})$ and $\mathbf{\Pi}_i^B(\mathcal{L})$: formulas of these classes must be in prenex form, with no string quantifier occurs within the scope of any number quantifier. For example, $\mathbf{\Sigma}_1^B(\mathcal{L}_A^2)$ is usually called *strict* $\Sigma_1^{1,b}$ by other authors. Also notice that the bounding terms $\vec{t}$ must be in the basic language $\mathcal{L}_A^2$.

In Section 3.3.1 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 R 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*

$$R(\vec{x}, \vec{X}) \leftrightarrow \varphi(\vec{x}, \vec{X}) \tag{4.3}$$

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 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 $w(X)$ (4.2) by deleting the initial 1 (and $w'(\varnothing)$ is the empty string).

Example 4.16. *The language PAL (page 68) of binary palindromes is represented by the formula*

$$\varphi_{PAL}(X) \leftrightarrow (|X| \leq 1) \vee \forall x, y < |X|,\ x + y + 2 = |X| \supset (X(x) \leftrightarrow X(y))$$

Two-sorted $\mathbf{AC}^0$ restricted to numerical relations $R(\vec{x})$ is exactly the same as single-sorted **LTH** as defined in Section 3.4.1. Both classes can be defined in terms of alternating Turing machines: $\mathbf{AC}^0$ requires log time and constant alternations, and **LTH** requires linear time and constant alternations. But for $\mathbf{AC}^0$, numbers are represented in unary notation, so the length n of an input x is x, whereas for **LTH**, numbers are represented in binary, so the length n of an input x is about $\log x$. Since an ATM accesses its input tape using an index register written in binary (and it can guess and verify the binary number), it does not matter whether an input number is written in binary or unary.

Thus for numerical relations, the following representation theorem is the same as the **LTH** Theorem 3.58 ($\mathbf{LTH} = \mathbf{\Delta}_0^{\mathbb{N}}$). For string relations, it can be considered a restatement of Theorem 4.2 ($\mathbf{FO} = \mathbf{AC}^0$).

Theorem 4.17 ($\mathbf{\Sigma}_0^B$ Representation Theorem). *A relation $R(\vec{x}, \vec{X})$ is in $\mathbf{AC}^0$ iff it is represented by some $\mathbf{\Sigma}_0^B$ formula $\varphi(\vec{x}, \vec{X})$.*

Proof. In light of the above discussion, the proof is essentially the same as for Theorem 3.58. The string arguments pose no problem: Each X_i is represented on the input tape of the ATM by the binary string $w(X_i)$ whose bits can be accessed in the formula $\varphi(\vec{x}, \vec{X})$ by atomic formulas $X_i(t)$ for suitable terms t. □

Notation For X a finite subset of $\mathbb{N}$, let $bin(X)$ be the number whose binary notation is $w(X)$ (see (4.2)). Thus

$$bin(X) = \sum_i X(i) 2^i \tag{4.4}$$

where here we treat the predicate $X(i)$ as a 0–1-valued function. 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 $\mathbf{\Sigma}_0^B$ formula. However R_+ is in $\mathbf{AC}^0$. To represent it as a $\mathbf{\Sigma}_0^B$ 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)

$$Carry(i, X, Y) \leftrightarrow \exists k < i, (X(k) \wedge Y(k)) \wedge \forall j < i[k < j \supset (X(j) \vee Y(j))] \quad (4.5)$$

Thus

$$\begin{aligned} R_+(X, Y, Z) \leftrightarrow & |Z| \leq |X| + |Y| \wedge \forall i < |X| + |Y|, \\ & Z(i) \leftrightarrow (X(i) \oplus Y(i) \oplus Carry(i, X, Y)) \end{aligned}$$

where $\oplus$ represents exclusive or.

Note that the $\mathbf{\Sigma}_0^B$ 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 (4.1) can be equivalently defined as

$$[0, max, +, \cdot; X, \leq, =],$$

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 $\mathbf{\Sigma}_0^B$ Representation Theorem can be proved by translating any $\mathbf{\Sigma}_0^B$ formula φ into an $\mathbf{FO}$ formula φ' that *describes* the relation *represented* by φ, and vice versa.

We use $\mathbf{\Sigma}_i^P$ to denote level $i \geq 1$ of the two-sorted polynomial hierarchy. In particular, $\mathbf{\Sigma}_1^P$ denotes two-sorted $\mathbf{NP}$. Thus a relation $R(\vec{x}, \vec{X})$ is in $\mathbf{\Sigma}_i^P$ iff it is accepted by some polynomial time ATM with at most i alternations, starting with existential, using the input conventions described in Section 4.3.1.

Theorem 4.18 ($\mathbf{\Sigma}_i^B$ and $\mathbf{\Sigma}_1^1$ Representation Theorem). *For $i \geq 1$, a relation $R(\vec{x}, \vec{X})$ is in $\mathbf{\Sigma}_i^P$ iff it is represented by some $\mathbf{\Sigma}_i^B$ formula. The relation is recursively enumerable iff it is represented by some $\mathbf{\Sigma}_1^1$ formula.*

Proof. We show that a relation $R(\vec{x}, \vec{X})$ is in $\mathbf{NP}$ iff it is represented by a $\mathbf{\Sigma}_1^B$ formula. First suppose that $R(\vec{x}, \vec{X})$ is accepted by a nondeterministic polytime Turing machine M. Then the $\mathbf{\Sigma}_1^B$ formula that represents R has the form

$$\exists Y \leq t(\vec{x}, \vec{X})\ \varphi(\vec{x}, \vec{X}, Y)$$

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 $\mathbf{\Sigma}_0^B$ 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 $\mathbf{\Sigma}_1^B$ formula

$$\exists \vec{Y} \le \vec{t}(\vec{x}, \vec{X})\ \varphi(\vec{x}, \vec{X}, \vec{Y})$$

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 $\mathbf{\Sigma}_0^B$ Representation Theorem). □

4.3.3 The LTH Revisited

Consider **LTH** (Linear Time Hierarchy, Section 3.4) as a two-sorted complexity class. Here we can define the relations in this class by *linearly bounded formulas*, a concept defined below.

Definition 4.19. *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 4.20 (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 $t(x_1, \dots, x_k, |X_1|, \dots, |X_m|)$ 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 4.21. 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 $X(im)\ \dots\ X(im + m - 1)$".*

c) *Now define $\mathit{Numones}_2(a, i, X)$ using linearly bounded formula.*

Exercise 4.22. *Complete the proof of the Two-Sorted* **LTH** *Theorem.*

In [**?**], 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 4.23. *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 $\mathbf{\Delta}_0$ formula in Section 3.3.3. Here it is much easier to define this relation using linearly bounded formulas.

Exercise 4.24. *Show how to express $y = 2^x$ using linearly bounded formula. (Hint: Use $Numones_2$ from Exercise 4.21.)*

4.4 The Proof System $\mathbf{LK}^2$

Now we extend the sequent system **LK** (Section 2.3) 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 2.3, 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

$$\textbf{left}: \frac{\varphi(T), \Gamma \longrightarrow \Delta}{\forall X \varphi(X), \Gamma \longrightarrow \Delta} \qquad \textbf{right}: \frac{\Gamma \longrightarrow \Delta, \varphi(\beta)}{\Gamma \longrightarrow \Delta, \forall X \varphi(X)}$$

String $\exists$ introduction rules

$$\textbf{left}: \frac{\varphi(\beta), \Gamma \longrightarrow \Delta}{\exists X \varphi(X), \Gamma \longrightarrow \Delta} \qquad \textbf{right}: \frac{\Gamma \longrightarrow \Delta, \varphi(T)}{\Gamma \longrightarrow \Delta, \exists X \varphi(X)}$$

Restriction The free variable β must not occur in the conclusion of $\forall$-**right** and $\exists$-**left**.

The notions of $\mathbf{LK}^2$ *proofs* and $\mathbf{LK}^2$ *anchored proofs* generalize the notion of **LK** proofs and anchored **LK** proofs. Then the Derivational Soundness, the Completeness Theorem (2.24), and the Anchored Completeness Theorem (2.29) 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 2.22. 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 2.36. Recall that we write $=$ for both $=_1$ and $=_2$.

Definition 4.25 ($\mathbf{LK}^2$ 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 $\mathbf{E4}'$, $\mathbf{E4}''$ *and* $\mathbf{E5}'$.*) Here t, u, t_i, u_i are number terms, and T, U, T_i, U_i are string terms.*

$\mathbf{E1}'$. $\longrightarrow t = t$
$\mathbf{E1}''$. $\longrightarrow T = T$
$\mathbf{E2}'$. $t = u \longrightarrow u = t$
$\mathbf{E2}''$. $T = U \longrightarrow U = T$
$\mathbf{E3}'$. $t = u, u = v \longrightarrow t = v$
$\mathbf{E3}''$. $T = U, U = V \longrightarrow T = V$
$\mathbf{E4}'$. $\Lambda \longrightarrow ft_1 \dots t_n T_1 \dots T_m = fu_1 \dots u_n U_1 \dots U_m$ *for each f in $\mathcal{L}$*
$\mathbf{E4}''$. $\Lambda \longrightarrow Ft_1 \dots t_n T_1 \dots T_m = Fu_1 \dots u_n U_1 \dots U_m$ *for each F in $\mathcal{L}$:*
$\mathbf{E5}'$. $\Lambda, Pt_1 \dots t_n T_1 \dots T_m \longrightarrow Pu_1 \dots u_n U_1 \dots U_m$ *for each P in $\mathcal{L}$ (here P is not $=_1$ or $=_2$).*

Definition 4.26 ($\mathbf{LK}^2$-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 is an* $\mathbf{LK}^2$-Ψ *proof in the sense of Definition 2.22, where Ψ is Φ together with all instances of the* $\mathbf{LK}^2$ *Equality Axioms* $\mathbf{E1}'$, $\mathbf{E1}''$, …, $\mathbf{E4}'$, $\mathbf{E4}''$, $\mathbf{E5}'$ *for $\mathcal{L}$. If Φ is empty, we simply refer to an* $\mathbf{LK}^2$*-proof (but allow* $\mathbf{E1}', \dots, \mathbf{E5}'$ *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 2.38, and is proved in the same way.

Theorem 4.27 (Soundness and Completeness of $\mathbf{LK}^2$). *For any set Φ of formulas and sequent S,*

$$\forall\Phi \models S \textit{ iff } S \textit{ has an } \mathbf{LK}^2\text{-}\Phi \textit{ proof}$$

Below we will state the two-sorted analogue of the Anchored **LK** Completeness Theorem and the Subformula Property of Anchored **LK** Proofs (Theorems 2.40 and 2.41). They can be proved just as in the case of **LK**.

Definition 4.28 (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* $\mathbf{E1}', \mathbf{E1}'', \ldots, \mathbf{E5}'$*).*

Theorem 4.29 (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 4.30 (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* $\mathbf{E1}', \ldots, \mathbf{E4}'', \mathbf{E5}'$*).*

As in the case for **LK** where the Anchored **LK** Completeness Theorem is used to prove the Compactness Theorem (Theorem 2.43), the above Anchored $\mathbf{LK}^2$ Completeness Theorem can be used to prove the following (two-sorted) Compactness Theorem.

Theorem 4.31 (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 1.16.)

Form 1 of the Herbrand Theorem (Theorem 2.49) 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 $\mathbf{E1}'$, $\mathbf{E1}''$, ..., $\mathbf{E4}''$, $\mathbf{E5}'$ 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 4.32 (Herbrand Theorem for Two-Sorted Logic). a) *Let* $\mathcal{T}$ *be a universal (two-sorted) theory, and let* $\varphi(x_1, \ldots, x_k, X_1, \ldots, X_m, Z)$ *be a quantifier-free formula with all free variables displayed such that*

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_k \forall X_1 \ldots \forall X_m \exists Z \varphi(\vec{x}, \vec{X}, Z).$$

Then there exist finitely many string terms $T_1(\vec{x}, \vec{X}), \ldots, T_n(\vec{x}, \vec{X})$ *such that*

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_k \forall X_1 \ldots \forall X_m \; [\varphi(\vec{x}, \vec{X}, T_1(\vec{x}, \vec{X})) \vee \ldots \vee \varphi(\vec{x}, \vec{X}, T_n(\vec{x}, \vec{X}))]$$

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*

$$\mathcal{T} \vdash \forall x_1 \ldots \forall x_k \forall X_1 \ldots \forall X_m \; [\varphi(\vec{x}, t_1(\vec{x}, \vec{X}), \vec{X}) \vee \ldots \vee \varphi(\vec{x}, t_n(\vec{x}, \vec{X}), \vec{X})]$$

The theorem easily extends to the cases where

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists Z_1 \ldots \exists Z_n \varphi(\vec{x}, \vec{X}, \vec{Z}).$$

and

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists z_1 \ldots \exists z_n \varphi(\vec{x}, \vec{z}, \vec{X}).$$

4.4.1 Two-Sorted Free Variable Normal Form

The notion of free variable normal form (Section 2.3.1) 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 $\mathcal{L}_A^2$, there is no string constant symbol, so we expand the notion of a $\mathbf{LK}^2$-Φ proof over $\mathcal{L}_A^2$ 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 $\mathcal{L}_A^2$ 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 $\mathcal{L}_A^2$ can be transformed to one in free variable normal form with the same endsequent, and similarly for $\mathbf{LK}^2$-Φ for suitable Φ.

4.5 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 4.4 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
$$\forall\vec{x}\forall\vec{y},\ (\mathsf{FS}(x_1) \wedge \ldots \mathsf{FS}(x_n) \wedge \mathsf{SS}(y_1) \ldots \wedge \mathsf{SS}(y_m)) \supset \mathsf{FS}(f(\vec{x},\vec{y}))$$
(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
$$\forall\vec{x}\forall\vec{y},\ (\mathsf{FS}(x_1) \wedge \ldots \mathsf{FS}(x_n) \wedge \mathsf{SS}(y_1) \ldots \wedge \mathsf{SS}(y_m)) \supset \mathsf{SS}(F(\vec{x},\vec{y}))$$
(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
$$\forall\vec{x}\forall\vec{y},\ P(\vec{x},\vec{y}) \supset (\mathsf{FS}(x_1) \wedge \ldots \mathsf{FS}(x_n) \wedge \mathsf{SS}(y_1) \ldots \wedge \mathsf{SS}(y_m))$$

Lemma 4.33. *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 4.34. *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 4.33. 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 4.33. Now we can prove by structural induction on φ that $\mathcal{M}$ is a model for φ. Therefore φ is also satisfiable. □

Exercise 4.35. *Prove the Compactness Theorem for the two-sorted logic (4.31) from the Compactness Theorem for single-sorted logic (2.43).*

Exercise 4.36. *Prove the Herbrand Theorem for the two-sorted logic (4.32) from Form 2 of the Herbrand Theorem for single-sorted logic (3.38).*

4.6 Notes

The main reference for Section 4.1 is [**?**] Sections 1.1, 1.2, 5.5. Our two-sorted language $\mathcal{L}_A^2$ is from Zambella [**?**, **?**].

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 theories $\mathbf{S}^i_2$ (Section 3.5). The theory $\mathbf{V}^0$ characterizes $\mathbf{AC}^0$ in the same way that $\mathbf{I\Delta}_0$ characterizes $\mathbf{LTH}$. 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 $\mathcal{L}^2_A$, together with $\mathbf{\Sigma}^B_0$-**COMP** (the comprehension axiom scheme for $\mathbf{\Sigma}^B_0$ formulas). For $i \geq 1$, $\mathbf{V}^i$ is the same as $\mathbf{V}^0$ except that $\mathbf{\Sigma}^B_0$-**COMP** is replaced by $\mathbf{\Sigma}^B_i$-**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 $\mathbf{LTH}$ by $\mathbf{I\Delta}_0$ (Theorem 3.62). 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.

5.1 Definition and Basic Properties of $\mathbf{V}^i$

The set 2-**BASIC** of axioms is given in Figure 5.1. Recall that $t < u$ stands for $(t \leq u \wedge t \neq u)$.

Axioms **B1**, …, **B8** are taken from the axioms in 1-**BASIC** for $\mathbf{I\Delta}_0$, and **B9**, …, **B12** are theorems of $\mathbf{I\Delta}_0$ (see Examples 3.8 and 3.9). Axioms **L1** and **L2** 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. $x + 1 \neq 0$	**B7.** $(x \leq y \wedge y \leq x) \supset x = y$
B2. $x + 1 = y + 1 \supset x = y$	**B8.** $x \leq x + y$
B3. $x + 0 = x$	**B9.** $0 \leq x$
B4. $x + (y + 1) = (x + y) + 1$	**B10.** $x \leq y \vee y \leq x$
B5. $x \cdot 0 = 0$	**B11.** $x \leq y \leftrightarrow x < y + 1$
B6. $x \cdot (y + 1) = (x \cdot y) + x$	**B12.** $x \neq 0 \supset \exists y \leq x(y + 1 = x)$
L1. $X(y) \supset y < \lvert X\rvert$	**L2.** $y + 1 = \lvert X\rvert \supset X(y)$
SE. $[\lvert X\rvert = \lvert Y\rvert \wedge \forall i < \lvert X\rvert(X(i) \leftrightarrow Y(i))] \ \supset \ X = Y$	

Figure 5.1: 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. *Using* 2-**BASIC***, show that*

a) $\neg x < 0$.
b) $x < x + 1$.
c) $0 < x + 1$.
d) $x < y \supset x + 1 \leq y$. *(Use* **B10***,* **B11***,* **B7***.)*
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*

$$\exists X \leq y \forall z < y(X(z) \leftrightarrow \varphi(z)), \tag{5.1}$$

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 $\mathbf{\Sigma}_i^B$.

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* $\mathbf{\Sigma}_i^B$-**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*

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

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*

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

*where φ is a formula in Φ. Similarly the number maximization axioms for Φ are denoted Φ-**MAX** and consist of the formulas*

$$\varphi(0) \supset \exists x \le y,\ \varphi(x) \wedge \neg\exists z \le y(x < z \wedge \varphi(z))$$

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 72). 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 $\mathbf{\Delta}_0$ formulas. In fact, we will show that it proves $\mathbf{\Sigma}_0^B$-**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 $\mathbf{\Sigma}_0^B$-**COMP** there is a set Y such that $|Y| \le |X|$ and for all $z < |X|$

$$Y(z) \leftrightarrow \forall y \le z\ \neg X(y) \qquad (5.2)$$

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 **B12** and **L2**, hence (ii) holds vacuously by Exercise 5.1 **a**. Also, $X(0)$ holds, since otherwise $Y(0)$ holds by **B7** and **B9**. Thus we have proved (i).

Now suppose that Y is not empty, i.e., $Y(y)$ holds for some y. Then $y < |Y|$ by **L1**, and thus $|Y| \ne 0$ by Exercise 5.1 **a**. By **B12**, $|Y| = z+1$ for some z and hence $(Y(z) \wedge \neg Y(z+1))$ by **L1** and **L2**. Hence by (5.2) we have

$$\forall y \le z\ \neg X(y)\ \wedge\ \exists i \le z+1\ X(i)$$

It follows that $i = z+1$ in the second conjunct, since if $i < z+1$ then $i \le z$ by **B11**, which contradicts the first conjunct. This establishes (i) and (ii), since $i = z+1 = |Y|$. □

Consider the following instance of $\mathbf{\Sigma}_0^B$-**IND**:

$$X\textbf{-IND} \;\equiv\; [X(0) \wedge \forall y < z(X(y) \supset X(y+1))] \supset X(z)$$

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 $\mathbf{\Sigma}_0^B$-**COMP**, there is a set Y with $|Y| \le z+1$ 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 **L1**. 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 **B12**. 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 $\mathbf{\Sigma}_0^B$-**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

$$\mathcal{T} \vdash [\varphi(0) \wedge \forall y,\ \varphi(y) \supset \varphi(y+1)] \supset\ \varphi(z)$$

Reasoning in $\mathbf{V}^0$, assume

$$\varphi(0) \wedge \forall y,\ \varphi(y) \supset \varphi(y+1) \tag{5.3}$$

By Φ-**COMP**, there exists X such that $|X| \le z+1$ and

$$\forall y < z+1\ (X(y) \leftrightarrow \varphi(y)). \tag{5.4}$$

By **B11**, Exercise 5.1 **c,e** and (5.3) 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 (5.4) and Exercise 5.1 **b**. □

It follows from the corollary that for all $i \geq 0$, $\mathbf{V}^i$ proves $\mathbf{\Sigma}_i^B$-**IND**, $\mathbf{\Sigma}_i^B$-**MIN**, and $\mathbf{\Sigma}_i^B$-**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 $\mathbf{B1}, \ldots, \mathbf{B8}$ and the $\mathbf{\Delta}_0$-**IND** axioms. Since $\mathbf{B1}, \ldots, \mathbf{B8}$ are also axioms of $\mathbf{V}^0$, and we have just shown that $\mathbf{V}^0$ proves the $\mathbf{\Sigma}_0^B$-**IND** axioms (which include the $\mathbf{\Delta}_0$-**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.)
$\square$

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 $\mathbf{B1}, \ldots, \mathbf{B12}$, so $\mathcal{M}$ satisfies these axioms. According to the semantics for $\mathcal{L}_A^2$ (Section 4.2.2), 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 **L1** and **L2**, 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 $\mathbf{\Sigma}_0^B$-**COMP** axioms must be satisfied. We will construct U_2 to consist of all bounded subsets of M defined by $\mathbf{\Delta}_0$-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

$$U_2 = \{S(c, \varphi(x, \vec{d})) \mid c, d_1, ..., d_k \in M \text{ and } \varphi(x, \vec{y}) \text{ is a } \mathbf{\Delta}_0(\mathcal{L}_A) \text{ formula}\} \quad (5.5)$$

We must show that every nonempty element S of U_2 has a largest element, so that $|S|$ can be defined to satisfy **L1** and **L2**. The largest element exists because the differences between the upper bound c for S and elements of S have a minimum element, by $\mathbf{\Delta}_0$-**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 $\ell_\varphi(c, \vec{d}) = c \dot{-} m$.

Exercise 5.11. *Show that for each $\mathbf{\Delta}_0$ formula $\varphi(x, \vec{y})$, the function $\ell_\varphi(z, \vec{y})$ (extended to have the value 0 when $S(z, \varphi(x, \vec{y}))$ is empty) is provably total in $\mathbf{I\Delta}_0$.*

It remains to show that $\mathbf{\Sigma}_0^B$-**COMP** holds in $\mathcal{M}'$. This means that for every $\mathbf{\Sigma}_0^B$ 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

$$T = \{e \in M \mid e < c \text{ and } \mathcal{M}' \models \psi(e, \vec{d}, \vec{S})\} \tag{5.6}$$

must be in U_2. Suppose that

$$S_i = S(c_i, \varphi_i(u, \vec{d}_i))$$

for some $\mathbf{\Delta}_0$ 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 $\mathbf{\Sigma}_0^B$ 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 (5.6). Since the functions ℓ_{φ_i} are $\mathbf{\Sigma}_1$-definable in $\mathbf{I\Delta}_0$, by the Conservative Extension Lemma 3.35, θ can be transformed into an equivalent $\mathbf{\Delta}_0(\mathcal{L}_A)$ formula. Thus $T \in U_2$. □

Exercise 5.12. *Suppose that instead of defining U_2 according to (5.5), we defined U_2 to consist of all subsets of M which have a largest element, together with $\varnothing$. Explain why the $\mathbf{\Sigma}_0^B$-**COMP** axioms may not be satisfied in the resulting structure $\langle U_1, U_2 \rangle$.*

Exercise 5.13. *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.

5.2 Two-Sorted Functions

Complexity classes of two-sorted relations were discussed in Section 4.3 Now we associate with each two-sorted complexity class **C** of relations a two-sorted function class **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 5.14. *A function f or F is* polynomially bounded *(or* p-bounded*) if there is a polynomial $p(\vec{x}, \vec{y})$ such that $f(\vec{x}, \vec{Y}) \leq p(\vec{x}, |\vec{Y}|)$ or $|F(\vec{x}, \vec{Y})| \leq p(\vec{x}, |\vec{Y}|)$.*

All function complexity classes we consider here contain only p-bounded functions.

A natural way of defining function classes is in terms of *bit graph*.

Definition 5.15 (Bit Graph). *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 5.16 (Function Class). *If* **C** *is a two-sorted complexity class of relations, then the corresponding functions class* **FC** *consists of all p-bounded number functions whose graphs are in* **C***, together with all p-bounded string functions whose bit graphs are in* **C***.*

In particular, the string functions in $\mathbf{FAC}^0$ are those p-bounded functions whose bit graphs are in $\mathbf{AC}^0$.

The following characterization of $\mathbf{FAC}^0$ follows from the above definitions and the $\mathbf{\Sigma}_0^B$ Representation Theorem (Theorem 4.17).

Corollary 5.17. *A string function is in $\mathbf{FAC}^0$ if and only if it is p-bounded, and its bit graph is represented by a $\mathbf{\Sigma}_0^B$ 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 (4.4) 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 $\mathbf{\Sigma}_0^B$ formula given in (4.5).

The bit graph of $X + Y$ can be defined as follows.

Example 5.18 (Bit Graph of String Addition). *The bit graph of $X + Y$ is*

$$(X + Y)(i) \leftrightarrow i < |X| + |Y| \ \wedge\ [X(i) \oplus Y(i) \oplus Carry(i, X, Y)] \tag{5.7}$$

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

$$G_F(\vec{x}, \vec{Y}, Z) \ \leftrightarrow\ |Z| \le t \ \wedge\ \forall i < t\,(Z(i) \leftrightarrow B_F(i, \vec{x}, \vec{Y})) \tag{5.8}$$

where t is the bound on the length of F.

As we noted earlier (Section 4.1), the relation $R_\times$ is not in $\mathbf{AC}^0$, where

$$R_\times(X, Y, Z) \leftrightarrow \mathit{bin}(X) \cdot \mathit{bin}(Y) = \mathit{bin}(Z)$$

(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 $\mathbf{\Sigma}_0^B$ formula, where $X \times Y = \mathit{bin}(X) \cdot \mathit{bin}(Y)$ 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 $\mathbf{LTH}$ relations in the single-sorted setting (see the proof of the $\mathbf{\Sigma}_0^B$ Representation Theorem, 4.17), number functions with no string arguments are $\mathbf{AC}^0$ functions iff they are single-sorted $\mathbf{LTH}$ 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 5.19. *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 5.20. 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*

$$f = \begin{cases} g & \textit{if } \varphi, \\ h & \textit{otherwise} \end{cases} \qquad\qquad F = \begin{cases} G & \textit{if } \varphi, \\ H & \textit{otherwise} \end{cases}$$

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 $\mathbf{\Sigma}_0^B$ formula.

By the $\mathbf{\Sigma}_0^B$ Representation Theorem (4.17) there is a $\mathbf{\Sigma}_0^B$ 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 5.17 there is a $\mathbf{\Sigma}_0^B$ formula $\theta(i, \vec{x}, \vec{X})$ and a number term $t(\vec{x}, \vec{X})$ such that

$$F(\vec{x}, \vec{X})(i) \leftrightarrow i < t(\vec{x}, \vec{X}) \wedge \theta(i, \vec{x}, \vec{X}). \tag{5.9}$$

It follows from Exercise 5.19 that the relation $z = |F(\vec{x}, \vec{X})|$ is represented by a $\mathbf{\Sigma}_0^B$ formula η, so

$$z = |F(\vec{x}, \vec{X})| \;\leftrightarrow\; \eta(z, \vec{x}, \vec{X}) \tag{5.10}$$

The $\mathbf{\Sigma}_0^B$ 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 (5.9) and (5.10) 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

$$\exists z \le t(\vec{x}, \vec{X})\ (\eta(z, \vec{x}, \vec{X}) \wedge \psi(\vec{x}, \vec{X}, z)).$$

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 $\mathbf{\Sigma}_0^B$ formula which represents $Q(\vec{x}, \vec{X})$. □

Exercise 5.21. *Prove part* **a)** *of Theorem 5.20 for the case of substitution of number functions for variables. Also prove parts* **b)** *and* **c)** *of the theorem.*

5.3 Parikh's Theorem for Two-Sorted Logic

Recall (Section 3.2) that a term $t(\vec{x})$ is a bounding term for a function symbol f in a single-sorted theory $\mathcal{T}$ if

$$\mathcal{T} \vdash \forall \vec{x}\ f(\vec{x}) \le t(\vec{x})$$

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

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ f(\vec{x}, \vec{X}) \le t(\vec{x}, \vec{X})$$

Also, $t(\vec{x}, \vec{X})$ is a bounding term for a string function F in $\mathcal{T}$ if

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ |F(\vec{x}, \vec{X})| \le t(\vec{x}, \vec{X})$$

Definition 5.22. *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 5.23. *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*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ s(\vec{x}, \vec{X}) \le t(\vec{x}, \vec{X}) \qquad \textit{and} \qquad \mathcal{T} \vdash \forall \vec{x} \forall \vec{X}\ |T(\vec{x}, \vec{X})| \le t(\vec{x}, \vec{X})$$

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 5.24 (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 $\mathbf{V}^i, i \ge 0$, is a polynomial-bounded theory. In fact, all two-sorted theories considered in this book are polynomial-bounded.

Theorem 5.25 (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*

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \exists \vec{Y} \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y}) \tag{5.11}$$

Then

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \le t \exists \vec{Y} \le t \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y}) \tag{5.12}$$

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 5.23 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 5.26. *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

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists z \le t(\vec{x}, \vec{X})\varphi(z, \vec{x}, \vec{X})$$

for some term $t(\vec{x}, \vec{X})$ with all variables indicated.

Proof of Parikh's Theorem from Lemma 5.26. Define (omitting $\vec{x}$ and $\vec{X}$)

$$\psi(z) \equiv \exists\vec{y} \le z\exists\vec{Y} \le z\varphi(\vec{y}, \vec{Y})$$

From the assumption (5.11) we conclude that $\mathcal{T} \vdash \exists z\psi(z)$, since we can take

$$z = y_1 + ... + y_k + |Y_1| + ... + |Y_\ell|$$

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 (5.12) follows. □

Proof of Lemma 5.26. The proof is the same as the proof of Parikh's Theorem in the single-sorted logic (page 40), with minor modifications. Refer to Section 4.4 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 4.4.1).

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 4.4.1, 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 $\mathbf{E}$: $|\varnothing| = 0$.

The **Cases I–V** are supplemented to consider the four string quantifier rules, which are treated in the same way as their **LK** counterparts. □

5.4 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 5.27 (Two-Sorted Definability). *Let $\mathcal{T}$ be a theory with vocabulary $\mathcal{L} \supseteq \mathcal{L}^2_A$, 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*

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists!y\varphi(\vec{x}, y, \vec{X}) \tag{5.13}$$

and

$$y = f(\vec{x}, \vec{X}) \leftrightarrow \varphi(\vec{x}, y, \vec{X}) \tag{5.14}$$

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*

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists!Y\varphi(\vec{x}, \vec{X}, Y) \tag{5.15}$$

and

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \varphi(\vec{x}, \vec{X}, Y) \tag{5.16}$$

Then (5.14) is a defining axiom *for f and (5.16) is a* defining axiom *for F. We say that f or F is* definable in $\mathcal{T}$ *if it is Φ-definable in $\mathcal{T}$ for some Φ.*

The Extension by Definition Theorem (3.30) continues to hold. In particular, adding a definable function symbol together with its defining axiom to a two-sorted theory $\mathcal{T}$ results in a conservative extension of $\mathcal{T}$.

If Φ is the set of all $\mathcal{L}^2_A$-formulas, then every arithmetical function (that is, every function whose graph is represented by an $\mathcal{L}^2_A$-formula) is Φ-definable in $\mathbf{V}^0$. To see this, suppose that $F(\vec{x}, \vec{X})$ has defining axiom (5.16). 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 (5.15) 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 $\mathbf{\Sigma}_1$, and we defined the provably total functions of $\mathcal{T}$ to be the $\mathbf{\Sigma}_1$-definable functions in $\mathcal{T}$. Here our choice for Φ is $\mathbf{\Sigma}^1_1$ (recall (Definition 4.14) that a $\mathbf{\Sigma}^1_1$ formula is a formula of the form $\exists\vec{X}\varphi$, where φ is a $\mathbf{\Sigma}^B_0$ 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 $\mathbf{\Sigma}^1_1$-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 $\mathbf{\Sigma}_1^1$ 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 $\mathbf{\Sigma}_1^B$-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*

$$F(\vec{x},\vec{Y})(i) \leftrightarrow i < t(\vec{x},\vec{Y}) \wedge \varphi(i,\vec{x},\vec{Y}). \tag{5.17}$$

We say that the formula on the RHS of (5.17) *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 5.17.

Proposition 5.32. *A string function is $\mathbf{\Sigma}_0^B$-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 (5.17) for a string function F, where φ is a $\mathbf{\Sigma}_0^B$ formula. Show that there is a $\mathbf{\Sigma}_0^B$ 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 (5.17), 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 $\mathbf{\Sigma}_1^B$ bit definition. However, as we noted earlier, this function is not $\mathbf{\Sigma}_1^B$-definable in $\mathbf{V}^0$. The theory that results from adding this function together with its $\mathbf{\Sigma}_1^B$-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 ($\mathbf{\Sigma}_0^B$-Closure). *Let Φ be a set of formulas over a language $\mathcal{L}$ which extends $\mathcal{L}_A^2$. Then $\mathbf{\Sigma}_0^B(\Phi)$ is the closure of Φ under the operations $\neg, \wedge, \vee$ and bounded number quantification. That is, if φ and ψ are formulas in $\mathbf{\Sigma}_0^B(\Phi)$ and t is an $\mathcal{L}_A^2$-term not containing x, then the following formulas are also in $\mathbf{\Sigma}_0^B(\Phi)$: $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $\forall x \leq t\varphi$, and $\exists x \leq t\varphi$.*

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 $\Phi \supseteq \mathbf{\Sigma}_0^B$. 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 $\mathbf{\Sigma}_0^B(\Phi)$-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 (5.8), the graph G_F of F can be defined as follows:

$$G_F(\vec{x}, \vec{Y}, Z) \;\equiv\; |Z| \leq t \;\wedge\; \forall i < t\,(Z(i) \leftrightarrow \varphi(i, \vec{x}, \vec{Y})) \qquad (5.18)$$

Now since $\mathcal{T}$ proves the Φ-**COMP**, we have

$$\mathcal{T} \vdash \forall\vec{x}\forall\vec{Y}\exists Z\; G_F(\vec{x}, \vec{Y}, Z) \qquad (5.19)$$

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 $\mathbf{\Sigma}_0^B(\Phi)$, it follows that F is $\mathbf{\Sigma}_0^B(\Phi)$-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 $\mathbf{\Sigma}_0^B(\Phi)$ formula for its graph:

$$G_f(y,\vec{x},\vec{X}) \equiv \ \forall z < y \neg\varphi(z,\vec{x},\vec{X}) \ \wedge \ y < t \supset \varphi(y,\vec{x},\vec{X}) \tag{5.20}$$

(i.e., y is the least number $< t$ such that $\varphi(y)$ holds, or t if no such y exists). $\square$

In this lemma, if we take $\mathcal{T} = \mathbf{V}^0$ and $\Phi = \mathbf{\Sigma}_0^B$, then (since $\mathbf{\Sigma}_0^B(\mathbf{\Sigma}_0^B) = \mathbf{\Sigma}_0^B$) we can apply Corollary 5.17 and Proposition 5.32 to obtain the following:

Corollary 5.36. *Every function in* $\mathbf{FAC}^0$ *is* $\mathbf{\Sigma}_0^B$*-definable in* $\mathbf{V}^0$.

This result can be generalized, using the following definition.

Definition 5.37. [1] *A string function is* $\mathbf{\Sigma}_0^B$-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* $\mathbf{\Sigma}_0^B(\mathcal{L})$ *formula. Similarly, a number function is* $\mathbf{\Sigma}_0^B$-definable *from* $\mathcal{L}$ *if it is p-bounded and its graph is represented by a* $\mathbf{\Sigma}_0^B(\mathcal{L})$ *formula.*

This "semantic" notion of $\mathbf{\Sigma}_0^B$-definability should not be confused with $\mathbf{\Sigma}_0^B$-definability in a theory (Definition 5.27), 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* $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP** *axiom scheme. Then a function which is* $\mathbf{\Sigma}_0^B$*-definable from* $\mathcal{L}$ *is* $\mathbf{\Sigma}_0^B(\mathcal{L})$*-definable in* $\mathcal{T}$.

Later we will prove the Witnessing Theorem for $\mathbf{V}^0$, which says that any $\mathbf{\Sigma}_1^1$-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* $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP***, where* $\mathcal{L}$ *is the vocabulary of* $\mathcal{T}$*. Then the theory resulting from* $\mathcal{T}$ *by adding the* $\mathbf{\Sigma}_0^B(\mathcal{L})$*-defining axioms or the* $\mathbf{\Sigma}_0^B(\mathcal{L})$*-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 $\mathbf{\Sigma}_0^B$ defining axioms and bit-defining axioms, the resulting theory is not only conservative over $\mathbf{V}^0$, it also proves the $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP** and $\mathbf{\Sigma}_0^B(\mathcal{L})$-**IND** axioms, where $\mathcal{L}$ is the resulting vocabulary.

Lemma 5.40 ($\mathbf{\Sigma}_0^B$-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 number function* f *in* $\mathcal{L}$*,* $\mathcal{T}$ *proves a* $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ *defining axiom for* f*, and for every string function* F *in* $\mathcal{L}$*,* $\mathcal{T}$ *proves a* $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ *bit-defining axiom for* F*. Then for every* $\mathbf{\Sigma}_0^B(\mathcal{L})$ *formula* φ^+ *there is a* $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ *formula* φ *such that*

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

[1]This notion is important for our definition of $\mathbf{AC}^0$ reduction, Definition 9.1.

Proof. 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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 ϵ, 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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

$$\varphi^+ \leftrightarrow \exists \vec{z} < \vec{r}(\vec{t}, \vec{T}),\; \rho(\vec{z}) \wedge \bigwedge_i \eta_i(z_i, \vec{t}, \vec{T})$$

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 5.16, and the $\mathbf{\Sigma}_0^B$ Representation Theorem 4.17.

Corollary 5.41 ($\mathbf{FAC}^0$ Closed under $\mathbf{\Sigma}_0^B$-Definability). *Every function $\mathbf{\Sigma}_0^B$-definable from a collection of $\mathbf{FAC}^0$ functions is in $\mathbf{FAC}^0$.*

Below we give $\mathbf{\Sigma}_0^B$-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 $\mathbf{\Sigma}_0^B$-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

$$S(X)(i) \leftrightarrow i \leq |X| \wedge [(X(i) \wedge \exists j < i \neg X(j)) \vee (\neg X(i) \wedge \forall j < i X(j))]$$

Recall from (5.7) that binary addition $X+Y$ has the following bit-defining axiom:

$$(X+Y)(i) \leftrightarrow i < |X| + |Y| \ \wedge \ [X(i) \oplus Y(i) \oplus Carry(i, X, Y)]$$

where $\oplus$ is exclusive OR, and

$$Carry(i, X, Y) \equiv \exists k < i, (X(k) \wedge Y(k)) \wedge \forall j < i[k < j \supset (X(j) \vee Y(j))]$$

Exercise 5.43. *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) $X + S(Y) = S(X + Y)$

c) $X + Y = Y + X$ *(Commutativity).*

d) $(X + Y) + Z = X + (Y + Z)$ *(Associativity).*
For Associativity, first show in $\mathbf{V}^0(+)$ that

$$Carry(i, Y, Z) \oplus Carry(i, X, Y + Z) \leftrightarrow Carry(i, X, Y) \oplus Carry(i, X + Y, Z).$$

Derive a stronger statement than this, and prove it by induction on i.

Example 5.44 (The Pairing Function). *We define the pairing function $\langle x, y\rangle$ as the following term of $\mathbf{I\Delta}_0$:*

$$\langle x, y\rangle =_{\text{def}} (x + y)(x + y + 1) + 2y \tag{5.21}$$

Exercise 5.45. *Show using results in Section 3.1 that $\mathbf{I\Delta}_0$ proves $\langle x, y\rangle$ is a one-one function. That is*

$$\mathbf{I\Delta}_0 \vdash \ \langle x_1, y_1\rangle = \langle x_2, y_2\rangle \supset x_1 = x_2 \wedge y_1 = y_2 \tag{5.22}$$

(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

$$\langle x_1, \ldots, x_{k+1}\rangle = \langle\langle x_1, \ldots, x_k\rangle, x_{k+1}\rangle$$

We will refer to the term $\langle x_1, \ldots, x_{k+1}\rangle$ 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

$$Z(x_1, \ldots, x_k) =_{\text{def}} Z(\langle x_1, \ldots, x_k\rangle) \tag{5.23}$$

Example 5.46 (The Projection Functions). *Consider the (partial) projection functions:*

$$y = \mathit{left}(x) \leftrightarrow \exists z \le x\,(x = \langle y, z\rangle) \qquad z = \mathit{right}(x) \leftrightarrow \exists y \le x\,(x = \langle y, z\rangle)$$

To make these functions total, we define

$$\mathit{left}(x) = \mathit{right}(x) = 0 \qquad \text{if } \neg \mathit{Pair}(x)$$

where

$$\mathit{Pair}(x) \equiv \exists y \le x \exists z \le x\,(x = \langle y, z\rangle)$$

For constants n and $k \le n$, if x codes an n-tuple, then the k-th component $\langle x\rangle_k^n$ of x can be extracted using left and right, e.g.,

$$\langle x\rangle_2^3 = \mathit{right}(\mathit{left}(x))$$

Exercise 5.47. *Let $\mathbf{I\Delta}_0$(left,right) be the conservative extension of $\mathbf{I\Delta}_0$ resulting by adding the $\mathbf{\Delta}_0$ defining axioms for left and right. Show that $\mathbf{I\Delta}_0$(left,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 $\mathbf{\Sigma}_0^B$-comprehension axiom scheme to multiple dimensions.

Definition 5.48 (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*

$$\exists X \le \langle y_1, \ldots, y_k\rangle \forall z_1 < y_1 \ldots \forall z_k < y_k (X(z_1, \ldots, z_k) \leftrightarrow \varphi(z_1, \ldots, z_k)) \quad (5.24)$$

where $k \ge 2$ and $\varphi(z)$ is any formula in Φ which may contain other free variables, but not X.

Lemma 5.49 (Multiple Comprehension). *Suppose that $\mathcal{T} \supseteq \mathbf{V}^0$ is a theory with vocabulary $\mathcal{L}$ which proves the $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP** axioms. Then $\mathcal{T}$ proves the $\mathbf{\Sigma}_0^B(\mathcal{L})$-**MULTICOMP** axioms.*

Proof. For the case $\mathcal{L} = \mathcal{L}_A^2$ we could work in the conservative extension $\mathcal{T}$(*left*,*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

$$\psi(z) \equiv \exists z_1 \le z \exists z_2 \le z,\ z = \langle z_1, z_2\rangle \wedge \varphi(z_1, z_2)$$

Now by $\mathbf{\Sigma}_0^B$-**COMP**,

$$\mathcal{T} \vdash \exists X \le \langle y_1, y_2\rangle \forall z < \langle y_1, y_2\rangle,\ X(z) \leftrightarrow \psi(z)$$

By Exercise 5.45, $\mathcal{T}$ proves that such X satisfies (5.24). $\square$

Now we introduce the string function $Row(x, Z)$ (or $Z^{[x]}$) in $\mathbf{FAC}^0$ to represent row x of the binary array Z.

Definition 5.50 (*Row* **and** $\mathbf{V}^0(Row)$**).** *The function* $Row(x, Z)$ *(also denoted* $Z^{[x]}$*) has the bit-defining axiom*

$$Row(x, Z)(i) \leftrightarrow i < |Z| \wedge Z(x, i) \qquad (5.25)$$

$\mathbf{V}^0(Row)$ *is the extension of* $\mathbf{V}^0$ *obtained from* $\mathbf{V}^0$ *by adding to it the string function Row and its* $\mathbf{\Sigma}_0^B$*-bit-definition* (5.25)*.*

Note that by Corollary 5.39, $\mathbf{V}^0(Row)$ is a conservative extension of $\mathbf{V}^0$.

The next result follows immediately from Lemma 5.40.

Lemma 5.51 (*Row* **Elimination Lemma).** *For every* $\mathbf{\Sigma}_0^B(Row)$ *formula* φ*, there is* $\mathbf{\Sigma}_0^B$ *formula* φ' *such that* $\mathbf{V}^0(Row) \vdash \varphi \leftrightarrow \varphi'$*. Hence* $\mathbf{V}^0(Row)$ *proves the* $\mathbf{\Sigma}_0^B(Row)$*-***COMP** *axiom scheme.*

We can use *Row* to represent a tuple $X_1, ..., X_k$ of strings by a single string Z, where $X_i = Z^{[i]}$. The following result follows immediately from the Multiple Comprehension Lemma.

Lemma 5.52. $\mathbf{V}^0(Row)$ *proves*

$$\forall X_1 ... \forall X_k \exists Z \leq t(X_1 = Z^{[1]} \wedge ... \wedge X_k = Z^{[k]}) \qquad (5.26)$$

where $t = \langle k, |X_1| + ... + |X_k| \rangle$*.* □

Definition 5.53. *A single-*$\mathbf{\Sigma}_1^B(\mathcal{L})$ *formula is one of the form* $\exists X \leq t\varphi$*, where* φ *is* $\mathbf{\Sigma}_0^B(\mathcal{L})$*.*

Exercise 5.54. *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* $\mathbf{\Sigma}_1^B(\mathcal{L})$ *formula* φ *there is a single-*$\mathbf{\Sigma}_1^B(\mathcal{L})$ *formula* φ' *such that* $\mathcal{T} \vdash \varphi \leftrightarrow \varphi'$*.*

Now use Lemma 5.51 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 5.55 (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 $\mathbf{\Sigma}_0^B$-definable in $\mathbf{V}^0$. As in the case of *Row*, it follows from Lemma 5.40 that any $\mathbf{\Sigma}_0^B(seq)$ formula is provably equivalent in $\mathbf{V}^0(seq)$ to a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formula. (See also the $\mathbf{AC}^0$ Elimination Lemma 5.73 for a more general result.)

5.4.1 $\mathbf{\Delta}_1^1$-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 "$\mathbf{\Delta}_1^1(\mathcal{L})$-definable predicate symbol" and a "$\mathbf{\Delta}_1^B(\mathcal{L})$-definable predicate symbol". Note that here $\mathbf{\Delta}_1^1(\mathcal{L})$ and $\mathbf{\Delta}_1^B(\mathcal{L})$ depend on the theory $\mathcal{T}$, in contrast to Definition 3.27.

Definition 5.56 ($\mathbf{\Delta}_1^1(\mathcal{L})$ and $\mathbf{\Delta}_1^B(\mathcal{L})$ 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 $\mathbf{\Delta}_1^1(\mathcal{L})$-definable (or simply $\mathbf{\Delta}_1^1$-definable) in $\mathcal{T}$ if there are $\mathbf{\Sigma}_1^1(\mathcal{L})$ formulas $\varphi(\vec{x},\vec{Y})$ and $\psi(\vec{x},\vec{Y})$ such that*

$$R(\vec{x},\vec{Y}) \leftrightarrow \varphi(\vec{x},\vec{Y}), \qquad \textit{and} \qquad \mathcal{T} \vdash \varphi(\vec{x},\vec{Y}) \leftrightarrow \neg\psi(\vec{x},\vec{Y}). \tag{5.27}$$

We say that P is $\mathbf{\Delta}_1^B(\mathcal{L})$-definable (or simply $\mathbf{\Delta}_1^B$-definable) in $\mathcal{T}$ if the formulas φ and ψ above are $\mathbf{\Sigma}_1^B$ formulas.

The following exercise can be proved using Parikh's Theorem.

Exercise 5.57. *Show that if $\mathcal{T}$ is a polynomial-bounded theory, then a predicate is $\mathbf{\Delta}_1^1$-definable in $\mathcal{T}$ iff it is $\mathbf{\Delta}_1^B$-definable in $\mathcal{T}$.*

Definition 5.58 (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:*

$$f_R(\vec{x},\vec{X}) = \begin{cases} 1 & \textit{if } R(\vec{x},\vec{X}) \\ 0 & \textit{otherwise} \end{cases}$$

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 $\mathbf{\Delta}_1^1$ definable relations in $\mathbf{V}^0$. More generally we have the following theorem.

Theorem 5.59. *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 $\mathbf{\Sigma}_1^1$-definable functions in $\mathcal{T}$ coincides with $\mathbf{FC}$, then the class of $\mathbf{\Delta}_1^1$-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 $\mathbf{\Delta}_1^1$-definable in $\mathcal{T}$. Then there are $\mathbf{\Sigma}_0^B$ formulas φ and ψ such that

$$R(\vec{x},\vec{X}) \leftrightarrow \exists\vec{Y}\ \varphi(\vec{x},\vec{X},\vec{Y})$$

and

$$\mathcal{T} \vdash (\exists\vec{Y}\ \varphi(\vec{x},\vec{X},\vec{Y}) \leftrightarrow \neg\exists\vec{Y}\ \psi(\vec{x},\vec{X},\vec{Y})) \tag{5.28}$$

Thus the characteristic function $f_R(\vec{x},\vec{X})$ of R satisfies

$$y = f_R(\vec{x},\vec{X}) \leftrightarrow \theta(y,\vec{x},\vec{X}) \tag{5.29}$$

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 (5.28) and the uniqueness of y follows from the $\rightarrow$ direction of (5.28). Thus f_R is $\mathbf{\Sigma}_1^1$-definable in $\mathcal{T}$, so f_R is in **FC**, and therefore R is in **C**.

Conversely, suppose that $R(\vec{x}, \vec{X})$ is in **C**, so f_R is in **FC**. Then f_R is $\mathbf{\Sigma}_1^1$-definable in $\mathcal{T}$, so there is a $\mathbf{\Sigma}_1^1$ formula $\theta(y, \vec{x}, \vec{X})$ such that (5.29) 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 $\mathbf{\Sigma}_1^1$ formula, it follows that R is $\mathbf{\Delta}_1^1$-definable in $\mathcal{T}$. □

5.5 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 $\mathbf{\Sigma}_0^B$-definable in $\mathbf{V}^0$, and string functions in $\mathbf{FAC}^0$ are $\mathbf{\Sigma}_0^B$-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 5.60 (Witnessing Theorem for $\mathbf{V}^0$). *Suppose that $\varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$ is a $\mathbf{\Sigma}_0^B$ 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 5.61 ($\mathbf{\Sigma}_1^1$-Definability Theorem for $\mathbf{V}^0$). *A function is in $\mathbf{FAC}^0$ iff it is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{V}^0$ iff it is $\mathbf{\Sigma}_1^B$-definable in $\mathbf{V}^0$ iff it is $\mathbf{\Sigma}_0^B$-definable in $\mathbf{V}^0$.*

Corollary 5.62. *A relation is in $\mathbf{AC}^0$ iff it is $\mathbf{\Delta}_1^1$ definable in $\mathbf{V}^0$ iff it is $\mathbf{\Delta}_1^B$ definable in $\mathbf{V}^0$.*

It follows from the $\mathbf{\Sigma}_0^B$-Representation Theorem 4.17 that a relation is in $\mathbf{AC}^0$ iff its characteristic function is in $\mathbf{AC}^0$. Therefore Corollary 5.62 follows from the $\mathbf{\Sigma}_1^1$-Definability Theorem for $\mathbf{V}^0$ and Theorem 5.59. 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 $\mathbf{\Sigma}_0^B$ formula θ, it is obvious that they are $\mathbf{\Delta}_1^B$ (and hence $\mathbf{\Delta}_1^1$) definable in $\mathbf{V}^0$: In (5.27) simply let φ be θ, and ψ be $\neg\theta$.

On the other hand, suppose that R is a $\mathbf{\Delta}_1^1$-definable relation of $\mathbf{V}^0$. In other words, there are $\mathbf{\Sigma}_0^B$ formulas $\varphi(\vec{x},\vec{X},\vec{Y})$ and $\psi(\vec{x},\vec{X},\vec{Y})$ so that

$$\begin{aligned} & R(\vec{x},\vec{X}) \leftrightarrow \exists\vec{Y}\varphi(\vec{x},\vec{X},\vec{Y}) \\ \text{and}\quad & \mathbf{V}^0 \vdash \exists\vec{Y}\varphi(\vec{x},\vec{X},\vec{Y}) \leftrightarrow \neg\exists\vec{Y}\psi(\vec{x},\vec{X},\vec{Y}) \end{aligned} \tag{5.30}$$

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

$$\mathbf{V}^0(F_1,\ldots,F_k) \vdash \forall\vec{x}\forall\vec{X}(\varphi(\vec{x},\vec{X},\vec{F}(\vec{x},\vec{X})) \vee \psi(\vec{x},\vec{X},\vec{F}(\vec{x},\vec{X}))) \tag{5.31}$$

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 (5.30) and (5.31).

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 5.20 **a**, R is also an $\mathbf{AC}^0$ relation. □

5.5.1 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 5.63. *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 $\mathbf{\Sigma}^1_1$-definable in $\mathbf{V}^0$. In the next chapter we will show that *parity* is $\mathbf{\Sigma}^1_1$-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 $\mathbf{\Sigma}^B_0$ formula $\varphi_{parity}(X,Y)$:

$$\neg Y(0) \wedge \forall i < |X|(Y(i+1) \leftrightarrow (X(i) \oplus Y(i))) \tag{5.32}$$

where $\oplus$ is exclusive OR. Thus $\varphi_{parity}(X,Y)$ asserts that for $0 \le i < |X|$, bit $Y(i+1)$ is 1 iff the number of 1's among bits $X(0),...,X(i)$ is odd. Define

$$\varphi(X) \equiv \exists Y \le (|X|+1)\ \varphi_{parity}(X,Y)$$

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 $\mathbf{V}^0$,

$$\mathbf{V}^0 \nvdash \forall X\exists Y \le (|X|+1)\ \varphi_{parity}(X,Y)$$

Note that this independence result does not follow from Parikh's Theorem.

5.5.2 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 $\mathbf{\Sigma}_1$ theorem of $\mathbf{I\Delta}_0$ can be "witnessed" by a single-sorted **LTH** function) which is proved in Theorem 3.62. There we use the Bounded Definability Theorem 3.33 (which follows from Parikh's Theorem) to show that the graph of any $\mathbf{\Sigma}_1$-definable function of $\mathbf{I\Delta}_0$ is actually definable by a $\mathbf{\Delta}_0$ 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 $\mathbf{\Sigma}^1_1$-definable function of $\mathbf{V}^0$ is representable by a $\mathbf{\Sigma}^B_1$ formula. However this does not suffice, since there are string functions whose graphs are in $\mathbf{AC}^0$ (i.e., representable by $\mathbf{\Sigma}^B_0$ formulas), but which do not belong to $\mathbf{FAC}^0$. An example is the counting function whose graph is given by the $\mathbf{\Sigma}^B_0$ formula $\delta_{NUM}(x,X,Y)$ (9.2).

Our first proof is by the Anchored $\mathbf{LK}^2$ Completeness Theorem 4.29. 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 5.6.1) 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 5.64. *Suppose that* $\varphi(\vec{x},\vec{X},Y)$ *is a* $\mathbf{\Sigma}^B_0$ *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 5.60 from Lemma 5.64. 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 5.52. 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 $\mathbf{\Sigma}_0^B$ formula $\varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$. Then since $\mathbf{V}^0(\mathit{Row})$ extends $\mathbf{V}^0$, we have also

$$\mathbf{V}^0(\mathit{Row}) \vdash \forall \vec{x} \forall \vec{X} \exists \vec{y} \exists \vec{Y}\ \varphi(\vec{x}, \vec{y}, \vec{X}, \vec{Y})$$

Note that

$$\mathbf{V}^0(\mathit{Row}) \vdash \forall y_1 \dots \forall y_k \forall Y_1 ... \forall Y_m \exists Z (\bigwedge_{1 \le i \le k} |Z^{[i]}| = y_i \wedge \bigwedge_{1 \le j \le m} Z^{[k+j]} = Y_j)$$

(See also Lemma 5.52.) Thus

$$\mathbf{V}^0(\mathit{Row}) \vdash \forall \vec{x} \forall \vec{X} \exists Z\ \varphi(\vec{x}, |Z^{[1]}|, \dots, |Z^{[k]}|, \vec{X}, Z^{[k+1]}, \dots, Z^{[k+m]})$$

i.e.,

$$\mathbf{V}^0(\mathit{Row}) \vdash \forall \vec{x} \forall \vec{X} \exists Z \psi(\vec{x}, \vec{X}, Z)$$

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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2 \cup \{\mathit{Row}\})$ formula.

Now by Lemma 5.51, there is a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formula $\psi'(\vec{x}, \vec{X}, Z)$ so that

$$\mathbf{V}^0(\mathit{Row}) \vdash \forall \vec{x} \forall \vec{X} \forall Z (\psi(\vec{x}, \vec{X}, Z) \leftrightarrow \psi'(\vec{x}, \vec{X}, Z))$$

As a result, since $\mathbf{V}^0(\mathit{Row})$ 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 5.64, 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

$$\mathbf{V}^0(\mathit{Row}, F) \vdash \forall \vec{x} \forall \vec{X} \psi(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$$

i.e.,

$$\mathbf{V}^0(\mathit{Row}, F) \vdash \forall \vec{x} \forall \vec{X}\ \varphi(\vec{x}, |F^{[1]}|, \dots, |F^{[k]}|, \vec{X}, F^{[k+1]}, \dots, F^{[k+m]})$$

where we write F for $F(\vec{x}, \vec{X})$.

Let $f_i(\vec{x}, \vec{X}) = |(F(\vec{x}, \vec{X}))^{[i]}|$ for $1 \leq i \leq k$ and $F_j(\vec{x}, \vec{X}) = (F(\vec{x}, \vec{X}))^{[k+j]}$ for $1 \leq j \leq m$ and denote $\{f_1, \ldots, f_k, F_1, \ldots, F_m\}$ 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,

$$\mathbf{V}^0(\mathcal{L}) \vdash \forall\vec{x}\forall\vec{X}\ \varphi(\vec{x}, \vec{f}, \vec{X}, \vec{F})$$

$\square$

The rest of this section is devoted to the proof of Lemma 5.64.

Proof of Lemma 5.64. The proof method is similar to that of Lemma 5.26 (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 4.4.1).

Note that all instances of the $\mathbf{\Sigma}_0^B$-**COMP** axioms (5.1) are $\mathbf{\Sigma}_1^1$ formulas (they are in fact $\mathbf{\Sigma}_1^B$ formulas). Since the endsequent of π is also a $\mathbf{\Sigma}_1^1$ formula, by the Subformula Property (Theorem 4.30), all formulas in π are $\mathbf{\Sigma}_1^1$ formulas, and in fact they contain at most one string quantifier $\exists X$ in front. In particular, every sequent in π has the form

$$\exists X_1\theta_1(X_1), \ldots, \exists X_m\theta_m(X_m), \Gamma \longrightarrow \Delta, \exists Y_1\psi_1(Y_1), \ldots, \exists Y_n\psi_n(Y_n) \qquad (5.33)$$

for $m, n \geq 0$, where θ_i and ψ_j and all formulas in Γ and Δ are $\mathbf{\Sigma}_0^B$.

We will prove by induction on the depth in π of a sequent $\mathcal{S}$ of the form (5.33) that there are $\mathbf{\Sigma}_0^B$-bit-definable string functions $F_1, ..., F_n$ (i.e., the witnessing functions) such that there is a collection of $\mathbf{\Sigma}_0^B$-bit-definable functions $\mathcal{L}$ including $F_1, ..., F_n$ and an $\mathbf{LK}^2$-$\mathbf{V}^0(\mathcal{L})$ proof of

$$\mathcal{S}' =_{\text{def}}\ \theta_1(\beta_1), \ldots, \theta_m(\beta_m), \Gamma \longrightarrow \Delta, \psi_1(F_1), \ldots, \psi_n(F_n) \qquad (5.34)$$

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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-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 5.26).

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 2.17) 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 (5.34) in the form

$$\mathcal{S}' = \quad A_1, ..., A_k \longrightarrow B_1, ..., B_\ell$$

then we assert

$$\mathbf{V}^0(\mathcal{L}) \vdash \ \forall\vec{x}\forall\vec{X}\forall\vec{Y}[(A_1 \wedge ... \wedge A_k) \supset (B_1 \vee ... \vee B_\ell)] \qquad (5.35)$$

Case I: S is an axiom of $\mathbf{V}^0$. If the axiom only involves $\mathbf{\Sigma}_0^B$ formulas, then no witnessing functions are needed. Otherwise $\mathcal{S}$ comes from a $\mathbf{\Sigma}_0^B$-**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 ∃-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 (5.34)). 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 ∃-right** or **number ∀-left**. No new witnessing functions are required.

Case V: $\mathcal{S}$ follows from an application of rule **number ∃-left** or **number ∀-right**. We consider **number ∃-left**, since **number ∀-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 $\mathbf{\Sigma}_0^B$. For $i = 1, 2$, let $F_1^i(\vec{a}, \vec{\alpha}), \ldots, F_n^i(\vec{a}, \vec{\alpha})$ be the witnessing functions for Π in $\mathcal{S}_i'$. Then we define witnessing functions $F_1, \ldots, F_n$ 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 $\mathbf{\Sigma}_0^B$, so ψ has the form

$$\psi \equiv \exists X \theta(X) \tag{5.36}$$

where $\theta(X)$ is $\mathbf{\Sigma}_0^B$. 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 5.65. *Show correctness of this definition of F in the special case where the cut formula ψ has the form* (5.36), *and Π has only one $\mathbf{\Sigma}_1^1$ formula, by arguing that $\mathbf{V}^0(\mathcal{L})$ can prove the semantic translation (5.35) 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 (5.33),

$$\begin{aligned} &\Lambda =_{\text{def}} \exists X_1\theta_1(X_1),\dots,\exists X_m\theta_m(X_m),\Gamma \\ \text{and}\quad &\Pi =_{\text{def}} \Delta, \exists Y_1\psi_1(Y_1),\dots,\exists Y_n\psi_n(Y_n)\end{aligned}$$

for $m,n \geq 0$, where θ_i and ψ_j and all formulas in Γ and Δ are $\mathbf{\Sigma}_0^B$. Also, A and B are $\mathbf{\Sigma}_0^B$ 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 $\vec{a},\vec{\alpha}$):

$$F_j(\dots)(z) \;\leftrightarrow\; (\psi_j(F_j(\dots)) \wedge F_j^1(\dots)(z)) \vee (\neg\psi_j(F_j(\dots)) \wedge F_j^2(\dots)(z))$$

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 5.66. *Show that in the* **Cases V**, **VI**, *and* **VII** *above, the new functions introduced have $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-bit-definitions.*

5.6 $\overline{\mathbf{V}}^0$: 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 3.3.2.

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 4.32 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 $\mathbf{\Sigma}_0^B$-**COMP** and **B12**, **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 $F_{\varphi,t}(\vec{x}, \vec{X})$ be the string function with bit definition

$$F_{\varphi,t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t(\vec{x}, \vec{X}) \wedge \varphi(z, \vec{x}, \vec{X}) \tag{5.37}$$

Also, let $f_{\varphi,t}(\vec{x}, \vec{X})$ be the number function defined as in (3.19) to be the least $y < t$ such that $\varphi(y, \vec{x}, \vec{X})$ holds, or t if no such y exists. Then $f_{\varphi,t}$ has defining axiom (we write f for $f_{\varphi,t}$, t for $t(\vec{x}, \vec{X})$, and $\ldots$ for $\vec{x}, \vec{X}$):

$$f(\ldots) \leq t \;\wedge\; [f(\ldots) < t \supset \varphi(f(\ldots), \ldots)] \;\wedge\; [v < f(\ldots) \supset \neg\varphi(v, \ldots)] \tag{5.38}$$

Recall that the predecessor function pd has the defining axioms:

$$\textbf{B12}'.\ pd(0) = 0 \qquad\qquad \textbf{B12}''.\ x \neq 0 \supset pd(x) + 1 = x \tag{5.39}$$

(**B12**$'$ and **B12**$''$ are called respectively **D1**$'$ and **D2**$''$ in Section 3.3.2.)

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 (5.38), 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.

$$\begin{aligned} &f_{\mathbf{SE}}(X, Y) \leq |X| \wedge \\ &f_{\mathbf{SE}}(X, Y) < |X| \supset (X(f_{\mathbf{SE}}(X, Y)) \not\leftrightarrow Y(f_{\mathbf{SE}}(X, Y))) \wedge \\ &z < f_{\mathbf{SE}}(X, Y) \supset (X(z) \leftrightarrow Y(z)). \end{aligned} \tag{5.40}$$

Let **SE**$'$ be the following axiom

$$(|X| = |Y| \wedge f_{\mathbf{SE}}(X, Y) = |X|) \supset X = Y. \tag{5.41}$$

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 $\mathcal{L}_{\mathbf{\Delta}_0}$ (Definition 3.41).

Definition 5.67. $\mathcal{L}_{\mathbf{FAC}^0}$ *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 $\mathcal{L}^2_A$ there is a string function $F_{\varphi,t}$ and a number function $f_{\varphi,t}$ in $\mathcal{L}_{\mathbf{FAC}^0}$.*

Definition 5.68. *$\overline{\mathbf{V}}^0$ is the theory over $\mathcal{L}_{\mathbf{FAC}^0}$ with the following set of axioms:* **B1**-**B11**, **L1**, **L2** *(Figure 5.1),* **B12′** *and* **B12″** (5.39), (5.40), **SE′** (5.41)*, and* (5.37) *for each function $F_{\varphi,t}$ and* (5.38) *for each function $f_{\varphi,t}$ 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 $\mathbf{\Sigma}^B_0$-**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 5.69. *For every $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FAC}^0})$ formula φ there is an open $\mathcal{L}_{\mathbf{FAC}^0}$-formula φ^+ such that $\overline{\mathbf{V}}^0 \vdash \varphi \leftrightarrow \varphi^+$.*

Lemma 5.70. *$\overline{\mathbf{V}}^0$ proves the $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FAC}^0})$-**COMP**, $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FAC}^0})$-**IND**, and $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FAC}^0})$-**MIN** axiom schemes.*

Proof. For comprehension, we need to show, for each $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FAC}^0})$ 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 (5.37). For induction and minimization we use Corollary 5.8. □

Theorem 5.71. *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 **B12**, **SE** and $\mathbf{\Sigma}^B_0$-**COMP**. First, **B12** follows from **B12″**. 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 (5.40) we have $f_{\mathbf{SE}}(X, Y) = |X|$. Hence by (5.41) we obtain $X = Y$.

That $\overline{\mathbf{V}}^0$ proves $\mathbf{\Sigma}^B_0$-**COMP** follows from Lemma 5.70.

Now we show that $\overline{\mathbf{V}}^0$ is conservative over $\mathbf{V}^0$. Let

$$pd, f_{\mathbf{SE}}, \ldots \tag{5.42}$$

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 $\mathcal{L}^2_A$ 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 $\mathcal{L}_n$ $(n \geq 0)$. Then

$$\overline{\mathbf{V}}^0 = \bigcup_{n \geq 0} \mathbf{V}^0(\mathcal{L}_n)$$

First we prove:

Claim For $n \geq 1$, $\mathbf{V}^0(\mathcal{L}_n)$ satisfies the hypothesis of Lemma 5.40.

From Lemma 5.40 and the claim we have $\mathbf{V}^0(\mathcal{L}_n)$ proves the $\mathbf{\Sigma}_0^B(\mathcal{L}_n)$-**COMP** axiom scheme. 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 $n \geq 1$. 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-bit-defining axiom in $\mathbf{V}^0(\mathcal{L}_n)$, and each number function in $\mathcal{L}_n$ has a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-defining axiom in $\mathbf{V}^0(\mathcal{L}_n)$, and thus establishing the claim.

For the base case, $n = 1$, by **B12′** and **B12″** *pd* has a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-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 (5.42) has a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-defining axiom or a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formula. Hence $\mathbf{V}^0(\mathcal{L}_{n+1})$ proves that f_{n+1}/F_{n+1} has a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-defining axiom, and each string function of $\mathcal{L}_{\mathbf{FAC}^0}$ has a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-bit-defining axiom.

Corollary 5.72. *The $\mathcal{L}_{\mathbf{FAC}^0}$ functions are precisely the functions of* $\mathbf{FAC}^0$.

Proof. By the above remark and the $\mathbf{\Sigma}_0^B$-Representation Theorem 4.17, the $\mathcal{L}_{\mathbf{FAC}^0}$ functions are in $\mathbf{FAC}^0$. The other inclusion follows from the $\mathbf{\Sigma}_0^B$-Representation Theorem 4.17 and Lemma 5.69. □

The next lemma follows from Lemma 5.40 and the claim in the above proof of Theorem 5.71. It generalizes the *Row* Elimination Lemma 5.51.

Lemma 5.73 ($\mathbf{FAC}^0$ Elimination Lemma). *Suppose that $\mathcal{L} \subseteq \mathcal{L}_{\mathbf{FAC}^0}$. Then for every $\mathbf{\Sigma}_0^B(\mathcal{L})$ formula φ, there is a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formula φ' so that $\mathbf{V}^0(\mathcal{L}) \vdash \varphi \leftrightarrow \varphi'$.*

5.6.1 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 5.60. For notational simplicity, we consider the case of a single existential string quantifier, and prove Lemma 5.64.

Suppose that $\varphi(\vec{x}, \vec{X}, Z)$ is a $\mathbf{\Sigma}_0^B$ formula such that

$$\mathbf{V}^0 \vdash \forall\vec{x}\forall\vec{X}\exists Z\ \varphi(\vec{x}, \vec{X}, Z)$$

By Lemma 5.69 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 4.32, there are terms $T_1(\vec{x}, \vec{X}), \ldots, T_n(\vec{x}, \vec{X})$ of $\overline{\mathbf{V}}^0$ such that

$$\overline{\mathbf{V}}^0 \vdash \forall\vec{x}\forall\vec{X}[\varphi'(\vec{x}, \vec{X}, T_1(\vec{x}, \vec{X})) \vee \ldots \vee \varphi'(\vec{x}, \vec{X}, T_n(\vec{x}, \vec{X}))]$$

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 (5.37), 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 5.71 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 $\mathbf{\Sigma}_0^B$ axioms to a theory does not increase the set of provably total functions in the theory. For example, let $\mathit{True}\mathbf{\Sigma}_0^B$ be the set of all $\mathbf{\Sigma}_0^B$ formulas which are true in the standard model $\underline{\mathbb{N}}_2$. Let $\mathbf{V}^0(\mathit{True}\mathbf{\Sigma}_0^B)$ be the result of adding $\mathit{True}\mathbf{\Sigma}_0^B$ as axioms to $\mathbf{V}^0$, and let $\overline{\mathbf{V}}^0(\mathit{True}\mathbf{\Sigma}_0^B)$ be the result of adding $\mathit{True}\mathbf{\Sigma}_0^B$ as axioms to $\overline{\mathbf{V}}^0$. Then $\overline{\mathbf{V}}^0(\mathit{True}\mathbf{\Sigma}_0^B)$ is a conservative extension of $\mathbf{V}^0(\mathit{True}\mathbf{\Sigma}_0^B)$, and the above proof goes through to show that the same class $\mathbf{FAC}^0$ of functions serve to witness the $\mathbf{\Sigma}_1^1$ theorems of $\mathbf{V}^0(\mathit{True}\mathbf{\Sigma}_0^B)$. Thus we have shown

Corollary 5.74. *The provably total functions in $\mathbf{V}^0(\mathit{True}\mathbf{\Sigma}_0^B)$ are precisely the functions in $\mathbf{FAC}^0$.*

5.7 Finite Axiomatizability

Theorem 5.75. $\mathbf{V}^0$ *is finitely axiomatizable.*

Proof. It suffices to show that all $\mathbf{\Sigma}_0^B$-**COMP** axioms follow from finitely many theorems of $\mathbf{V}^0$. Let 2-**BASIC**$^+$ (or simply B^+) denote the 2-**BASIC** axioms (Fig. 5.1) 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 semi-ring.

We show more generally that both $\mathbf{\Sigma}_0^B$-**COMP** and the multiple comprehension axioms (5.24) for all $\mathbf{\Sigma}_0^B$ 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 $\mathbf{\Sigma}_0^B$ 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

$$\exists Y \leq \langle b_1, ..., b_k\rangle \forall x_1 < b_1 ... \forall x_k < b_k (Y(\vec{x}) \leftrightarrow \varphi(\vec{x})) \tag{5.43}$$

We will show that $\mathbf{COMP}_\varphi$ for the following 12 formulas φ will suffice.

$\varphi_1(x_1, x_2)$	$\equiv$	$x_1 = x_2$
$\varphi_2(x_1, x_2, x_3)$	$\equiv$	$x_3 = x_1$
$\varphi_3(x_1, x_2, x_3)$	$\equiv$	$x_3 = x_2$
$\varphi_4[Q_1, Q_2](x_1, x_2)$	$\equiv$	$\exists y \leq x_1 (Q_1(x_1, y) \wedge Q_2(y, x_2))$
$\varphi_5[a](x, y)$	$\equiv$	$y = a$
$\varphi_6[Q_1, Q_2](x, y)$	$\equiv$	$\exists z_1 \leq y \exists z_2 \leq y (Q_1(x, z_1) \wedge Q_2(x, z_2) \wedge y = z_1 + z_2)$
$\varphi_7[Q_1, Q_2](x, y)$	$\equiv$	$\exists z_1 \leq y \exists z_2 \leq y (Q_1(x, z_1) \wedge Q_2(x, z_2) \wedge y = z_1 \cdot z_2)$
$\varphi_8[Q_1, Q_2, c](x)$	$\equiv$	$\exists y_1 \leq c \exists y_2 \leq c (Q_1(x, y_1) \wedge Q_2(x, y_2) \wedge y_1 \leq y_2)$
$\varphi_9[X, Q, c](x)$	$\equiv$	$\exists y \leq c (Q(x, y) \wedge X(y))$
$\varphi_{10}[Q](x)$	$\equiv$	$\neg Q(x)$
$\varphi_{11}[Q_1, Q_2](x)$	$\equiv$	$Q_1(x) \wedge Q_2(x)$
$\varphi_{12}[Q, c](x)$	$\equiv$	$\forall y \leq c Q(x, y)$

In the following lemmas, we abbreviate $\mathbf{COMP}_{\varphi_i}(...)$ by C_i.

Lemma 5.76. *For each* $k \geq 1$ *and* $1 \leq i \leq k$ *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 $1 \leq i < k$ use C_4 with Q_1 defined by C_2 and Q_2 defined by $\mathbf{COMP}_{\psi_{i,k-1}}$.
□

Lemma 5.77. *Let $\vec{x} = x_1, \cdots, x_k$, $k \geq 1$, 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 5.76. We then build monomials using C_7 repeatedly, and build the general case by repeated use of C_6. □

Lemma 5.78. *Let $t_1(\vec{x}), t_2(\vec{x})$ be terms with variables among $\vec{x}, \vec{a}, \vec{Q}$. Suppose*

$$\begin{aligned} \psi_1[\vec{a}, \vec{Q}](\vec{x}) &\equiv t_1(\vec{x}) \leq t_2(\vec{x}) \\ \psi_2[\vec{a}, \vec{Q}, X](\vec{x}) &\equiv X(t_1(\vec{x})) \end{aligned}$$

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 5.77 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 5.77 similarly to Q_1 above. □

Now we can complete the proof of the theorem. Lemma 5.78 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 $t_1(\vec{x}) \leq t_2(\vec{x}) \wedge t_2(\vec{x}) \leq t_1(\vec{x})$. 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 $\varphi(\vec{x}) \equiv \forall y \leq t(\vec{x}) \psi(\vec{x}, y)$. We assume as an induction hypothesis that we can define Q satisfying

$$\forall \vec{x} < \vec{b} \forall y < t(\vec{b}) + 1 [Q(\vec{x}, y) \leftrightarrow (y \leq t(\vec{x}) \supset \psi(\vec{x}, y))]$$

Then $\mathbf{COMP}_{\varphi}(\vec{b})$ follows from $\mathbf{COMP}_{\varphi_{12}}(Q, c, b)$ with $c \leftarrow t(\vec{b})$ and $b \leftarrow \langle b_1, ..., b_k \rangle$. □

5.8 Notes

The system $\mathbf{V}^0$ we introduce in this chapter is essentially $\mathbf{\Sigma}_0^p$-*comp* in [**?**], and $\mathbf{I\Sigma}_0^{1,b}$ (without #) in [**?**]. Zambella [**?**] 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 [**?**].

The set 2-**BASIC** is similar to the axioms for Zambella's theory Θ in [**?**], and forms the two-sorted analog of Buss's single-sorted axioms **BASIC** [**?**]. It is slightly different from that which are presented in [**?**] and [**?**].

The statement and proof of Theorem 5.60 (witnessing) are inspired by [**?**], 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 [**?**].

Theorem 5.75 (finite axiomatizability) is taken from Section 7 of [**?**].

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 $\mathbf{\Sigma}^1_1$-Definability Theorem for $\mathbf{V}^1$: A function is $\mathbf{\Sigma}^1_1$-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 viewed, to a large extent, as he 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 [**?**] has shown how to formalize lower bound proofs for Boolean complexity in $\mathbf{V}^1$.

In Chapter 8 we will introduce other theories for polynomial time, and compare them with $\mathbf{V}^1$.

6.1 Induction Schemes in $\mathbf{V}^i$

Recall (Definition 5.3) that $\mathbf{V}^i$ is axiomatized by 2-**BASIC** and $\mathbf{\Sigma}^B_i$-**COMP**, where $\mathbf{\Sigma}^B_i$-**COMP** consists of all formulas of the form

$$\exists X \leq y \forall z < y (X(z) \leftrightarrow \varphi(z)), \tag{6.1}$$

where $\varphi(z)$ is a $\mathbf{\Sigma}^B_i$ formula, and X does not occur free in $\varphi(z)$.

The next result follows from Corollary 5.8.

Corollary 6.1. *For $i \geq 0$, $\mathbf{V}^i$ proves the $\mathbf{\Sigma}^B_i$-***IND***, $\mathbf{\Sigma}^B_i$-***MIN***, and $\mathbf{\Sigma}^B_i$-***MAX*** *axiom schemes.*

It turns out that $\mathbf{V}^i$ proves these schemes for a wider class of formulas than just $\mathbf{\Sigma}^B_i$. To show this, we start with a partial generalization of the Multiple Comprehension Lemma 5.49.

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 $\mathbf{\Sigma}_i^B(\mathcal{L})$-***COMP** *axioms, then $\mathcal{T}$ proves the multiple comprehension axiom*

$$\exists X \leq \langle \vec{y} \rangle \forall \vec{z} < \vec{y}(X(\vec{z}) \leftrightarrow \varphi(\vec{z})) \tag{6.2}$$

*(see (5.24)) for any $k \geq 2$ and any $\varphi \in \mathbf{\Sigma}_i^B(\mathcal{L})$. In particular, for all $i \geq 0$, $\mathbf{V}^i$ proves $\mathbf{\Sigma}_i^B$-***MULTICOMP***.*

Proof. The method used to prove the earlier version, Lemma 5.49, does not work here, because for $i \geq 1$ the $\mathbf{\Sigma}_i^B(\mathcal{L})$-formulas are not closed under bounded number quantification.

For notational simplicity we prove the case $k = 2$. First we consider the case the $\mathcal{L}$ includes *left* and *right*. Assuming that $\varphi(z_1, z_2)$ is in $\mathbf{\Sigma}_i^B(\mathcal{L})$ and $\mathcal{T}$ proves the $\mathbf{\Sigma}_i^B(\mathcal{L})$-**COMP** axioms it follows that $\mathcal{T}$ proves

$$\exists X \leq \langle y_1, y_2 \rangle \forall z < \langle y_1, y_2 \rangle, X(z) \leftrightarrow \varphi(\mathit{left}(z), \mathit{right}(z))$$

and (6.2) follows.

For the case $\mathcal{L} = \mathcal{L}_A^2$, we work in the conservative extension $\mathcal{T}(\mathit{left},\mathit{right})$ of $\mathcal{T}$. Note that the conclusion of Lemma 5.40 applies to transform a $\mathbf{\Sigma}_i^B(\mathit{left},\mathit{right})$ formula φ^+ to an equivalent $\mathbf{\Sigma}_i^B$ formula φ, since a $\mathbf{\Sigma}_i^B$ formula is just a $\mathbf{\Sigma}_0^B$ formula with a prefix of string quantifiers. Therefore if $\mathcal{T}$ proves the $\mathbf{\Sigma}_i^B$-**COMP** axioms, it follows that $\mathcal{T}(\mathit{left},\mathit{right})$ proves the $\mathbf{\Sigma}_i^B(\mathit{left}, \mathit{right})$-**COMP** axioms.
$\square$

The next result refers to the $\mathbf{\Sigma}_0^B$-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 (6.2) for every $k \geq 1$ and every φ in some class Φ of $\mathcal{L}$-formulas. Then $\mathcal{T}$ proves the $\mathbf{\Sigma}_0^B(\Phi)$-***COMP** *axioms.*

The following result is an immediate consequence of this theorem, Lemma 6.2, and Corollary 5.8, since every $\mathbf{\Pi}_i^B$ formula is equivalent to a negated $\mathbf{\Sigma}_i^B$ formula.

Corollary 6.4. *For $i \geq 0$ let Φ_i be $\mathbf{\Sigma}_0^B(\mathbf{\Sigma}_i^B \cup \mathbf{\Pi}_i^B)$. 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 (6.2) for $\varphi \in \mathbf{\Sigma}_0^B(\Phi)$, 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 $\mathbf{\Sigma}_0^B$-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

$$\exists X \le \langle \vec{y} \rangle \forall \vec{z} < \vec{y}(X(\vec{z}) \leftrightarrow \neg Y(\vec{z}))$$

Thus $\mathcal{T}$ proves (6.2).

The cases $\wedge$ and $\vee$ are similar. Finally we consider the case that $\varphi(\vec{z})$ has the form $\forall x \le t\psi(x, \vec{z})$. By the induction hypothesis $\mathcal{T}$ proves

$$\exists Y \le \langle t+1, \vec{y} \rangle \forall x \le t \forall \vec{z} < \vec{y}(Y(x, \vec{z}) \leftrightarrow \psi(x, \vec{z}))$$

By Lemma 5.49 $\mathbf{V}^0$ proves

$$\exists X \le \langle \vec{y} \rangle \forall \vec{z} < \vec{y}(X(\vec{z}) \leftrightarrow \forall x \le tY(x, \vec{z}))$$

Now (6.2) follows from these two formulas. $\square$

6.2 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 5.16) **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 74). (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 ($\mathbf{\Sigma}_1^1$-Definability Theorem for $\mathbf{V}^1$). *A function is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{V}^1$ iff it is in* **FP**.

The "if" direction is proved in Section 6.2.1. 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 5.24). The following corollary follows from the $\mathbf{\Sigma}_1^1$-Definability Theorem for $\mathbf{V}^1$ above, and Parikh's Theorem (see Corollary 5.29).

Corollary 6.7. *A function is in* **FP** *iff it is $\mathbf{\Sigma}_1^B$-definable in $\mathbf{V}^1$.*

The next corollary follows from the results above and Theorem 5.59.

Corollary 6.8. *A relation is in* $\mathbf{P}$ *iff it is is* $\mathbf{\Delta}_1^1$*-definable in* $\mathbf{V}^1$ *iff it is* $\mathbf{\Delta}_1^B$*-definable in* $\mathbf{V}^1$*.*

Recall (Theorem 4.18) that the $\mathbf{\Sigma}_1^B$ formulas represent precisely the $\mathbf{NP}$ relations, and hence by Definition 5.56 a relation is $\mathbf{\Delta}_1^B$ 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 106) which are in $\mathbf{P}$ but not in $\mathbf{AC}^0$. □

Exercise 6.10 (*parity*(X) **in** $\mathbf{V}^1$**).** *Recall the formula* $\varphi_{parity}(X, Y)$ *((5.32) on page 107). Show that the function parity*(X)*, which is the characteristic function of PARITY (page 106), is* $\mathbf{\Sigma}_1^1$*-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* $X \otimes Y$ *where*

$$X \times Y = Z \leftrightarrow bin(Z) = bin(X) \cdot bin(Y)$$

(see (4.4) on page 76). Consider the the $\mathbf{\Sigma}_1^1$ *defining axiom for* $X \times Y$ *in* $\mathbf{V}^1$ *that is based on the "school" algorithm for multiplying two numbers in binary. First, we construct the* table $X \otimes Y$ *that has* $|Y|$ *rows and whose ith row is either 0, if* $Y(i) = 0$ *(i.e.,* $\neg Y(i)$*), or a copy of* X *shifted by* i *bits, if* $Y(i) = 1$*. Thus,* $X \otimes Y$ *can be defined by (see Definition 5.50 for row notation)*

$$|X \otimes Y| \le \langle |Y|, |X| + |Y| \rangle \wedge$$
$$\forall i < |Y| \forall z < i + |X|, (X \otimes Y^{[i]})(z) \leftrightarrow (Y(i) \wedge \exists u \le z\, (u + i = z \wedge X(u)))$$

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*

$$|W| \le 1 + \langle |Y|, |X| + |Y| \rangle \wedge |W^{[0]}| = 0 \wedge \forall i < |Y|, W^{[i+1]} = W^{[i]} + Z^{[i]}$$

(Hint: Use $\mathbf{\Sigma}_1^B$**-IND***. For the bound on* $|W|$*, show that* $|W^{[i]}| \le |X| + i$*.)*

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) $X \times S(Y) = (X \times Y) + X$*.*

It will follow from our discussion in Section 8.2 that $\mathbf{V}^1(\varnothing, S, +, \times)$ proves the *string induction axiom scheme* for $\mathbf{\Sigma}_0^B(\varnothing, S, +, \times)$ formulas (see Corollary 8.42 and Theorem 8.11). Consequently, $\mathbf{V}^1(\varnothing, S, +, \times)$ proves the properties of the string functions $\varnothing, S, +$ and $\times$ as listed in Example 3.8.

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 $\mathbf{\Sigma}_1^1$-defined in $\mathbf{V}^1$ by the following steps. Suppose that $Y \le X$, and let z be such that $z + |Y| = |X|$.*

a) *Give a $\mathbf{\Sigma}_0^B$-bit-definition for the table U, where the row $U^{[i]}$ of U is Y "shifted" by i bits, for $0 \le i \le z$.*

b) *Prove in $\mathbf{V}^1$ the existence and uniqueness of a table W such that*

$$W^{[z]} = X \wedge \forall i < z, ((W^{[i+1]} < U^{[i+1]} \supset W^{[i]} = W^{[i+1]}) \wedge \\ (U^{[i+1]} \le W^{[i+1]} \supset W^{[i]} + U^{[i+1]} = W^{[i+1]}))$$

c) *Define $X \div Y$ and $Rem(X, Y)$ using W.*

d) *Show in $\mathbf{V}^1(+, \times, \div, Rem)$ that*

$$X = (Y \times (X \div Y)) + Rem(X, Y)$$

6.2.1 The "if" Direction of Theorem 6.6

We will give two proofs of the fact that every polynomial time function is $\mathbf{\Sigma}_1^1$-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 5.50 for notation)

$$Z = \langle Z^{[0]}, Z^{[1]}, \ldots Z^{[m]} \rangle$$

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 $\mathbf{\Sigma}_0^B$-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 5.50). Because these functions are $\mathbf{\Sigma}_0^B$-definable in $\mathbf{V}^0$, it follows from the $\mathbf{FAC}^0$ Elimination Lemma 5.73 that any $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2 \cup \{Init, Next, Out, Row\})$ formula can be transformed into a provably equivalent $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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 $\mathbf{\Sigma}_0^B$ (resp. $\mathbf{\Sigma}_1^B$) formula below with the new functions is provably equivalent to a $\mathbf{\Sigma}_0^B$ (resp. $\mathbf{\Sigma}_1^B$) 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

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \exists Z \le \langle t, t\rangle(\varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z) \wedge Y = Out_{\mathsf{M}}(Z^{[t]})) \tag{6.3}$$

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 (6.3) is a defining axiom for F in $\mathbf{V}^1$, i.e.,

$$\mathbf{V}^1 \vdash \forall \vec{x} \forall \vec{X} \exists ! Y \exists Z \le \langle t, t\rangle(\varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z) \wedge Y = Out_{\mathsf{M}}(Z^{[t]}))$$

For the uniqueness of Y, it suffices to verify that if Z_1 and Z_2 are two strings satisfying

$$|Z_k| \le \langle t, t\rangle \wedge \varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z_k)$$

(for $k = 1, 2$), then for all z,

$$z \le t \supset Z_1^{[z]} = Z_2^{[z]} \tag{6.4}$$

This follows in $\mathbf{V}^1$ using $\mathbf{\Sigma}_0^B$-**IND** on the formula (6.4) with induction on z.

For the existence of Y, we need to show that $\mathbf{V}^1$ proves

$$\forall \vec{x} \forall \vec{X} \exists Z \le \langle t, t\rangle\ \varphi_{\mathsf{M}}(\vec{x}, \vec{X}, Z)$$

This formula can be proved in $\mathbf{V}^1$ by using number induction axiom (Corollary 6.1) on b for the $\mathbf{\Sigma}_1^B$ formula

$$\exists W \le \langle b, t\rangle,\ W^{[0]} = Init_{\mathsf{M}}(\vec{x}, \vec{X}) \wedge \forall z < b\, W^{[z+1]} = Next_{\mathsf{M}}(W^{[z]})$$

□

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 $\mathbf{\Sigma}_0^B$-bit-defining axiom

$$Cut(x, X)(z) \leftrightarrow z < x \wedge X(z) \tag{6.5}$$

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*

$$F(0, \vec{x}, \vec{X}) = G(\vec{x}, \vec{X}) \tag{6.6}$$

$$F(y+1, \vec{x}, \vec{X}) = (H(y, \vec{x}, \vec{X}, F(y, \vec{x}, \vec{X})))^{<t(y,\vec{x},\vec{X})} \tag{6.7}$$

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 **FP**, and that applying the operations composition and limited recursion to functions in **FP** results in functions in **FP**.

For the $\Longrightarrow$ direction, the function F computed by a polytime Turing machine M can be defined from the $\mathbf{AC}^0$ functions $Init_{\mathsf{M}}$, $Next_{\mathsf{M}}$ and Out_{M} by limited recursion and composition. In more detail, we can define a string function $Conf_{\mathsf{M}}(y, \vec{x}, \vec{X})$ to be the string coding the configuration of M on input $(\vec{x}, \vec{X})$ at time y. Then $Conf_{\mathsf{M}}$ satisfies the recursion

$$Conf_{\mathsf{M}}(0, \vec{x}, \vec{X}) = Init_M(\vec{x}, \vec{X})$$
$$Conf_{\mathsf{M}}(y+1, \vec{x}, \vec{X}) = Next_{\mathsf{M}}(Conf_{\mathsf{M}}(y, \vec{x}, \vec{X}))$$

To turn this recursion into one fitting Definition 6.15 we apply $Cut(t(y, \vec{x}, \vec{X}), \ldots)$ to the RHS of the second equation, for a suitable $\mathcal{L}_A^2$-term t bounding the run time of M. Then

$$F(\vec{x}, \vec{X}) = Out_{\mathsf{M}}(Conf_{\mathsf{M}}(t(\vec{x}, \vec{X}), \vec{x}, \vec{X}))$$

□

6.2.2 Application of Cobham's Theorem

Second proof of the $\Longleftarrow$ Direction of Theorem 6.6. We use Cobham's Characterization of **FP** to show that the polytime string functions are $\mathbf{\Sigma}_1^1$-definable in $\mathbf{V}^1$. It follows from Proposition 6.5 that the polytime number functions are also $\mathbf{\Sigma}_1^1$-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 $\mathbf{\Sigma}_1^1$-definable in $\mathbf{V}^0$ (Corollary 5.61), hence also in $\mathbf{V}^1$. For the induction step, we need to show that the $\mathbf{\Sigma}_1^1$-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 $\mathbf{\Sigma}_1^1$-definable functions in $\mathbf{V}^1$, and $F(y, \vec{x}, \vec{X})$ is defined by limited recursion from G and H as in (6.6) and (6.7) for some polynomial p. Then we can $\mathbf{\Sigma}_1^1$-define F by coding the sequence of values $F(0), F(1), \ldots, F(y)$ as the rows $W^{[0]}, W^{[1]}, \ldots, W^{[y]}$ of a single array W. Thus (omitting $\vec{x}, \vec{X}$):

$$\begin{aligned} Y = F(y) \leftrightarrow \exists W,\ & W^{[0]} = G() \wedge \\ & \forall z < y\ W^{[z+1]} = (H(z, W^{[z]}))^{<t(z)} \wedge \\ & Y = W^{[y]} \end{aligned}$$

The RHS is not immediately equivalent to a $\mathbf{\Sigma}_1^1$ formula when the equations involving G and H are replaced by $\mathbf{\Sigma}_1^1$ 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 $\mathbf{\Sigma}_1^1$-defining axiom for F from the RHS as follows:

By assumption, G and H have $\mathbf{\Sigma}_1^1$-defining axioms. Therefore there are $\mathbf{\Sigma}_0^B$ 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

$$\mathbf{V}^1 \vdash \exists! W \exists \vec{U} \varphi_G(\vec{U}, W) \tag{6.8}$$

$$\mathbf{V}^1 \vdash \forall y \forall Z \exists! W \exists \vec{V} \varphi_H(y, Z, \vec{\;} W) \tag{6.9}$$

The $\mathbf{\Sigma}_1^1$-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 (6.9) when evaluating $H(z, W^{[z]})$.

$$\begin{aligned} Y = F(y) \leftrightarrow & \exists W \exists \vec{U} \exists \vec{V},\ \varphi_G(\vec{U}, W^{[0]}) \wedge \\ & \forall z < y (\varphi_H(z, W^{[z]}, \vec{V}^{[z]}, (W^{[z+1]})^{<t(z)}) \wedge \\ & Y = W^{[y]} \end{aligned} \tag{6.10}$$

Since the terms such as $(W^{[z+1]})^{<t(z)}$ are easily seen to be $\mathbf{\Sigma}_0^B$-bit-definable, it follows from Lemma 5.73 that this defining axiom can be replaced by an equivalent $\mathbf{\Sigma}_1^1$-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 (6.10), 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 (6.8) and (6.9).

Now we show that $\mathbf{V}^1$proves the existence of Y satisfying the RHS of (6.10). We start by noting that all of the initial string quantifiers can be bounded. This follows from Parikh's Theorem, using (6.8) and (6.9). Let $\psi(y)$ be the $\mathbf{\Sigma}_1^B$-formula obtained from this bounded form of the RHS of (6.10), with the final conjunct $Y = W^{[y]}$ deleted. Thus $\psi(y)$ asserts the existence of an array

$$W = (W^{[0]}, W^{[1]}, \dots, W^{[y]})$$

whose rows are the successive values

$$F(0), F(1), \dots, F(y)$$

We show that $\mathbf{V}^1$ proves $\psi(y)$ by induction on y. The base case follows from (6.8): If W' satisfies the existential quantifier $\exists W$ in (6.8), 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 (6.9) 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 (6.10): just set $Y = W^{[y]}$. Hence $F(y)$ is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{V}^1$. $\square$

6.3 The Replacement Axiom Scheme

Recall that the classes $\mathbf{\Sigma}_i^B$ and $\mathbf{\Pi}_i^B$ 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*

$$\mathbf{g\Sigma}_0^B(\mathcal{L}) = \mathbf{g\Pi}_0^B(\mathcal{L}) = \mathbf{\Sigma}_0^B(\mathcal{L})$$

For $i \ge 0$, $\mathbf{g\Sigma}_{i+1}^B$ is the closure of $\mathbf{g\Pi}_i^B$ under $\wedge$, $\vee$, $\forall x \le t$, $\exists x \le t$ and $\exists X \le t$. Similarly, $\mathbf{g\Pi}_{i+1}^B$ is the closure of $\mathbf{g\Sigma}_i^B$ under $\wedge$, $\vee$, $\forall x \le t$, $\exists x \le t$ and $\forall X \le t$.

As usual, we will drop mention of $\mathcal{L}$ when it is clear from context. Notice that for $i \geq 0$, $\mathbf{\Sigma}_i^B \subset \mathbf{\Sigma}_0^B(\mathbf{\Sigma}_i^B) \subset \mathbf{g\Sigma}_i^B$, and $\mathbf{\Pi}_i^B \subset \mathbf{\Sigma}_0^B(\mathbf{\Pi}_i^B) \subset \mathbf{g\Pi}_i^B$. Also

$$\mathbf{\Sigma}_0^B \subset \mathbf{g\Sigma}_1^B \subset \mathbf{g\Sigma}_2^B \subset \dots \qquad \text{and} \qquad \mathbf{\Sigma}_0^B \subset \mathbf{g\Pi}_1^B \subset \mathbf{g\Pi}_2^B \subset \dots$$

For any formula φ^+ in $\mathbf{g\Sigma}_i^B$, there is a formula φ in $\mathbf{\Sigma}_i^B$ 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 $\mathbf{\Sigma}_0^B$ 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 $\mathcal{L} \cup \{Row\}$):*

$$\forall x \leq b \exists X \leq c\ \varphi(x, X)\ \supset\ \exists Z \leq \langle b, c\rangle\ \forall x \leq b,\ |Z^{[x]}| \leq c \wedge \varphi(x, Z^{[x]}) \qquad (6.11)$$

where φ is in Φ.

Note that in (6.11) the LHS is a logical consequence of the RHS. Also (6.11) 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 (6.11), but by the *Row* Elimination Lemma 5.51 (or more generally the $\mathbf{FAC}^0$ Elimination Lemma 5.73), any $\mathbf{\Sigma}_0^B(Row)$ formula is equivalent to a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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 (6.11) to be the equivalent $\mathcal{L}_A^2$ formula which is obtained by transforming every atomic sub-formula containing *Row* into a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formula.

Notation When we say that a theory $\mathcal{T}$ with vocabulary $\mathcal{L}$ that proves a **REPL** axiom scheme (e.g., $\mathbf{\Sigma}_0^B(\mathcal{L})$-**REPL**), then either $\mathcal{L}_A^2 \cup \{Row\} \subseteq \mathcal{L}$, or $\mathcal{L} = \mathcal{L}_A^2$ and (6.11) is as above.

Recall that a *single*-$\mathbf{\Sigma}_1^B$ formula has the form $\exists X \leq t\psi(X)$, where ψ is a $\mathbf{\Sigma}_0^B$ formula.

Lemma 6.19. *Suppose that $\mathcal{T}$ is a polynomial–bounded theory which proves the $\mathbf{\Sigma}_0^B(\mathcal{L})$-**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-$\mathbf{\Sigma}_1^B(\mathcal{L})$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L})$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L})$ formula. By the induction hypothesis, $\theta(x)$ is equivalent in $\mathcal{T}$ to a *single*-$\mathbf{\Sigma}_1^B(\mathcal{L})$ formula $\exists X \le t\psi(x, X)$, where ψ is a $\mathbf{\Sigma}_0^B(\mathcal{L})$ 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*-$\mathbf{\Sigma}_1^B(\mathcal{L})$ formula by $\mathbf{\Sigma}_0^B(\mathcal{L})$-**REPL**.

The other cases for the induction step follow easily with the help of exercise 5.54, 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 $\mathbf{\Pi}_i^B(\mathcal{L})$-**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 $\mathbf{\Sigma}_{i+1}^B(\mathcal{L})$ 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 $\mathbf{\Pi}_{i+1}^B(\mathcal{L})$ 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 $\mathbf{\Sigma}_{i+1}^B(\mathcal{L})$-**REPL** axiom scheme.*

The next lemma shows that $\mathbf{V}^1$ proves the $\mathbf{\Sigma}_1^B$-**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 8.5) that $\mathbf{V}^0$ *does not* prove the $\mathbf{\Sigma}_0^B$-**REPL** axiom scheme. 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 $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_{i+1}^B(\mathcal{L})$-**IND** axiom scheme, for some $i \ge 0$. Then $\mathcal{T}$ also proves the $\mathbf{\Pi}_i^B(\mathcal{L})$-**REPL** axiom scheme.*

Proof. Let φ be a $\mathbf{\Pi}_i^B(\mathcal{L})$ formula. We will show that $\mathcal{T}$ proves (6.11). Intuitively, the RHS of (6.11) 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 (6.11) is equivalent to a $\mathbf{\Sigma}^B_{i+1}(\mathcal{L})$ formula. First consider the case where $i = 0$, so φ is a $\mathbf{\Sigma}^B_0(\mathcal{L})$ formula. Let

$$\psi(z) \equiv \exists Z \le \langle z, c\rangle \forall x \le z\ (|Z^{[x]}| \le c \wedge \varphi(x, Z^{[x]}))$$

Then $\psi(z)$ is a $\mathbf{\Sigma}^B_1(\mathcal{L})$ formula and the RHS of (6.11) is just $\psi(b)$. Our task is to show in $\mathcal{T}$ that $\psi(z)$ holds for $z \le b$, assuming the LHS of (6.11). This is proved in $\mathcal{T}$ by induction on $z \le b$. For the base case, $\psi(0)$ follows from the LHS of (6.11) by putting $x = 0$. The induction step follows from the induction hypothesis and the LHS of (6.11), using $\mathbf{\Sigma}^B_0$-**COMP**.

For the case where $i \ge 1$, note that when φ is a $\mathbf{\Pi}^B_i(\mathcal{L})$ formula, the RHS of (6.11) is not really a $\mathbf{\Sigma}^B_{i+1}(\mathcal{L})$ formula. But it is equivalent (in $\mathcal{T}$) to:

$$\exists Z \le \langle b, c\rangle \forall Y \le b\ (|Z^{[|Y|]}| \le c \wedge \varphi(x, Z^{[|Y|]}))$$

which is equivalent to a $\mathbf{\Sigma}^B_{i+1}(\mathcal{L})$ formula. Let ψ be the equivalent $\mathbf{\Sigma}^B_{i+1}(\mathcal{L})$ 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 $i \ge 1$, the theory $\mathbf{V}^i$ proves the $\mathbf{g\Sigma}^B_i$-**REPL** axiom scheme. For each $\mathbf{g\Sigma}^B_1$ formula φ, there is a single-$\mathbf{\Sigma}^B_1$ formula φ' such that $\mathbf{V}^1 \vdash \varphi \leftrightarrow \varphi'$. Also $\mathbf{V}^1$ proves $\mathbf{\Sigma}^B_0(\mathbf{g\Sigma}^B_1 \cup \mathbf{g\Pi}^B_1)$-**IND**.*

6.3.1 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 $\mathbf{\Sigma}^1_1$-definable functions (i.e., polytime functions), $\mathbf{\Delta}^1_1$-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 $\mathbf{\Sigma}^B_1(\mathcal{L})$-**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 $\mathbf{\Sigma}^B_0(\mathcal{L})$-**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 $\mathbf{\Sigma}^B_0$-bit-definable in $\mathbf{V}^0$.

Here it suffices to show that any $\mathbf{\Sigma}^B_1(\mathcal{L})$ formula is provably equivalent in $\mathbf{V}^1(\mathcal{L})$ to a $\mathbf{\Sigma}^B_1(\mathcal{L}^2_A)$ formula. We will prove this by structural induction on the $\mathbf{\Sigma}^B_1(\mathcal{L})$ formula. For the induction step, we use Corollary 6.24 above. More generally, we prove:

Lemma 6.25 ($\mathbf{\Sigma}^B_1$-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 $\mathbf{\Sigma}^B_0(\mathcal{L})$-**REPL**. Let $\mathcal{T}'$ be the extension of $\mathcal{T}$ which is obtained by adding to $\mathcal{T}$ a $\mathbf{\Sigma}^1_1(\mathcal{L})$-definable function or a $\mathbf{\Delta}^1_1(\mathcal{L})$-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 $\mathbf{\Sigma}_1^B(\mathcal{L}')$ formula φ^+, there is a $\mathbf{\Sigma}_1^B(\mathcal{L})$ formula φ so that $\mathcal{T}' \vdash \varphi^+ \leftrightarrow \varphi$;*
c) *For any $\mathbf{\Sigma}_0^B(\mathcal{L}')$ formula φ^+, there are a $\mathbf{\Sigma}_1^B(\mathcal{L})$ formula φ_1 and a $\mathbf{\Pi}_1^B(\mathcal{L})$ formula φ_2 so that $\mathcal{T}' \vdash \varphi^+ \leftrightarrow \varphi_1$, and $\mathcal{T} \vdash \varphi_1 \leftrightarrow \varphi_2$;*
d) *$\mathcal{T}'$ proves the $\mathbf{\Sigma}_1^B(\mathcal{L}')$-**REPL** axiom scheme.*

Indeed, by Exercise 5.54, the formulas φ and φ_1 can be taken to be *single*-$\mathbf{\Sigma}_1^B(\mathcal{L})$ formulas, and φ_2 can be taken to be a *single*-$\mathbf{\Pi}_1^B(\mathcal{L})$ 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 $\mathbf{\Sigma}_1^1$-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 $\mathbf{\Sigma}_1^1$-definable string function. The case of adding a $\mathbf{\Sigma}_1^1$-definable number function or a $\mathbf{\Delta}_1^1$-definable predicate is similar, and is left as an exercise.

Let F be the $\mathbf{\Sigma}_1^1(\mathcal{L})$-definable function in $\mathcal{T}$. Since $\mathcal{T}$ is a polynomial-bounded theory, F is polynomially bounded in $\mathcal{T}$, and is $\mathbf{\Sigma}_1^B(\mathcal{L})$-definable in $\mathcal{T}$ (Corollary 5.29). So there is a $\mathbf{\Sigma}_1^B(\mathcal{L})$ formula $\varphi_F(\vec{x}, \vec{X}, Y)$ such that

$$Y = F(\vec{x}, \vec{X}) \leftrightarrow \varphi_F(\vec{x}, \vec{X}, Y) \text{ and } \mathcal{T} \vdash \forall\vec{x}\forall\vec{X}\exists!Y \le t\varphi_F(\vec{x}, \vec{X}, Y) \qquad (6.12)$$

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 (6.12) 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 $\mathbf{\Sigma}_1^B(\mathcal{L}')$ formula and φ_2' is equivalent to a $\mathbf{\Pi}_1^B(\mathcal{L}')$ 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. □

Exercise 6.26. *Prove Lemma 6.25* **c** *for the cases of extending $\mathcal{T}$ by a $\mathbf{\Sigma}_1^1$-definable number function and a $\mathbf{\Delta}_1^1$-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 $\mathbf{\Sigma}_0^B(\mathcal{L}_0)$-***REPL** *axiom scheme. 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 $\mathcal{T}_i$ has vocabulary $\mathcal{L}_i$ and each $\mathcal{T}_{i+1}$ is obtained from $\mathcal{T}_i$ by adding the defining axiom for a $\mathbf{\Sigma}_1^1(\mathcal{L}_i)$-definable function or a $\mathbf{\Delta}_1^1(\mathcal{L}_i)$-definable predicate. Let*

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

*Then $\mathcal{T}$ is a polynomial-bounded theory which is conservative over $\mathcal{T}_0$ and proves the $\mathbf{\Sigma}_1^B(\mathcal{L})$-***REPL** *axiom scheme, where $\mathcal{L}$ is the vocabulary of $\mathcal{T}$. Furthermore, each function in $\mathcal{L}$ is $\mathbf{\Sigma}_1^1(\mathcal{L}_0)$-definable in $\mathcal{T}_0$, and each predicate in $\mathcal{L}$ is $\mathbf{\Delta}_1^1(\mathcal{L}_0)$-definable in $\mathcal{T}_0$. Finally each $\mathbf{\Sigma}_1^B(\mathcal{L})$ formula is provably equivalent in $\mathcal{T}$ to a $\mathbf{\Sigma}_1^B(\mathcal{L}_0)$ 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 $\mathbf{\Sigma}_1^B$-**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 6.4.2 when we prove the Witnessing Theorem for $\mathbf{V}^1$.

6.4 The Witnessing Theorem for $\mathbf{V}^1$

To prove the $\Longrightarrow$ direction of Theorem 6.6, i.e., every $\mathbf{\Sigma}_1^1$-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 $\mathbf{\Sigma}_1^1$-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 $\mathbf{\Sigma}_0^B$ 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 $\mathbf{\Sigma}_1^1$-definable functions and $\mathbf{\Delta}_1^1$-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 [**?**]) gives a predicate $Prime(X)$ which according to Corollary 6.8 is $\mathbf{\Delta}_1^B$ 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*

$$Prime(X) \leftrightarrow 2 \le |X| \wedge [\forall Y\forall Z,\ Y\times Z = X \supset (X = Y \vee X = Z)]$$

(Recall that $Y \times Z$ is $\mathbf{\Sigma}_1^1$ definable in $\mathbf{V}^1$, by Exercise 6.11). In fact, the right-to-left direction of this correctness statement implies

$$\forall X\exists Y\exists Z,\ [Y\times Z = X \wedge X \neq Y \wedge X \neq Z)] \vee Prime(X) \vee |X| < 2$$

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 $|X| \ge 2$.

Exercise 6.31 (Integer Factoring [?]). *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 $\mathbf{\Sigma}_1^B$-**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 5.5.2), 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 $\mathbf{\Sigma}_0^B$ 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 $\exists Y \le t\, \varphi(\vec{x}, \vec{X}, Y)$, and proceed as in the proof of Lemma 5.64. In this case, however, a $\mathbf{\Sigma}_1^B$-**COMP** axiom

$$\exists X \le y \forall z < y(X(z) \leftrightarrow \varphi(z)) \tag{6.13}$$

is not in general provably equivalent to a $\mathbf{\Sigma}_1^B$ 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 $\mathbf{\Sigma}_1^1$. To get around this difficulty, we begin by showing that $\mathbf{V}^1$ can be axiomatized by $\mathbf{\Sigma}_1^B$-**IND** and $\mathbf{\Sigma}_0^B$-**COMP** instead of $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_1^B$-**IND** axiom scheme.*

By Exercise 5.54, $\tilde{\mathbf{V}}^1$ can be axiomatized by $\mathbf{V}^0$ and the *single*-$\mathbf{\Sigma}_1^B$-**IND** axiom scheme.

Lemma 6.34. *$\tilde{\mathbf{V}}^1$ proves the $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_1^B$-**IND** axiom scheme. Therefore $\tilde{\mathbf{V}}^1 \subseteq \mathbf{V}^1$. It remains to prove the other direction.

As noted earlier, (6.13) is not in general equivalent to a $\mathbf{\Sigma}_1^B$ formula, so we cannot use $\mathbf{\Sigma}_1^B$-**IND** directly on (6.13) to prove the existence of X. We introduce the number function $\mathit{numones}(y, X)$, which is the number of elements of X that are $< y$. Recall that $\mathit{seq}(u, Z) = (Z)^u$ is the $\mathbf{AC}^0$ function used for coding a finite sequence of numbers (Definition 5.55). The function *numones* has the defining axiom:

$$\begin{aligned}&\mathit{numones}(y, X) = z \leftrightarrow z \le y \wedge \exists Z \le 1 + \langle y, y\rangle,\ (Z)^0 = 0 \wedge (Z)^y = z \wedge \\ &\quad \forall u < y,\ (X(u) \supset (Z)^{u+1} = (Z)^u + 1) \wedge (\neg X(u) \supset (Z)^{u+1} = (Z)^u)\end{aligned} \tag{6.14}$$

Here Z codes a sequence of $(y+1)$ number so that $(Z)^u = \mathit{numones}(u, X)$, for $u \le y$.

Exercise 6.36. **a)** *Show that* (6.14) *is a $\mathbf{\Sigma}_1^B$ 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* (6.14).

b) *Show that the following is a theorem of $\tilde{\mathbf{V}}^1(\mathit{numones})$.*

$$\begin{gathered}\exists x < y(X(x) \wedge \neg Y(x) \wedge \forall u < y,\ u \ne x \supset (X(u) \leftrightarrow Y(u))) \\ \supset \mathit{numones}(y, X) = \mathit{numones}(y, Y) + 1\end{gathered}$$

Although (6.13) may not be $\mathbf{\Sigma}_1^B$, the result of replacing $\leftrightarrow$ by $\supset$ is $\mathbf{\Sigma}_1^B$. 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 (6.13). Then $\eta(y,Y)$ asserts $Y \subseteq X$.

Now consider the formula

$$\psi(w,y) \equiv \exists Y \le y,\ \eta(y,Y) \wedge w = numones(y,Y)$$

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 $\mathbf{\Sigma}_1^B$-**MAX** axioms, which by Definition 5.5 have the form

$$\varphi(0) \supset \exists x \le y,\ \varphi(x) \wedge \neg\exists z \le y(x < z \wedge \varphi(z))$$

where $\varphi(x)$ is $\mathbf{\Sigma}_1^B$. These are provable in $\mathbf{V}^1$ by Corollary 6.1.

Exercise 6.37. *Show that* $\tilde{\mathbf{V}}^1$ *proves the* $\mathbf{\Sigma}_1^B$-**MAX** *axioms. Hint: Apply* $\mathbf{\Sigma}_1^B$-**IND** *to the formula* $\varphi'(x)$ *given by*

$$\exists z \le y, x \le z \wedge \varphi(z)$$

Since *numones* is $\mathbf{\Sigma}_1^1$-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 $\mathbf{\Sigma}_1^B(numones)$-formula is equivalent to some $\mathbf{\Sigma}_1^B$-formula. Hence by Exercise 6.37, $\tilde{\mathbf{V}}^1(numones)$ proves the $\mathbf{\Sigma}_1^B$-**MAX**$(numones)$ axioms.

Now apply $\mathbf{\Sigma}_1^B$-**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 $w_0 \le y$ satisfying $\psi(w_0,y)$. We argued above that the set Y corresponding to w_0 is the set X satisfying (6.13), and this argument can be formalized in $\tilde{\mathbf{V}}^1$ using Exercise 6.36. □

6.4.1 The Sequent System 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 4.26 (for $\Phi = \tilde{\mathbf{V}}^1$), but now we replace the $\mathbf{\Sigma}_1^B$-**IND** axiom scheme by the $\mathbf{\Sigma}_1^B$-**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*

$$\frac{\Gamma, A(b) \longrightarrow A(b+1), \Delta}{\Gamma, A(0) \longrightarrow A(t), \Delta} \tag{6.15}$$

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 $\mathbf{\Sigma}_1^B$.

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 4.4), together with the single-$\mathbf{\Sigma}_1^B$-**IND** rule* (6.15). *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 $\mathbf{\Sigma}_0^B$-**COMP** axiom or a 2-**BASIC** axiom (Figure 5.1) or an $\mathbf{LK}^2$ equality axiom (Definition 4.25).*

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 4.28) to include the rule $\mathbf{\Sigma}_1^B$-**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*** (6.15).

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 $\mathbf{LK}^2$+IND). *Let Ψ and Φ be sets of formulas. Show that if A has an $\mathbf{LK}^2$-Ψ proof, where the Φ-**IND** rule is allowed, then A is a theorem of the theory axiomatized by $\Psi \cup \Phi$-**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 $\mathbf{LK}^2$+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 4.25). Suppose that $\mathcal{T}$ is the theory which is axiomatized by the set of axioms $\Psi \cup \Phi$-**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'-**IND** rule are allowed.*

To apply this to $\tilde{\mathbf{V}}^1$ (and hence to $\mathbf{V}^1$, by Theorem 6.35) take $\mathcal{T} = \tilde{\mathbf{V}}^1$, $\Phi = \mathbf{\Sigma}_1^B$ and $\Psi =$ 2-**BASIC** $\cup\ \mathbf{\Sigma}_0^B$-**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 $\mathbf{LK}^2$+**IND** proof of the type stated above simply as an anchored $\mathbf{LK}^2$-Ψ' proof, with the understanding that the Φ'-**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 2.17 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 2.25 and the Anchored **LK** Completeness Theorem 2.29 (outlined in Exercise 2.30). 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 **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 20, 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 2.25. 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 2.25, 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 2.27) 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 4.25 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 $\mathbf{\Sigma}_1^B$ formulas (indeed, *single*-$\mathbf{\Sigma}_1^B$ formulas), while normally, an $\mathbf{LK}^2$ proof with axiom from $\tilde{\mathbf{V}}^1$ (Definition 2.22) contains cut formulas which are in general not $\mathbf{\Sigma}_1^B$. 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 subsection.

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 $\Psi \cup \Phi$.*

6.4.2 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 5.5.2). Here it suffices to prove Lemma 6.32.

Suppose that $\exists Z\varphi(\vec{a},\vec{\alpha},Z)$ is a $\mathbf{\Sigma}_1^1$ theorem of $\mathbf{V}^1$, where φ is a $\mathbf{\Sigma}_0^B$ 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 2.21 is modified to allow applications of the $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_1^1$ formulas, and in fact they are $\mathbf{\Sigma}_0^B$ formulas or *single*-$\mathbf{\Sigma}_1^1$ formulas. As a result, every sequent in π has the form (5.33):

$$\exists X_1\theta_1(X_1),\ldots,\exists X_m\theta_m(X_m),\Gamma \longrightarrow \Delta,\exists Y_1\psi_1(Y_1),\ldots,\exists Y_n\psi_n(Y_n) \qquad (6.16)$$

for $m,n \geq 0$, where θ_i and ψ_j and all formulas in Γ and Δ are $\mathbf{\Sigma}_0^B$.

We will prove by induction on the depth in π of a sequent $\mathcal{S}$ of the form (6.16) that there is a finite collection of polytime functions

$$\mathcal{L} = \{F_1,\ldots,F_n,\ldots\}$$

so that $\mathbf{V}^1(\mathcal{L})$ proves the (semantic equivalent of the) sequent

$$\mathcal{S}' =_{\text{def}} \theta_1(\beta_1),\ldots,\theta_m(\beta_m),\Gamma \longrightarrow \Delta,\psi_1(F_1),\ldots,\psi_n(F_n) \qquad (6.17)$$

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 Subsection 5.5.2 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 $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_0^B$.

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

$$\Lambda', |\beta| \le r(b) \wedge \psi(b,\beta) \longrightarrow |G(b,\beta)| \le r(b+1) \wedge \psi(b+1, G(b,\beta)), \Pi' \tag{6.18}$$

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 $\exists X \le r(t)\psi(t,X)$ in the succedent of $\mathcal{S}$ by limited recursion (Definition 6.15) as follows:

$$\hat{G}(0,\beta) = \beta \tag{6.19}$$

$$\hat{G}(z+1,\beta) = (G(z,\hat{G}(z,\beta)))^{<r(z+1)} \tag{6.20}$$

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

$$\Lambda', |\hat{G}(b,\beta)| \le r(b) \wedge \psi(b,\hat{G}(b,\beta)) \longrightarrow$$
$$|\hat{G}(b+1,\beta)| \le r(b+1) \wedge \psi(b+1,\hat{G}(b+1,\beta)), \Pi'' \tag{6.21}$$

which is obtained from (6.18) by substituting $\hat{G}(b,\beta)$ for β, and writing $\hat{G}(b+1,\beta)$ for $G(b,\hat{G}(b,\beta))$ (using (6.20)). 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 (6.18) is a theorem of $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L})$, (6.21) is a theorem of $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}')$. Note that (6.21) is of the form

$$\Lambda', \rho(b,\beta) \longrightarrow \rho(b+1,\beta), \Pi'' \tag{6.22}$$

where

$$\rho(b,\beta) \equiv |\hat{G}(b,\beta)| \le r(b) \wedge \psi(b,\hat{G}(b,\beta))$$

Here ρ is a $\mathbf{\Sigma}_0^B(\mathcal{L}')$ formula.

Notice that in Π'', b occurs (only) as an argument to F_i^2. So we cannot apply the **IND** rule to (6.22). 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 $\mathbf{\Sigma}_0^B(\mathcal{L}')$-defining axiom

$$h(\beta) = y \leftrightarrow y \le t \wedge (y = t \vee \neg\rho(y+1,\beta)) \wedge \forall z < y\rho(z+1,\beta) \tag{6.23}$$

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:

$$\mathcal{S}' = \Lambda', |\beta| \le r(0) \wedge \psi(0,\beta) \longrightarrow |G^*(\beta)| \le r(t) \wedge \psi(t, G^*(\beta)), \Pi''' \tag{6.24}$$

Then $\mathcal{S}'$ is of the form (6.17). 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 (6.19) the sequent (6.24) is equivalent to

$$\Lambda', \rho(0,\beta) \longrightarrow \rho(t,\beta), \Pi''' \tag{6.25}$$

Then by replacing b in (6.22) with $h(\beta)$, $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$ proves

$$\Lambda', \rho(h(\beta),\beta) \longrightarrow \rho(h(\beta)+1,\beta), \Pi''' \tag{6.26}$$

Next, by the definition of h (6.23), $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$ 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 (6.26), it follows that $\mathbf{LK}^2$-$\mathbf{V}^1(\mathcal{L}'')$ proves (6.25), and hence (6.24). □

6.5 Notes

Our theory $\mathbf{V}^1$ is essentially Zambella's Theory $\mathbf{\Sigma}_1^p$-*comp* in [**?**], and is a variation of the theory V_1^1 in [**?**], which in turn is defined in the style of Buss's second-order theories [**?**]. It is a two-sorted version of Buss's $\mathbf{S}_2^1$. Our $\mathbf{\Sigma}_1^B$ formulas correspond to *strict* $\mathbf{\Sigma}_1^b$ formulas, but this does not really matter, as shown in Section 6.3.

The $\mathbf{\Sigma}_1^1$ Definability Theorem for $\mathbf{V}^1$ is essentially due to Buss [**?**] who proved it for his first-order theory $\mathbf{S}_2^1$. The interesting part of Theorem 6.35, that $\tilde{\mathbf{V}}^1$ proves the $\mathbf{\Sigma}_1^B$-**COMP** axioms, is essentially Theorem 1 in [**?**].

Chapter 7

Propositional Translations

In Chapter 1 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. In particular a proof system is called *polynomially bounded* if there is a polynomial $p(n)$ such that for every n, every tautology of length n has a proof in the system of size at most $p(n)$. The question of existence (or nonexistence) of a polynomially bounded proof system plays a central role in Theoretical Computer Science.

Each of the theories that we introduce is associated with a proof system. Each $\mathbf{\Sigma}_0^B$ 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, 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 this chapter we will present the propositional translations for $\mathbf{V}^0$ and $\mathbf{V}^1$. Here the corresponding proof systems are constant-depth **Frege** (**AC^0-Frege**), and extended Frege (**eFrege**).

We also generalize the propositional calculus to the quantified propositional calculus (QPC), and introduce various proof systems, such as $\mathbf{G}_1^\star$, for QPC. We show that each $\mathbf{\Sigma}_1^B$ theorem of $\mathbf{V}^1$ can be translated into a family of valid QPC formulas with polynomial size $\mathbf{G}_1^\star$ proofs.

7.1 Propositional Proof Systems

Recall (Chapter 1) 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 1.1). 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 Chapter 1, where a formal proof of a formula A is 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*

$$\mathbf{PK}(X) = \begin{cases} A & \textit{if } X \textit{ codes a } \mathbf{PK} \textit{ proof of} \longrightarrow A \\ \top \ (\textit{True}) & \textit{otherwise} \end{cases}$$

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 **NP** $=$ *co*-**NP**.

Theorem 7.4. *There exists a polynomially bounded proof system iff* **NP** $=$ *co*-**NP**.

Proof. Since $TAUT$ is *co*-**NP**-complete, we have **NP** $=$ *co*-**NP** iff $TAUT \in$ **NP**.

($\Longrightarrow$) Suppose that F is a polynomially bounded proof system. Then by definition, there is a polynomial $p(n)$ such that

$$A \in TAUT \Leftrightarrow \exists X \le p(|A|) F(X) = A$$

This shows that $TAUT \in$ **NP**: The witness for the membership of A in $TAUT$ is the proof X.

($\Longleftarrow$) If $TAUT \in$ **NP**, then there is a polytime relation $R(Y, A)$, and a polynomial $p(n)$ such that

$$A \in TAUT \Leftrightarrow \exists Y \le p(|A|) R(Y, A)$$

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 **NP** $\neq$ *co*-**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.*

7.1.1 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 if "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 ([?]). *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 $\mathcal{S}$:

$$A_\mathcal{S} \equiv \neg A_1 \vee \ldots \vee \neg A_k \vee B_1 \vee \ldots \vee B_\ell \tag{7.1}$$

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

$$\longrightarrow A_{\mathcal{S}_1};\quad \longrightarrow (A_{\mathcal{S}_1} \wedge A_{\mathcal{S}_2});\quad \ldots;\quad \longrightarrow (A_{\mathcal{S}_1} \wedge \ldots \wedge A_{\mathcal{S}_n});\quad \longrightarrow A_{\mathcal{S}_n}$$

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 $A_{\mathcal{S}_1} \wedge \ldots \wedge A_{\mathcal{S}_k}$ are parenthesized. The claim follows easily from the exercise below. □

Exercise 7.9. **a)** *Show that the following sequents have polynomial size cut-free* **PK** *proofs:*

- $\longrightarrow A_\mathcal{S}$, *where* $\mathcal{S}$ *is any axiom of* **PK**.
- $A \wedge B \longrightarrow B$, *for any* **PK** *formulas* A, B.
- $A \wedge B \longrightarrow A \wedge B \wedge B$, *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* **PK** *proofs, for any formula* A:

- $A \wedge A_{\mathcal{S}_1} \longrightarrow A \wedge A_\mathcal{S}$.
- $A \wedge A_{\mathcal{S}_1} \wedge A_{\mathcal{S}_2} \longrightarrow A \wedge A_\mathcal{S}$.

The next result wil 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 **PK** *proof of size bounded by a polynomial in its endsequent.*

Exercise 7.11. *Prove the lemma, using structural induction on $A(p)$.*

7.1.2 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:

$$(p_{i,0} \vee \ldots \vee p_{i,n-1}), \qquad 0 \le i \le n \tag{7.2}$$

$$(\neg p_{i,j} \vee \neg p_{k,j}), \qquad 0 \le i < k \le n,\ 0 \le j < n \tag{7.3}$$

Here, (7.2) says that the pigeon i is placed in some hole, and (7.3) 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*

$$\Big(\bigwedge_{0\le i\le n} \bigvee_{0\le j<n} p_{i,j}\Big) \supset \bigvee_{0\le i<k\le n, 0\le j<n} (p_{i,j} \wedge p_{k,j}) \tag{7.4}$$

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 $p_1, \ldots, p_n$ 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 $(p_1 \vee \ldots \vee p_n)$ has depth 1, for any n, no matter how the parentheses are inserted. The depth of each clause (7.2) 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 $\leq d$. Our definition given above is more general: For proving a formula of depth $\leq d$, 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 1.8), 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 a formula of depth d which is some parenthesization of $(B_1 \vee ... \vee B_n)$, and A' is another such parenthesization, then there is a d-**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. 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 [?]). *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.

7.2 Translating $\mathbf{V}^0$ to bPK

In this section we give evidence that the propositional proof system **bPK** corresponds naturally to the theory $\mathbf{V}^0$. Intuitively a $\mathbf{V}^0$ proof of a $\mathbf{\Sigma}_0^B$ formula is able to use concepts from the complexity class $\mathbf{AC}^0$. Recall from Subsection 4.1 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 $\mathbf{AC}^0$ concepts.

7.2.1 Translating $\mathbf{\Sigma}_0^B$ Formulas

We begin by showing how to translate each $\mathbf{\Sigma}_0^B$ 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 $\mathbf{\Sigma}_0^B$ 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 $\mathbf{\Sigma}_0^B$ formula $\varphi(X)$ which has a single free (string) variable X. We introduce propositional variables $p_0^X, p_1^X, \ldots$, 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

$$\varphi(X)[n] =_{\text{def}} \begin{cases} \top & \text{if } val(t(n)) = val(u(n)) \\ \bot & \text{otherwise} \end{cases}$$

- Similarly if $\varphi(X)$ is $t(|X|) \leq (|X|)$.
- If $\varphi(X)$ is $X(t(|X|))$, then we set $j = val(t(n))$. Define $\varphi(X)[0] =_{\text{def}} \bot$, 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

$$\begin{aligned} \psi(X) \wedge \eta(X)[n] &=_{\text{def}} (\psi(X)[n] \wedge \eta(X)[n]) \\ \psi(X) \vee \eta(X)[n] &=_{\text{def}} (\psi(X)[n] \vee \eta(X)[n]) \\ \neg\psi(X)[n] &=_{\text{def}} \neg\psi(X)[n] \end{aligned}$$

Otherwise, if $\psi(X)[n]$ (or $\eta(X)[n]$) is a logical constant $\bot$ or $\top$, then we simplify the above definitions in the obvious way, e.g.,

$$(\top \wedge \eta(X)[n]) \text{ is simplified to } \eta(X)[n], \qquad (\psi(X)[n] \wedge \bot) \text{ to } \bot,$$

etc.

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

$$\begin{aligned} (\exists y \leq t(|X|)\, \psi(y, X))[n] \quad &=_{\text{def}} \quad \bigvee_{i=0}^{m} \psi(\underline{i}, X)[n] \\ (\forall y \leq t(|X|)\, \psi(y, X))[n] \quad &=_{\text{def}} \quad \bigwedge_{i=0}^{m} \psi(\underline{i}, X)[n] \end{aligned}$$

where $m = val(t(\underline{n}))$, 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 $val(t(\underline{n})) \geq 1$ we have

$$\begin{aligned}(\exists y < t(|X|)\ \psi(y, X))[n] \quad &\leftrightarrow \quad \bigvee_{i=0}^{m-1} \psi(\underline{i}, X)[n] \\ (\forall y < t(|X|)\ \psi(y, X))[n] \quad &\leftrightarrow \quad \bigwedge_{i=0}^{m-1} \psi(\underline{i}, X)[n]\end{aligned}$$

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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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).

$$\begin{aligned}\mathbf{PHP}(y, X) \equiv \forall i \leq y \exists j < y X(i, j)\ \supset& \\ &\exists i \leq y \exists k \leq y \exists j < y (i \neq k \wedge X(i, j) \wedge X(k, j)) \qquad (7.5)\end{aligned}$$

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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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 5.1):

$$|X_i| = |X_k| \wedge \forall x < |X_i|(X_i(x) \leftrightarrow X_k(x))$$

Lemma 7.19. *For every $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ 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. $\square$

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 $\mathbf{\Sigma}_0^B$ 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 $1 \le \vec{m}, \vec{n} \in \mathbb{N}$, $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ has a d-***PK** *proof of size at most $\mathbf{p}(\vec{m}, \vec{n})$.*

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 $\mathbf{\Sigma}_0^B$ formula $\forall y \forall X$ $\mathbf{PHP}(y, X)$ (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 $\mathbf{\Sigma}_1^B$ formulas (i.e., the $\mathbf{\Sigma}_0^B$-**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.

7.2.2 $\widetilde{\mathbf{V}}^0$ 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 $\mathbf{\Sigma}_0^B$-***IND** *axiom scheme.*

Thus $\widetilde{\mathbf{V}}^0$ is the same as $\mathbf{V}^0$, except the $\mathbf{\Sigma}_0^B$-**COMP** axioms are replaced by the $\mathbf{\Sigma}_0^B$-**IND** axioms. By Corollary 5.8, $\mathbf{V}^0$ proves the $\mathbf{\Sigma}_0^B$-**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 $\mathbf{\Sigma}_0^B$-**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 $\mathbf{\Sigma}_0^B$.

Lemma 7.24. $\mathbf{V}^0$ *is $\mathbf{\Sigma}_0^B$-conservative over of $\widetilde{\mathbf{V}}^0$.*

By our definition of semantics (Subsection 4.2.2 and Section 2.2), this is the same as saying that $\mathbf{V}^0$ is $\forall\mathbf{\Sigma}_0^B$-conservative over $\widetilde{\mathbf{V}}^0$, where $\forall\mathbf{\Sigma}_0^B$ is the universal closure of $\mathbf{\Sigma}_0^B$ (Definition 2.23).

Proof. We noted earlier that $\widetilde{\mathbf{V}}^0 \subseteq \mathbf{V}^0$ (by Corollary 5.8). The proof that every $\mathbf{\Sigma}_0^B$ 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 (5.5), 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 $\mathbf{\Sigma}_0^B$ 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 $\mathbf{\Sigma}_0^B$, 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 $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^1$:

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 4.4), together with the $\mathbf{\Sigma}_0^B$-**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 5.1) or an $\mathbf{LK}^2$ equality axiom (Definition 4.25).*

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.

7.2.3 Proof of the Translation Theorem for $\mathbf{V}^0$

By assumption, $\varphi(\vec{a}, \vec{\alpha})$ is a $\mathbf{\Sigma}_0^B$ 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 6.4.2) we modify Definition 2.21 to allow the rule $\mathbf{\Sigma}_0^B$-**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 $\mathbf{\Sigma}_0^B$. 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 $\mathbf{\Sigma}_0^B$ formulas, and $(\vec{b}, \vec{\beta})$ are all the free variables in $\mathcal{S}$ (which may be different for different sequents). 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

$$\mathcal{S}[\vec{m}; \vec{n}] =_{\text{def}} \qquad \ldots, \psi_i(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}], \ldots \longrightarrow \ldots, \eta_j(\vec{b}, \vec{\beta})[\vec{m}; \vec{n}], \ldots$$

has a d-**PK** proof of size at most $\mathbf{p}(\vec{m}, \vec{n})$, for all $\vec{m}, \vec{n} \in \mathbb{N}$.

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 4.25). First, any string variable X can occur in an instance of **B1**–**B12** 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 **B1**–**B12**, then $\longrightarrow \eta$ translates into an axiom of **PK**.

Instances of **L1** and **L2** translate into axioms of **PK**. Consider, for example, an instance of **L1**:

$$\eta(\vec{b}, \gamma, \vec{\beta}) \equiv \gamma(t) \supset t < |\gamma|$$

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 $\mathbf{\Sigma}^B_0$, 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 $\neg$, 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 $\mathcal{S}_2[\vec{m}; \vec{n}]$). No new cut is needed for this step.

For the case of the cut rule, the cut formula $\psi(\vec{b}, \vec{\beta})$ is $\mathbf{\Sigma}^B_0$, and since π is in free variable normal form, no variable is eliminated by the rule. 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 $\mathcal{S}_1[\vec{m}, i; \vec{n}]$:

$$\Lambda[\vec{m}; \vec{n}] \longrightarrow \Pi[\vec{m}; \vec{n}],\ (c \le t(\vec{b}, |\vec{\beta}|))[\vec{m}, i; \vec{n}] \supset \eta(\vec{b}, c, \vec{\beta})[\vec{m}, i; \vec{n}]$$

Note that $c \le t(\vec{b}, |\vec{\beta}|)[\vec{m}, i; \vec{n}]$ is just $\top$ for $i \le r$, where $r = val(t(\vec{m}, \vec{n}))$. So 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

$$\mathcal{S}[\vec{m}; \vec{n}] =_{\text{def}} \Lambda[\vec{m}; \vec{n}] \longrightarrow \Pi[\vec{m}; \vec{n}],\ \bigwedge_{i=0}^{r} \eta(\vec{b}, \underline{i}, \vec{\beta})[\vec{m}; \vec{n}]$$

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 $\mathbf{\Sigma}_0^B$-**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

$$\mathcal{S}_1[\vec{m}, i; \vec{n}] =_{\text{def}} \Lambda[\vec{m}; \vec{n}], \psi(c)[\vec{m}, i; \vec{n}] \longrightarrow \psi(c+1)[\vec{m}, i; \vec{n}], \Pi[\vec{m}; \vec{n}]$$

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 $\psi(i)[\vec{m}; \vec{n}]$, $1 \le i \le r - 1$. 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 $\mathbf{V}^0$. $\square$

Note that the $\mathbf{\Sigma}_0^B$-**IND** axioms are $\mathbf{\Sigma}_0^B$. So in fact we could have defined $\mathbf{LK}^2$-$\widetilde{\mathbf{V}}^0$ to include the $\mathbf{\Sigma}_0^B$-**IND** *axiom scheme* instead of the $\mathbf{\Sigma}_0^B$-**IND** *rule*. Here we can use the following version of the $\mathbf{\Sigma}_0^B$-**IND** axiom:

$$(\varphi(0) \wedge \forall x < t(\varphi(x) \supset \varphi(x+1))) \supset \forall z \leq t\varphi(z) \tag{7.6}$$

where t is any term not involving x or z, and φ is a $\mathbf{\Sigma}_0^B$ formula which may contain other free variables.

In this way, the case of the $\mathbf{\Sigma}_0^B$-**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 $\mathbf{\Sigma}_0^B$-**IND** axiom, and one for the induction step, in the case of the cut rule where the cut formula is an instance of the $\mathbf{\Sigma}_0^B$-**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* (7.6) *above has polynomial size d-**PK** proofs, where d depends only on φ.*

7.3 Quantified Propositional Calculus

Quantified Propositional Calculus (QPC) is an extension of the Propositional Calculus (Chapter 1) 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:*

$$\forall x \exists y[(\neg y \vee (\neg x \wedge p)) \wedge (y \vee x \vee \neg p)] \tag{7.7}$$

A truth assignment is an assignment of truth values F, T 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 1.1) generalize to QPC in the obvious way. So, for example, the formula (7.7) 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. 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. 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 ($\mathbf{\Sigma}_i^q$ and $\mathbf{\Pi}_i^q$). $\mathbf{\Sigma}_0^q = \mathbf{\Pi}_0^q$ *is the class of quantifier-free formulas of QPC. For $i \geq 0$, $\mathbf{\Sigma}_{i+1}^q$ (resp. $\mathbf{\Pi}_{i+1}^q$) is the set of all formulas which have a prenex form where there are at most i alternations of quantifiers, with the outermost quantifier being $\exists$ (resp. $\forall$) if there are* exactly *i alternations.*

Thus

$$\mathbf{\Sigma}_0^q = \mathbf{\Pi}_0^q \subset \ldots \subset \mathbf{\Sigma}_i^q \cap \mathbf{\Pi}_i^q \subset \mathbf{\Sigma}_i^q \cup \mathbf{\Pi}_i^q \subset \mathbf{\Sigma}_{i+1}^q \cap \mathbf{\Pi}_{i+1}^q \subset \ldots$$

Also, checking the validity of a $\mathbf{\Sigma}_i^q$ (resp. $\mathbf{\Pi}_i^q$) sentence is $\mathbf{\Sigma}_i^p$-complete (resp. $\mathbf{\Pi}_i^p$-complete), for $i \geq 1$. For $i = 0$, this problem is $\mathbf{NC}^1$-complete.

7.3.1 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* **NP = PSPACE**.

The assertion **NP** = **PSPACE** is considerably more implausible than **NP** = *co*-**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.

7.3.2 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:

$$\frac{A(B), \Gamma \longrightarrow \Delta}{\forall x A(x), \Gamma \longrightarrow \Delta}\ \textbf{$\forall$-left} \qquad \frac{\Gamma \longrightarrow \Delta, A(p)}{\Gamma \longrightarrow \Delta, \forall x A(x)}\ \textbf{$\forall$-right}$$

$\exists$ introduction rules:

$$\frac{A(p), \Gamma \longrightarrow \Delta}{\exists x A(x), \Gamma \longrightarrow \Delta}\ \textbf{$\exists$-left} \qquad \frac{\Gamma \longrightarrow \Delta, A(B)}{\Gamma \longrightarrow \Delta, \exists x A(x)}\ \textbf{$\exists$-right}$$

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).

Proofs in **G** are dags of sequents, which generalizes the treelike structure of **LK** proofs (see Subsection 7.1.1).

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 1.8. 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

$$\forall x A(x), \Gamma \longrightarrow \Delta \tag{7.8}$$

We can reduce the quantifier depth in $\forall x A(x)$ by showing that (7.8) is valid iff the sequent

$$A(\top), A(\bot), \Gamma \longrightarrow \Delta \tag{7.9}$$

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 ([?]). *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 **G** defined in [**?**] is actually **KPG** as defined in the above exercise. Thus the original **G** and our **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 (7.8) begins with k universal quantifiers, then eliminating them all using (7.9) 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* **G** *with* n *distinct free variables, then* S *has a treelike* **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* **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**. □

7.4 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 $\mathbf{\Sigma}_i^q \cup \mathbf{\Pi}_i^q$. 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 theories are restricted to proving $\mathbf{\Sigma}_i^q$ 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 the arguing that if $\pi = S_1, \ldots, S_n$ is a daglike $\mathbf{G}$ proof then

$$\longrightarrow \forall A_{S_1}; \quad \longrightarrow (\forall A_{S_1} \wedge \forall A_{S_2}); \ldots; \quad \longrightarrow (\forall A_{S_1} \wedge \ldots \wedge A_{S_n}); \quad \longrightarrow A_{S_n} \quad (7.10)$$

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 $\mathbf{\Sigma}_i^q \cup \mathbf{\Pi}_i^q$ then the formula $\forall A_S$ is in $\mathbf{\Pi}_{i+1}^q$.

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)***.)*

- $A \wedge \forall A_{S_1} \longrightarrow A \wedge \forall A_S$.
- $A \wedge \forall A_{S_1} \wedge \forall A_{S_2} \longrightarrow A \wedge \forall A_S$.

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 **G** *proof.*

The next result strengthens Theorem 7.41 for the case $i = 0$.

Theorem 7.43 ([?]). $\mathbf{G}_0^\star$ p-*simulates* $\mathbf{G}_0$ *restricted to proving prenex* $\mathbf{\Sigma}_1^q$ *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 [**?**], 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

$$\longrightarrow \exists x_1 \ldots \exists x_m C(\vec{p}, x_1, \ldots, x_m) \tag{7.11}$$

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

$$\exists x_k \ldots \exists x_m C(\vec{p}, B_1 \ldots B_{k-1}, x_k, \ldots, x_m) \tag{7.12}$$

for some quantifier-free formulas $B_1, \ldots, B_{k-1}$ and some k, $1 \le k \le m$. 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 (7.12), 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 (7.12)) 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 (7.11) using the rules $\exists$-**right**, exchange and contraction. □

The notion of free variable normal form (Definition 2.21) 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.

We now show that for $\mathbf{G}_i^\star$ we may as well assume that all cut formulas are prenex $\mathbf{\Sigma}_i^q$.

Theorem 7.44 ([?]). *Let $\hat{\mathbf{G}}_i^\star$ be $\mathbf{G}_i^\star$ with cut formulas restricted to prenex $\mathbf{\Sigma}_i^q$ 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 $\mathbf{\Sigma}_i^q$, since if A is $\mathbf{\Pi}_i^q$ 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 2.57 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 $\mathbf{\Sigma}_j^q$ or $\mathbf{\Pi}_j^q$ 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. □

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.

7.4.1 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 7.45 (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 $A(\vec{x},\vec{p})$.*

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 show in a later chapter that if π is a $\mathbf{G}_1$ proof (as opposed to a $\mathbf{G}_1^\star$ proof), then the problem becomes complete for the search class **PLS**. Since it seems unlikely that **PLS** problems can all be solved in polynomial time, it seems unlikely that $\mathbf{G}_1^\star$ p-simulates $\mathbf{G}_1$.

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 7.2), or that of the **Frege** systems and $\mathbf{NC}^1$, as we discussed in the Preface.

Definition 7.46 (Extension Cedent). *The sequence of formulas*

$$\Lambda = \qquad e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_n \leftrightarrow B_n \tag{7.13}$$

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 $e_1, \dots, e_{i-1}$. In an **ePK** proof of an existential statement, some of these extension variables are used to witness the existential quantifiers.

Definition 7.47 (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 (7.13) 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 7.48 (Krajíček [?]). *$\mathbf{G}_1^\star$, restricted to proving prenex $\mathbf{\Sigma}_1^q$ 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 7.45. 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

$$e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_n \leftrightarrow B_n \longrightarrow A(\vec{e}_1,\vec{p}) \tag{7.14}$$

Now given the the assignment τ to $\vec{p}$, vaules 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 7.48. 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 (7.14). 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

$$e_1 \leftrightarrow B_1, e_2 \leftrightarrow B_2, ..., e_n \leftrightarrow B_n \longrightarrow \exists\vec{x}A(\vec{x},\vec{p}) \tag{7.15}$$

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

$$\longrightarrow \exists x(x \leftrightarrow B) \tag{7.16}$$

Now apply $\exists$**-left** to (7.15) 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 (7.15), so the variable restriction for this rule is satisfied.) Now apply the cut rule to this and (7.16) 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 (7.15) 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 7.44 we may assume that all cut formulas in π are prenex $\mathbf{\Sigma}_1^q$, so each sequent of π has the form

$$S = \qquad \ldots, \exists\vec{x^i}\alpha_i(\vec{x^i},\vec{r}), \ldots, \Gamma \longrightarrow \Delta, \ldots, \exists\vec{y^j}\beta_j(\vec{y^j},\vec{r}), \ldots \tag{7.17}$$

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 $\vec{q^i} = q^i_1, \ldots, q^i_{\ell_i}$ 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

$$S' = \qquad \Lambda, \ldots, \alpha_i(\vec{q^i}, \vec{r}), ..., \Gamma \longrightarrow \Delta, ..., \beta_j(\vec{e^j}, \vec{r}), ..., \tag{7.18}$$

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 5.64 for the **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

$$S'_1 = \Lambda_1, \Gamma' \to \Delta, \beta(\vec{e}, \overrightarrow{r^1}), C$$

and

$$S'_2 = \Lambda_2, \Gamma' \to \Delta, \beta(\vec{s}, \overrightarrow{r^2}), D$$

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

$$\Lambda_1, \Lambda_2, \Gamma' \to \Delta, \beta(\vec{e}, \vec{r}), \beta(\vec{s}, \vec{r}), (C \wedge D) \tag{7.19}$$

Now we introduce new extension variables $\vec{t}$, and introduce the extension formulas

$$E_i =_{def} [(\beta(\vec{e}, \vec{r}) \wedge e_i) \vee (\neg\beta(\vec{r}, \vec{r}) \wedge s_i)]$$

and define the extension cedent

$$\Lambda_3 = t_1 \leftrightarrow E_1, t_2 \leftrightarrow E_2, ...$$

Then define

$$S' = \qquad \Lambda_1, \Lambda_2, \Lambda_3, \Gamma' \to \Delta, \beta(\vec{t}, \vec{r}), (C \wedge D)$$

One can show with the help of Lemma 7.10 that each of the sequents

$$\Lambda_3, \beta(\vec{e}, \vec{r}) \to \beta(\vec{t}, \vec{r}) \tag{7.20}$$
$$\Lambda_3, \beta(\vec{s}, \vec{r}) \to \beta(\vec{t}, \vec{r}) \tag{7.21}$$

has a short **PK** proof. Using these and (7.19) 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** 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:

$$\frac{S_1 \ S_2}{S_3} = \qquad \frac{\Gamma \to \Delta, C \qquad\qquad C, \Gamma \to \Delta}{\Gamma \to \Delta}$$

Since π is in free variable normal form, every free variable in C also occurs in the conclusion S_3. Supppose 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 5.64 used to prove the $\mathbf{V}^0$ Witnessing Theorem. $\square$

Exercise 7.49. *Carry out the details of* **Case V** *in the above proof.*

7.5 Translating $\mathbf{V}^1$ to $\mathbf{G}_1^\star$

In this section we show that $\mathbf{G}_1^\star$ is closely related to the theory $\mathbf{V}^1$. In fact, $\mathbf{G}_1^\star$ can be considered a nonuniform version of the bounded fragment of $\mathbf{V}^1$.

7.5.1 Translating Bounded $\mathcal{L}_A^2$-Formulas

It is straightforward to extend the propositional translation of $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formulas (Section 7.2) 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 $\mathbf{\Sigma}_i^q$ (resp. $\mathbf{\Pi}_i^q$) 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

$$\underline{\mathbb{N}}_2 \models \forall \vec{X}, (\bigwedge |\vec{X}| = \vec{n}) \supset \varphi(\underline{\vec{m}}, \vec{X})$$

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 $0 \le j < n_i - 1$ for each $n_i \ge 2$.

We define the translation of a bounded $\mathcal{L}_A^2$ formula φ inductively, starting with the $\mathbf{\Sigma}_0^B$ formulas, which is described in Section 7.2. For the induction step, consider the case where $\varphi(\vec{x}, \vec{X}, Y) \equiv \exists Y \le t \psi(\vec{x}, \vec{X}, Y)$, where 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). We drop mention of the $p_j^{X_i}$, and denote $\psi(\vec{x}, \vec{X}, Y)[\vec{m}; \vec{n}, k]$ by $\psi_k(p_0^Y, \ldots, p_{k-2}^Y)$. (If $k \le 1$ then ψ_k does not contain any of the variables $p_0^Y, p_1^Y, \ldots$.) Define

$$\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}] =_{\mathrm{def}} \exists p_0^Y \ldots \exists p_{r-2}^Y \bigvee_{k=0}^{r} \psi_k(p_0^Y, \ldots, p_{k-2}^Y) \tag{7.22}$$

where $r = val(t(\vec{m}, \vec{n}))$. Here the free variables p_j^Y become bound, and if $r \le 1$ 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 $\perp$ or $\top$, then we simplify $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$ in the obvious way.

The case where $\varphi \equiv \forall Y \le t\psi(Y)$ is similar:

$$\forall Y \le t\psi(\vec{x}, \vec{X}, Y)[\vec{m}; \vec{n}] =_{\mathrm{def}} \forall p_0^Y \ldots \forall p_{r-2}^Y \bigwedge_{k=0}^{r} \psi_k(p_0^Y, \ldots, p_{k-2}^Y) \tag{7.23}$$

The cases of the Boolean connectives $\wedge$, $\vee$, $\neg$ or the number quantifiers are the same as for $\mathbf{\Sigma}_0^B$ formulas.

Proposition 7.50. *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 $\mathbf{\Sigma}_i^q$ (resp. $\mathbf{\Pi}_i^q$). 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 $\mathbf{V}^1$ and the proof system $\mathbf{G}_1^\star$ is as follows.

Theorem 7.51 ($\mathbf{V}^1$ Translation Theorem). *For any bounded theorem $\varphi(\vec{x}, \vec{X})$ of $\mathbf{V}^1$, 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 $\varphi(\vec{x}, \vec{X})[\vec{m}; \vec{n}]$, 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. 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 $\mathbf{\Sigma}_1^B$, its translation is a $\mathbf{\Sigma}_1^q$ 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 $\mathbf{\Sigma}_0^B$ formulas, and these are treated as in the proof of the Translation Theorem for $\mathbf{V}^0$. The remaining axioms are instances of $\mathbf{\Sigma}_0^B$-**COMP**, so

$$\mathcal{S} = \qquad \longrightarrow \exists X \le t \forall z < t(X(z) \leftrightarrow \eta(z))$$

and η is a $\mathbf{\Sigma}_0^B$ 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 (replace $\ldots$ by $\vec{m}; \vec{n}$]):

$$\longrightarrow \exists p_0^X \ldots \exists p_{r-2}^X, \bigvee_{k=0}^{r} \Big(\bigwedge_{i=0}^{k-2} (p_i^X \leftrightarrow \eta(\underline{i})[\ldots]) \wedge \eta(\underline{k-1})[\ldots] \wedge \bigwedge_{i=k}^{r-1} \neg\eta(\underline{i})[\ldots]\Big)$$

where the conjunct $\eta(\underline{k-1})$ is deleted when $k = 0$.

Exercise 7.52. *Let $A_0, \ldots, A_\ell$ be any* **PK** *formulas ($\ell \geq 0$). Show that the sequent*

$$\longrightarrow \bigvee_{j=-1}^{\ell} (A_j \wedge \bigwedge_{i=j+1}^{\ell} \neg A_i)$$

(where for $j = -1$ the conjunct A_j is deleted) has a polynomial size treelike cut-free **PK** *derivation.*

We get $\mathcal{S}[\vec{m}; \vec{n}]$ by first using the above exercise for $\ell = r - 1$ and $A_i \equiv \eta(\underline{i})[\vec{m}; \vec{n}]$, then repeatedly applying the $\exists$-**right** rule. Thus $\mathcal{S}[\vec{m}; \vec{n}]$ has a polynomial size cut-free **G** proof.

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 155), although now the translation will require cuts on $\mathbf{\Sigma}_i^q$ 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 $|\gamma| \leq t[\ldots, n]$ is either $\top$ or $\bot$. Let $r = \mathit{val}(t)$, then

$$\mathcal{S}_1[\ldots, n] =_{\text{def}} \begin{cases} \Lambda[\ldots, n] \longrightarrow \Pi[\ldots, n], \psi(\gamma)[\ldots, n] & \text{if } n \leq r \\ \Lambda[\ldots, n] \longrightarrow \Pi[\ldots, n], \bot & \text{if } n > r \end{cases} \tag{7.24}$$

By definition (see (7.22)),

$$\mathcal{S}[\ldots, n] =_{\text{def}} \qquad \Lambda[\ldots, n] \longrightarrow \Pi[\ldots, n], \exists p_0^Z \ldots \exists p_{r-2}^Z \bigvee_{k=0}^{r} \psi(Z)[\ldots, k]$$

Consider the interesting case where $n \leq r$, First, by repeated applications of the rules **weakening** and $\vee$-**right**, we obtain from $\mathcal{S}_1[\ldots, n]$

$$\Lambda[\ldots, n] \longrightarrow \Pi[\ldots, n], \bigvee_{k=0}^{r} \psi(\gamma)[\ldots, k]$$

Then we can derive $\mathcal{S}[\ldots, n]$ 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 $\mathbf{\Sigma}_1^B$. Let $r = val(t)$, then for $n \le r$,

$$\mathcal{S}_1[\ldots, n] =_{\text{def}} \qquad \psi(\gamma)[\ldots, n], \Lambda[\ldots] \longrightarrow \Pi[\ldots] \tag{7.25}$$

Also,

$$\mathcal{S}[\ldots] =_{\text{def}} \qquad \exists p_0^Z \ldots \exists p_{r-2}^Z \bigvee_{n=0}^{r} \psi(Z)[\ldots, n],\ \Lambda[\ldots] \longrightarrow \Pi[\ldots]$$

Now if $r = 0$, then we are done. Otherwise, combine the sequents $\mathcal{S}_1[\ldots, n]$ for $n = 0, \ldots, r$ by the rule $\vee$-**left** we obtain

$$\bigvee_{n=0}^{r} \psi(\gamma)[\ldots, n],\ \Lambda[\ldots] \longrightarrow \Pi[\ldots]$$

Thus we get $\mathcal{S}[\ldots]$ by $r - 1$ applications of the $\exists$-**left** rule. □

Exercise 7.53. *Carry out the cases for the string $\forall$-introduction rules.*

7.6 Notes

Definitions 7.2, 7.5 and Theorem 7.4 are from [?]. Also, the fact that **Frege** proof systems are p-equivalent is proved in [?].

The first propositional translation of an arithmetic theory is described in [?]. The translation of $\mathbf{\Sigma}_0^B$ formulas given in Subsection 7.2.1 is from [?], 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 [?].

A proof system for the Quantified Propositional Calculus was introduced by Dowd [?]. The system $\mathbf{G}$ and its subsystems $\mathbf{G}_i$ were introduced by Krajíček and Pudlák [?] (see also Section 4.6 of [?]). 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 $\mathbf{\Sigma}_i^q$ or $\mathbf{\Pi}_i^q$. Our definitions are due to Morioka [?]. Theorem 7.37 is new.

The idea of $\mathbf{G}_i^\star$ (treelike $\mathbf{G}_i$) is from [?], and the $\mathbf{V}^1$ Translation Theorem 7.51 is adapted from a similar theorem for $\mathbf{S}_2^1$ also in [?]. Theorem 7.45 is from [?].

Chapter 8

Theories for Polynomial Time and Beyond

Here we introduce several equivalent "minimal" theories for polynomial time, and show that those with the basic vocabulary $\mathcal{L}_A^2$ are finitely axiomatizable. The theory $\mathbf{V}^1$ has the same $\mathbf{\Sigma}_1^B$ theorems as these minimal theories, but apparently has more $\mathbf{\Sigma}_2^B$ theorems. We also introduce the $\mathbf{TV}^i$ hierarchy and show that $\mathbf{TV}^0$ is one of the minimal theories for polynomial time, while $\mathbf{TV}^1$ is associated with the class **PLS** (Polynomial Local Search). Finally we show that our two-sorted theories are "RSUV-isomorphic" to appropriate single-sorted theories.

8.1 The Theory VPV

In Chapter 6 we proved that the $\mathbf{\Sigma}_1^1$-definable functions of $\mathbf{V}^1$ are precisely the polynomial time functions **FP**. However there is an (apparently) weaker theory, $\mathbf{TV}^0$, which captures the same class of functions in the same way, and proves the same $\mathbf{\Sigma}_1^B$ theorems as $\mathbf{V}^1$. (Apparently $\mathbf{TV}^0$ does not prove either the $\mathbf{\Sigma}_0^B$-**REPL** scheme or the $\mathbf{\Sigma}_1^B$-**COMP** scheme, but these do not consist of $\mathbf{\Sigma}_1^B$ formulas.) We argue that this theory seems to be the "minimal" theory which formalizes polynomial time reasoning.

To support the claim that $\mathbf{TV}^0$ is minimal, we first define an equivalent universal theory **VPV** which contains function symbols for all functions in **FP**. To argue that **VPV** is minimal, we take for granted that a minimal theory for any complexity class containing the $\mathbf{AC}^0$ functions should contain the basic theory $\mathbf{V}^0$ (Chapter 5) associated with $\mathbf{AC}^0$. Since the universal theory $\overline{\mathbf{V}}^0$ is a conservative extension of $\mathbf{V}^0$, we use $\overline{\mathbf{V}}^0$ as the starting point. To extend $\overline{\mathbf{V}}^0$ to our polytime theory **VPV**, we simply add a function symbol and its defining axiom for each way of defining a polytime function, using some standard method of defining polytime functions. The method we choose is based on

Cobham's Theorem 6.16. There are of course other ways of defining the polytime functions, but the resulting theories turn out to be equivalent to **VPV** (at least for standard methods of defining polytime functions).

The vocabulary $\mathcal{L}_{\mathbf{FP}}$ for **VPV** extends the vocabulary $\mathcal{L}_{\mathbf{FAC}^0}$ for $\overline{\mathbf{V}}^0$ (see Section 5.6). The difference is that now we introduce new functions based on Limited Recursion.

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

$$F(0,\vec{x},\vec{X}) = G(\vec{x},\vec{X}) \tag{8.1}$$

$$F(y+1,\vec{x},\vec{X}) = (H(y,\vec{x},\vec{X},F(y,\vec{x},\vec{X})))^{<t(y,\vec{x},\vec{X})} \tag{8.2}$$

where now the bounding term $t(y,\vec{x},\vec{X})$ is in $\mathcal{L}_A^2$.

Definition 8.1. *The vocabulary $\mathcal{L}_{\mathbf{FP}}$ is the smallest set that satisfies*

1) *$\mathcal{L}_{\mathbf{FP}}$ includes $\mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}\}$.*
2) *For each open formula $\varphi(z,\vec{x},\vec{X})$ over $\mathcal{L}_{\mathbf{FP}}$ and term $t = t(\vec{x},\vec{X})$ of $\mathcal{L}_A^2$ there is a string function $F_{\varphi,t}$ and a number function $f_{\varphi,t}$ 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 $F = F_{G,H,t}$ in $\mathcal{L}_{\mathbf{FP}}$ with defining equations (8.1,8.2).*

By Cobham's Theorem, it is clear that semantically the functions of $\mathcal{L}_{\mathbf{FP}}$ comprise the polytime functions.

We now define the theory **VPV** in the style of Definition 5.68 of $\overline{\mathbf{V}}^0$.

Definition 8.2. **VPV** *is the theory over $\mathcal{L}_{\mathbf{FP}}$ with the following set of axioms:* **B1**-**B11**, **L1**, **L2** *(Figure 5.1),* **B12′** *and* **B12″** (5.39), (5.40), **SE′** (5.41)*, and defining axioms* (5.37) *for each function $F_{\varphi,t}$ in $\mathcal{L}_{\mathbf{FP}}$ and defining axiom* (5.38) *for each function $f_{\varphi,t}$ in $\mathcal{L}_{\mathbf{FP}}$ and defining axioms (8.1,8.2) 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 8.1 is explicitly bounded by a term in $\mathcal{L}_A^2$, and hence **VPV** is a polynomial-bounded theory. Further it is easy to see, using the definitions of $F_{\varphi,t}$ and $f_{\varphi,t}$, that the functions in $\mathcal{L}_{\mathbf{FP}}$ are closed under composition. Hence by Cobham's Theorem 6.16 the symbols in **FP** represent precisely the functions in **FP**.

The following result can be proved by structural induction on φ in the same way as Lemma 3.44 and Lemma 5.69.

Lemma 8.3. *For every $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$ formula φ there is an open $\mathcal{L}_{\mathbf{FP}}$-formula φ^+ such that $\mathbf{VPV} \vdash \varphi \leftrightarrow \varphi^+$.*

Next we state a general witnessing theorem for universal theories, which applies to **VPV**.

Theorem 8.4 (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 $\mathcal{L}^2_A$ there is a function $F_{\varphi,t}$ in $\mathcal{L}$ such that*

$$\mathcal{T} \vdash F_{\varphi,t}(\vec{x}, \vec{X})(z) \leftrightarrow z < t \wedge \varphi(z, \vec{x}, \vec{X})$$

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 5.6.1. This proof defines the witnessing function F by cases, and in fact F has the form $F_{\varphi,t}$ for suitable φ, t. By our assumption that $\mathcal{T}$ is polynomial-bounded, we know that there is a bounding term t for $F_{\varphi,t}$ in $\mathcal{L}^2_A$ (as opposed to $\mathcal{L}$). □

Corollary 8.5 (Witnessing for VPV). *Every $\mathbf{\Sigma}^1_1(\mathcal{L}_{\mathbf{FP}})$ 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 quantier-free) can be witnessed, it is easy to generalize it to witness an arbitrary $\mathbf{\Sigma}^1_1(\mathcal{L}_{\mathbf{FP}})$ formula $\exists\vec{z}\exists\vec{Z}\varphi$. (See Lemma 5.64 and how it is used to prove the witnessing theorem for $\mathbf{V}^0$.) □

This witnessing result immediately implies the following.

Corollary 8.6. *Every function $\mathbf{\Sigma}^1_1$-definable in* **VPV** *is in* **FP***.*

Of course this holds whether we interpret $\mathbf{\Sigma}^1_1$-definable to mean $\mathbf{\Sigma}^1_1(\mathcal{L}^2_A)$-definable, or more generally $\mathbf{\Sigma}^1_1(\mathcal{L}_{\mathbf{FP}})$-definable. The converse of the latter, that every polytime function is $\mathbf{\Sigma}^1_1(\mathcal{L}_{\mathbf{FP}})$-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 is $\mathbf{\Sigma}^1_1(\mathcal{L}^2_A)$-definable in **VPV**. This is not straightforward to prove, mainly because we do not have the $\mathbf{\Sigma}^B_0$-**REPL** axioms available in **VPV**. (See Section 6.3.1 for how we can proceed if $\mathbf{\Sigma}^B_0$-**REPL** were available.) One method would be to introduce the aggregate function F^* of a function F, as we do in Section 9.2.3, to prove the analogous result for $\overline{\mathbf{VTC}}^0$. But here we take a different approach: Since $\mathbf{V}^1$ proves the $\mathbf{\Sigma}^B_0$-**REPL** axioms it is relatively easy to show that every $\mathcal{L}_{\mathbf{FP}}$ function is $\mathbf{\Sigma}^1_1(\mathcal{L}^2_A)$-definable in $\mathbf{V}^1$. From this we use the fact that $\mathbf{\Sigma}^1_1$ theorems of $\mathbf{V}^1$ are witnessed in **VPV** to get our desired result (Theorem 8.15).

The next result is proved in the same way as Lemma 5.70.

Lemma 8.7. **VPV** *proves the* $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FP}})$-**COMP***,* $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FP}})$-**IND***, and* $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FP}})$-**MIN** *axiom schemes.*

Definition 8.8 ($\mathbf{\Delta}_i^B$ Formula). *Let $\mathcal{T}$ be a theory over $\mathcal{L} \supseteq \mathcal{L}_A^2$. We say that a formula φ is $\mathbf{\Delta}_i^B(\mathcal{L})$ in $\mathcal{T}$ if there is a $\mathbf{\Sigma}_i^B(\mathcal{L})$ formula φ_1 and a $\mathbf{\Pi}_i^B(\mathcal{L})$ formula φ_2 such that $\mathcal{T} \vdash \varphi \leftrightarrow \varphi_1$ and $\mathcal{T} \vdash \varphi \leftrightarrow \varphi_2$.*

Corollary 8.9. *If φ is $\mathbf{\Delta}_1^B(\mathcal{L}_{\mathbf{FP}})$ in* **VPV** *then* **VPV** $\vdash \varphi \leftrightarrow \varphi_0$ *for some open $\mathcal{L}_{\mathbf{FP}}$-formula φ_0.*

Proof. Suppose that φ is $\mathbf{\Delta}_1^B(\mathcal{L}_{\mathbf{FP}})$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$-formulas ψ_1, ψ_2 we have

$$\varphi_1 \equiv \exists Y \leq t_1 \psi_1(\vec{x}, \vec{X}, Y)$$
$$\varphi_2 \equiv \forall Z \leq t_2 \psi_2(\vec{x}, \vec{X}, Z)$$

Since **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 8.5 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 **VPV** $\vdash \varphi \leftrightarrow \varphi_0$, where $\varphi_0 \equiv \psi_1(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}))$. By Lemma 8.3 we may assume ψ_1 is an open $\mathcal{L}_{\mathbf{FP}}$-formula, as required. □

8.1.1 Comparing VPV and V^1

Here we prove that every $\mathcal{L}_A^2$-theorem of **VPV** is provable in $\mathbf{V}^1$. We also prove a partial converse, that every $\mathbf{\Sigma}_1^1$ theorem of $\mathbf{V}^1$ is provable in **VPV**. Later we show evidence that not all $\mathbf{\Sigma}_2^B$ 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 $\mathbf{\Sigma}_1^1$ theorem of $\mathbf{V}^1$ can be, provably in **VPV**, witnessed by functions in $\mathcal{L}_{\mathbf{FP}}$.

Definition 8.10. *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 8.11. **a)** *Every function in $\mathcal{L}_{\mathbf{FP}}$ is $\mathbf{\Sigma}_1^B$-definable in $\mathbf{V}^1$.*

b) *Every $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$-formula is provably equivalent in $\mathbf{V}^1(\mathbf{VPV})$ to a $\mathbf{\Sigma}_1^B(\mathcal{L}_A^2)$-formula.*

c) *For $i \geq 1$,$\mathbf{V}^i(\mathbf{VPV})$ is conservative over $\mathbf{V}^i$.*

Corollary 8.12. $\mathbf{V}^1(\mathbf{VPV})$ *proves the* $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$**-COMP**, $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$**-IND**, *and* $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$**-MIN** *axiom schemes.*

Proof. The corollary follows immediately from part b) of the theorem, since $\mathbf{V}^1$ proves these schemes for $\mathbf{\Sigma}_1^B(\mathcal{L}_A^2)$-formulas.

The Theorem follows from Corollary 6.27, where we take $\mathcal{T}_0$ to be $\mathbf{V}^1(Row)$, or $\mathbf{V}^i$(Row) for part c) (we can get rid of the function *Row* by Lemma 5.51), 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 $\mathbf{\Sigma}_1^1$-definable in $\mathcal{T}_i$ (and even in $\mathcal{T}_0$) is proved in Section 6.2.2. □

Theorem 8.13. *Every* $\mathbf{\Sigma}_1^1(\mathcal{L}_{\mathbf{FP}})$ *theorem of* $\mathbf{V}^1(\mathbf{VPV})$ *is witnessed in* $\mathbf{VPV}$ *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 6.4.2 proves this theorem. Note that every witnessing function introduced is in **FP**, and, noting that **VPV** proves $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$-**IND** (by Lemma 8.7), we see that **VPV** proves the desired sequents. □

The following corollary is immediate from Theorem 8.13.

Corollary 8.14. **VPV** *and* $\mathbf{V}^1(\mathbf{VPV})$ *have the same* $\mathbf{\Sigma}_1^1(\mathcal{L}_{\mathbf{FP}})$ *theorems.*

In particular, every $\mathbf{\Sigma}_1^B$ theorem of $\mathbf{V}^1$ is provable in **VPV**. From this and Corollary 8.6 and part a) of Theorem 8.11 we have the following:

Theorem 8.15 ($\mathbf{\Sigma}_1^1$-Definability Theorem for VPV). *A function is* $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$*-definable in* **VPV** *iff it is in* **FP**.

Finally, from Corollary 8.14 and part b) of Theorem 8.11 we have

Theorem 8.16. *Every* $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$*-formula is provably equivalent in* **VPV** *to a* $\mathbf{\Sigma}_1^B(\mathcal{L}_A^2)$*-formula.*

8.2 $\mathbf{TV}^0$ and the $\mathbf{TV}^i$ Hierarchy

The theory **VPV** has an infinite vocabulary $\mathcal{L}_{\mathbf{FP}}$, and although it satisfies our desire for a "minimal" theory for **P** in terms of proving power, we would like to find an equivalent theory over the base vocabulary $\mathcal{L}_A^2$. We now introduce the theory $\mathbf{TV}^0$, which satisfies this condition. This theory is the first in a hierarchy of theories $\mathbf{TV}^i$, where for $i > 0$ $\mathbf{TV}^i$ corresponds — in the sense of Section 8.7 — to Buss's single-sorted theory $\mathbf{T}_2^i$.

The theory $\mathbf{TV}^i$ is the same as $\mathbf{V}^i$, except instead of the $\mathbf{\Sigma}_i^B$-**COMP** axioms we introduce the $\mathbf{\Sigma}_i^B$ "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 $\mathbf{\Sigma}_0^B$-bit definition

$$S(X)(i) \leftrightarrow \varphi_S^{bit}(i, X) \tag{8.3}$$

where

$$\varphi_S^{bit}(i, X) \;\equiv\; i \le |X| \wedge [(X(i) \wedge \exists j < i \neg X(j)) \vee (\neg X(i) \wedge \forall j < i X(j))]$$

Definition 8.17 (String Induction Axiom). *If Φ is a set of formulas, then the string induction axiom scheme, denoted Φ-**SIND**, is the set of all formulas*

$$[\varphi(\varnothing) \wedge \forall X(\varphi(X) \supset \varphi(S(X))] \supset \varphi(Y) \tag{8.4}$$

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 (8.4) as a formula over $\mathcal{L}_A^2$, using the standard method of eliminating $\mathbf{\Sigma}_0^B$-bit-definable function symbols (Lemma 5.40).

Definition 8.18. *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 $\mathbf{\Sigma}_i^B$-**SIND** scheme.*

Although the induction scheme (8.4) 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 $\mathbf{\Sigma}_{i+1}^B$-formulas.

Lemma 8.19. *For $i \geq 0$, $\mathbf{TV}^i$ proves $\mathbf{\Sigma}_i^B$-**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 $\mathbf{\Sigma}_i^B$.

We need the following easily verified fact:

$$\mathbf{V}^0 \vdash (|S(X)| = |X| \ \vee\ |S(X)| = |X| + 1) \tag{8.5}$$

Reasoning in $\mathbf{TV}^i$, assume

$$[\varphi(0) \wedge \forall x,\ \varphi(x) \supset \varphi(x+1)]$$

From this and (8.5) 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 $\mathbf{\Sigma}_i^B$-**SIND**, where X_z is a string with length z. Hence $\varphi(z)$. □

Theorem 8.20. *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 $\mathbf{V}^0 + \mathbf{\Sigma}_i^B$-**IND**. The proof of Theorem 6.35 easily generalizes to show $\mathbf{V}^i = \tilde{\mathbf{V}}^i$. Hence the theorem follows from Lemma 8.19. □

Just as $\mathbf{V}^i$ proves the number minimization and maximization axioms for $\mathbf{\Sigma}_i^B$-formulas (Corollary 5.8), $\mathbf{TV}^i$ proves the stronger string minimization and maximization axioms for $\mathbf{\Sigma}_i^B$-formulas. First, we define the ordering relation for strings.

Definition 8.21 (String Ordering). *The string relation $X \le Y$ has defining axiom*

$$X \le Y \leftrightarrow [X = Y \;\vee\; (|X| \le |Y| \wedge \exists z \le |Y|\,(Y(z) \wedge \neg X(z) \wedge \forall u \le |Y|,\, z < u \supset (X(u) \supset Y(u))))] \quad (8.6)$$

Often, our vocabularies do not contain extra relation symbols outside $\mathcal{L}_A^2$. Thus, the syntactic formula $X \le Y$ will be an abbreviation for the RHS of Equation (8.6).

Exercise 8.22. *Show that the following are theorems of $\mathbf{V}^0$:*

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 8.23 (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 8.24. *For $i \ge 0$, $\mathbf{TV}^i$ proves the $\mathbf{\Sigma}_i^B$-**SMIN** and $\mathbf{\Sigma}_i^B$-**SMAX** axioms.*

Proof. To prove $\mathbf{\Sigma}_i^B$-**SMAX**, let $\varphi(X)$ be a $\mathbf{\Sigma}_i^B$-formula. Let $\varphi'(X)$ be the $\mathbf{\Sigma}_i^B$-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 (8.4) applied to $\varphi'(X)$.

The proof of $\mathbf{\Sigma}_i^B$-**SMIN** is similar, but uses the binary subtraction function $Z \dot{-} Y$.

Exercise 8.25. *Show that the limited substraction function for string $Z \dot{-} Y$ is $\mathbf{\Sigma}_0^B$-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 8.26. **VPV** *is a conservative extension of* $\mathbf{TV}^0$.

Before proving this theorem, we list some of its corollaries.

Corollary 8.27. *For* $i \geq 0$, $\mathbf{TV}^i(\mathbf{VPV})$ *is a conservative extension of* $\mathbf{TV}^i$.

Proof. For $i = 0$ this follows from Theorem 8.26. For $i \geq 1$ we know $\mathbf{V}^1 \subseteq \mathbf{TV}^i$, and hence $\mathbf{TV}^i$ $\mathbf{\Sigma}_1^B$-defines all functions in $\mathcal{L}_{\mathbf{FP}}$, and also $\mathbf{TV}^i$ proves $\mathbf{\Sigma}_1^B$-**REPL** by Corollary 6.24. Therefore the corollary follows from Corollary 6.27. $\square$

From Theorem 8.26 and Theorem 8.15 we conclude

Theorem 8.28 ($\mathbf{\Sigma}_1^1$-Definability Theorem for $\mathbf{TV}^0$). *A function is* $\mathbf{\Sigma}_1^1$-*definable in* $\mathbf{TV}^0$ *iff it is in* **FP**.

From Theorem 8.26 and part c) of Theorem 8.11 we conclude $\mathbf{TV}^0 \subseteq \mathbf{V}^1$. From this and Corollary 8.14 we have the following (recall the notion of a Φ-conservative extension from Definition 7.23):

Corollary 8.29. $\mathbf{V}^1$ *is* $\mathbf{\Sigma}_1^B$-*conservative over* $\mathbf{TV}^0$.

As remarked above, $\mathbf{TV}^0$ is axiomatized by $\mathbf{\Sigma}_1^B$ formulas (unlike $\mathbf{V}^1$).

The proof of Theorem 8.26 takes up the next two subsections. In short, to show that **VPV** extends $\mathbf{TV}^0$, we indeed show that **VPV** proves the (contrapositive of) $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$-**SIND** by using the "binary search" function. To prove conservativity we introduce the bit recursion axiom scheme, and prove Theorem 8.38 and Lemma 8.39.

8.2.1 $\mathbf{TV}^0 \subseteq \mathbf{VPV}$

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 5.43. We also need the string relation $X \leq Y$ (Definition 8.21) and the string function $POW2(x)$ defined below. The intended meaning of $POW2(x)$ is such that (see Notation on page 76) $bin(POW2(x)) = 2^x$.

Example 8.30. *The string function* $POW2(x)$, *also denoted by* $\{x\}$, *has bit defining axiom*

$$POW2(x)(i) \leftrightarrow i = x$$

Exercise 8.31. *Show that* $\overline{\mathbf{V}}^0$ *proves the following:*

$$\begin{array}{rcl} X + POW2(0) & = & S(X) \\ X & < & POW2(|X|) \\ POW2(i) + POW2(i) & = & POW2(i+1) \end{array}$$

Now we prove the half of Theorem 8.26 stating that **VPV** is an extension of $\mathbf{TV}^0$. For this it suffices to show that **VPV** proves the $\mathbf{\Sigma}_0^B$-**SIND**-axioms. In fact, we prove a slightly stronger result.

Lemma 8.32. **VPV** *proves the* $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$-**SIND**-*axioms.*

Proof. By Lemma 8.3 we may assume that $\varphi(X)$ in (8.4) 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 **VPV** an $\mathcal{L}_{\mathbf{FP}}$ function $G(\vec{y}, \vec{Y}, X)$ such that **VPV** proves

$$(\varphi(\varnothing) \wedge \neg\varphi(X)) \supset ((\varphi(G(\vec{y}, \vec{Y}, X)) \wedge \neg\varphi(S(G(\vec{y}, \vec{Y}, X))) \tag{8.7}$$

from which (8.4) 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 (8.1,8.2) 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 $\dot{-}$ is limited subtraction, Section 3.3.3).

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 8.7), using Exercises 5.43 and 8.31. 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 (8.7) follows from these two formulas and $X + POW2(0) = S(X)$ (Exercise 8.31). □

Recall the notion of a $\mathbf{\Delta}_i^B$ formula in a theory (Definition 8.8).

Definition 8.33. *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* $\mathbf{\Delta}_i^B$-**AX** *if for any $\mathbf{\Delta}_i^B(\mathcal{L})$ formula φ in $\mathcal{T}$, $\mathcal{T}$ proves the* **AX** *axiom for φ.*

From Lemma 8.32 and Corollary 8.9 we have

Corollary 8.34. **VPV** *proves* $\mathbf{\Delta}_1^B$-**SIND**.

8.2.2 VPV is Conservative over $\mathbf{TV}^0$

In order to show that **VPV** is conservative over $\mathbf{TV}^0$, 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 $Cut(i, X)$ (6.5)

$$\varphi^{rec}(y, X) \equiv \forall i < y(X(i) \leftrightarrow \varphi(i, X^{<i}))$$

In case $\varphi(i, X)$ is an $\mathcal{L}^2_A$-formula we can interpret $\varphi^{rec}(y, X)$ as an $\mathcal{L}^2_A$-formula by eliminating occurrences of $Cut(i, X)$ using the standard method of eliminating $\mathbf{\Sigma}^B_0$-bit-definable function symbols (Lemma 5.40).

If $\varphi(i, X)$ is in $\mathbf{\Sigma}^B_0$ 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 8.35. *If Φ is a set of formulas, then the bit recursion axiom scheme, denoted Φ-**BIT-REC**, is the set of formulas*

$$\exists X \varphi^{rec}(y, X) \tag{8.8}$$

where $\varphi(i, X)$ is in Φ, and may have free variables other than X.

We will show that the axiom scheme $\mathbf{\Sigma}^B_0$-**SIND** in the definition of $\mathbf{TV}^0$ (Definition 8.18) can be replaced by $\mathbf{\Sigma}^B_0$-**BIT-REC**. First, we show that $\mathbf{\Sigma}^B_0$-**BIT-REC** can be used to formalize the computation of polytime Turing machines:

Theorem 8.36. *Every function in* **FP** *is $\mathbf{\Sigma}^B_1$-definable in $\mathbf{V}^0 + \mathbf{\Sigma}^B_0$-**BIT-REC**.*

Proof. We refer to the proof in Section 6.2.1 that every function in **FP** is $\mathbf{\Sigma}^B_1$-definable in $\mathbf{V}^1$. It suffices to show that any polytime string function $F(\vec{x}, \vec{X})$ is $\mathbf{\Sigma}^B_1$-definable in $\mathbf{V}^0 + \mathbf{\Sigma}^B_0$-**BIT-REC**, since every number function in **FP** has the form $|F(\vec{x}, \vec{X})|$ for some F in **FP** (Proposition 6.5).

Let M be a polynomial time Turing machine which computes $F(\vec{x}, \vec{X})$. According to Exercise 6.13 there are $\mathbf{\Sigma}^B_0$-bit-definable functions $Init_{\mathsf{M}}(\vec{x}, \vec{X})$, $Next_{\mathsf{M}}(Z)$ and $Out_{\mathsf{M}}(Z)$ which describe the computation of M on input $\vec{x}, \vec{X}$ by giving the initial configuration, next configuration, and output of the computation. Our goal is to find a $\mathbf{\Sigma}^B_0$-formula $\varphi(i, \vec{x}, \vec{X}, Z)$ such that $\varphi^{rec}(i, \vec{x}, \vec{X}, Z)$ asserts that the first i bits of Z are the first i bits of the computation of M on input $\vec{x}, \vec{X}$. In order to do this, we will let Z code the computation as a concatenation of the successive configurations of M, rather than our usual method of letting $Z^{[0]}, Z^{[1]}, \cdots$ be the successive configurations. (The problem with our usual method is that according to our definition of the pairing function (5.21), the index for an element in row $Z^{[i+1]}$ can be less than the index of some element in row $Z^{[i]}$.)

Definition 8.37 (The Substring Function). *The string function $Z[u, v]$ is intended to code the substring $Z(u), Z(u + 1), \cdots, Z(v - 1)$ of Z. It has the $\mathbf{\Sigma}_0^B$-bit-defining axiom*

$$Z[u, v](i) \leftrightarrow i < v \dot{-} u \wedge Z(u + i)$$

Let $t = t(\vec{x}, \vec{X})$ be an $\mathcal{L}_A^2$-term bounding the run-time of M on input $\vec{x}, \vec{X}$. If Z codes the computation of M on input $\vec{x}, \vec{X}$, then the successive configurations of M form the sequence

$$Z[0, t], Z[t, 2t], \cdots, Z[t^2 - t, t^2]$$

The i-th bit of Z is defined from the previous bits using the formula (suppressing the arguments $\vec{x}, \vec{X}$)

$$\begin{aligned}\varphi(i, Z) \ \equiv\ & (i < t \wedge \mathit{Init}_{\mathsf{M}}(\vec{x}, \vec{X})(i)) \vee \\ & \quad (t \le i \wedge \mathit{Next}_{\mathsf{M}}(Z[i \dot{-} t \dot{-} (i \bmod t), i \dot{-} (i \bmod t)])(i \bmod t))\end{aligned}$$

Thus the computation of M on input $\vec{x}, \vec{X}$ is the unique string Z of length t^2 satisfying $\varphi^{rec}(t^2, Z)$.

Arguing in the conservative extension of $\mathbf{V}^0$ formed by adding the $\mathbf{\Sigma}_0^B$-definable functions above, we note that $\varphi(i, Z)$ is provably equivalent to a $\mathbf{\Sigma}_0^B$-formula (Lemma 5.40).

The graph $Y = F(\vec{x}, \vec{X})$ of F is given by the $\mathbf{\Sigma}_1^B$-formula

$$\alpha(\vec{x}, \vec{X}, Y) \equiv \exists Z < t^2,\ \varphi^{rec}(t^2, Z) \wedge Y = \mathit{Out}_{\mathsf{M}}(Z[t^2 \dot{-} t, t^2]) \qquad (8.9)$$

Now, $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC** proves $\exists! Y \alpha(\vec{x}, \vec{X}, Y)$, so F is provably total in $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**. □

Theorem 8.38. $\mathbf{TV}^0$ *proves the* $\mathbf{\Sigma}_0^B$-**BIT-REC**-*scheme.*

Proof. We use $\mathbf{\Sigma}_0^B$-**SMAX** to prove the existence of X in (8.8). Informally, imagine computing the bits $X(0), \ldots, X(y - 1)$ 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 (8.8) from bit 0, while string comparison starts with high order bits. Thus, let the string reversal function $\mathit{Rev}(y, X)$ have bit-defining axiom

$$\mathit{Rev}(y, X)(i) \leftrightarrow i < y \wedge X(y \dot{-} i \dot{-} 1)$$

where $\dot{-}$ is limited substraction (Section 3.3.3). Then $\mathit{Rev}(y, X)$ is the reverse of the string $X(0) \ldots X(y - 1)$.

Let $\varphi'(y, Y)$ be the formula

$$\forall i < y,\ Rev(y, Y)(i) \supset \varphi(i, (Rev(y, Y))^{<i}) \tag{8.10}$$

We can tacitly assume that $\varphi'(y, Y)$ is $\mathbf{\Sigma}_0^B$ (by Lemma 5.40). It is easy to see that $\varphi'(y, \varnothing)$. Thus, by $\mathbf{\Sigma}_0^B$-**SMAX**, there is a maximal string $X' \leq POW2(y)$ that satisfies (8.10). 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 (8.8). □

The previous two theorems show that all functions in **FP** are $\mathbf{\Sigma}_1^B$-definable in $\mathbf{TV}^0$. But in order to show that **VPV** is conservative over $\mathbf{TV}^0$ we must show that every function in the vocabulary $\mathcal{L}_{\mathbf{FP}}$ is $\mathbf{\Sigma}_1^B$-definable in $\mathbf{TV}^0$, and these functions were introduced via Cobham's Theorem rather than by Turing machines. Since $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC** $\subseteq \mathbf{TV}^0$, the following lemma suffices.

Lemma 8.39. $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**+**VPV** *is a conservative extension of* $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**. *Every function in* $\mathcal{L}_{\mathbf{FP}}$ *is* $\mathbf{\Sigma}_1^B$*-definable in* $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**.

Proof. The functions in $\mathcal{L}_{\mathbf{FP}}$ can be introduced successively, each one either by a $\mathbf{\Sigma}_0^B$-bit-definition or by Limited Recursion, in terms of previously defined functions. Thus $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**+**VPV** is the union of theories $\mathcal{T}_i$ satisfying

$$\mathcal{T}_0 \subset \mathcal{T}_1 \subset \mathcal{T}_2 \subset \cdots$$

where $\mathcal{T}_0$ is $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC** 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 (or f_i). We show by induction on i that each new string function F_i is $\mathbf{\Sigma}_1^B$-definable in $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC** by a formula $\alpha_F(\vec{x}, \vec{X}, Y)$ satisfying

$$Y = F_i(\vec{x}, \vec{X}) \leftrightarrow \alpha_{F_i}(\vec{x}, \vec{X}, Y) \tag{8.11}$$

Further $\alpha_{F_i}(\vec{x}, \vec{X}, Y)$ has the form

$$|Y| \leq t \wedge (\exists Z \leq t,\ \varphi_{F_i}^{rec}(t, \vec{x}, \vec{X}, Z) \wedge Y = Out_{F_i}(\vec{x}, \vec{X}, Z)) \tag{8.12}$$

where $t = t(\vec{x}, \vec{X})$ is a term and φ_{F_i} is a $\mathbf{\Sigma}_0^B$-formula and Out_{F_i} is a $\mathbf{\Sigma}_0^B$-bit-definable function. Also, $\mathcal{T}_{i-1}$ together with (8.11) proves the original defining axiom for F_i in $\mathcal{T}_i$. (Similarly for number functions f_i.)

This shows that each $\mathcal{T}_i$ is conservative over $\mathcal{T}_{i-1}$, and hence $\bigcup \mathcal{T}_i$ is conservative over $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**.

The intuitive reason that the defining formula for F_i can have the form (8.12) is that F_i is in **FP** and the proof of Theorem 8.36 shows that every such F is $\mathbf{\Sigma}_1^B$-definable by a formula of the form (8.9).

We will prove the induction step for the case that $F = F_i$ is defined by Limited Recursion, and leave the cases $F_{\varphi,t}$ and $f_{\varphi,t}$ to the reader.

Thus suppose that the defining equations for F are (8.1) and (8.2), and assume by the induction hypothesis that $G(\vec{x}, \vec{X})$ and $H(y, \vec{x}, \vec{X}, W)$ are definable by formulas of the form (8.12). Then (suppressing $\vec{x}, \vec{X}$),

$$Y = G \leftrightarrow (|Y| \leq t_G \wedge \exists Z \leq t_G,\ \varphi_G^{rec}(t_G, Z) \wedge Y = Out_G(Z)) \tag{8.13}$$

and

$$\begin{aligned} Y = H(z, V) \ \leftrightarrow\ & (|Y| \leq t_H(z, V) \wedge \\ & \exists U \leq t_H(z, V),\ \varphi_H^{rec}(t_H, z, V, U) \wedge Y = Out_H(z, V, U)) \end{aligned} \tag{8.14}$$

We compute $F(y)$ by computing the sequence

$$F(0), F(1), \cdots, F(y)$$

To do this according to our formulas for computing G and H we can compute the string W which is a concatenation of computations

$$W = (Z, F(0),\ U_0, F(1),\ U_1, F(2), \cdots, U_{y-1}, F(y))$$

where $F(0) = G$ and Z is a witness for (8.13):

$$|Z| \leq t_G \wedge Z(z) \leftrightarrow \varphi_G(z, Z^{<z}) \qquad \text{and} \qquad |F(0)| \leq t_G \wedge F(0) = Out_G(Z)$$

and for $j \geq 0$, $F(j+1) = H(j, F(j))$ and U_j witnesses (8.14):

$$\begin{aligned} |U_j| \leq t_H(j, F(j)) \wedge U_j(z) \leftrightarrow \varphi_H(z, j, F(j), U_j^{<z}) \qquad \text{and} \\ |F(j+1)| \leq t_H(j, F(j)) \wedge F(j+1) = Out_H(j, F(j), U_j) \end{aligned}$$

Observe that the above conditions for W essentially state that a bit x of W can be computed from bits $W(0), \ldots, W(x-1)$. In more detail, the substrings of W that encode Z and $F(0)$ can be defined using (8.13), and those that encode U_j and $F(j+1)$ can be defined using (8.14) from the preceeding substring that encodes $F(j)$.

For a formal argument, it is convenient to assume that each of the substrings $Z, F(0), U_0, F(1), \cdots$ of W has the same length t, by padding with 0's if necessary, for some $\mathcal{L}_A^2$ term t big enough. Also, the following abbreviations are useful in indexing a particular substring of the form $F(j)$ or U_j of W in terms of the index x for the x-th bit of W:

$$j(x) = \lfloor x/(2t) \rfloor, \qquad x' = x \bmod (2t), \qquad x'' = x' \dot{-} t$$

Each bit $W(x)$ of W can now be defined (from $W^{<x}$) by first finding the substring it belongs to (by looking at $j(x)$ and x'), and then using the one of the formulas (8.13) or (8.14) for the appropirate substrings. Details are left as an exercise. □

Exercise 8.40. *Give explicit $\mathbf{\Sigma}_0^B$ formula $\varphi_F(x, y, W)$ for F. (Note that by Lemma 5.40, it suffices to give a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FAC}^0})$ formula.) Verify that $\mathbf{V}^0$ proves the recursion equations* (8.1) *and* (8.2) *for F.*

Lemma 8.39 and Theorem 8.38 complete the proof of Theorem 8.26. These two results together with Lemma 8.32 prove the following result.

Corollary 8.41. $\mathbf{TV}^0 = \mathbf{V}^0 + \mathbf{\Sigma}_0^B$-**BIT-REC**.

The next result follows from Theorem 8.26 and Corollary 8.34.

Corollary 8.42. $\mathbf{TV}^0$ *proves its* $\mathbf{\Delta}_1^B$-**SIND** *axioms.* $\mathbf{V}^1$ *proves its* $\mathbf{\Delta}_1^B$-**SIND** *axioms.*

Proof. The first sentence follows from Theorem 8.26 and Corollary 8.34. The second sentence follows from the first, since by Corollary 8.29 any $\mathbf{\Sigma}_1^B$-formula that is $\mathbf{\Delta}_1^B$ in $\mathbf{V}^1$ is also $\mathbf{\Delta}_1^B$ in $\mathbf{TV}^0$. □

8.2.3 A Finite Axiomatization of $\mathbf{TV}^0$

In Chapter 9 we will describe a general method of defining a finitely axiomatized $\mathcal{L}_A^2$-theory for a complexity class by extending $\mathbf{V}^0$ by a single axiom asserting the existence of a computation for a problem complete for the class. Here we show how to do this for the class $\mathbf{P}$. The relevant problem is the Monotone Circuit Value Problem MCVP. The resulting theory turns out to be $\mathbf{TV}^0$.

MCVP is the problem of deciding, given a monotone Boolean circuit and its input, whether the output of the circuit is 1. The version we describe here allows $\wedge$ and $\vee$ gates to have arbitrary fan-in. Consider a layered, monotone Boolean circuit C with $(d + 1)$ layers and g 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. To encode the gates on other layers, there is a string variable G such that for $1 \le z \le d$, $G(z, x)$ 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 inputs to C are specified by a string variable I of length $|I| \le g$.

We will formalize the following polytime algorithm which computes the output of C, given inputs I. It evaluates all gates of C using $(d+1)$ loops: in loop z it identifies all gates on layer z which output 1. In particular, loop 0 is to single out 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 $\delta_{MCVP}(g, d, E, G, I, Y)$ below formalizes this algorithm. The 2–dimensional array Y is used to store the result of computation: For $1 \le z \le d$, row $Y^{[z]}$ contains the gates on layer z that output 1.

Definition 8.43. *Define $MCVP \equiv \exists Y\, \delta_{MCVP}(g,d,E,G,I,Y)$, where δ_{MCVP} is the formula*

$$\begin{aligned}&\forall x < g \forall z < d,\ (Y(0,x) \leftrightarrow I(x)) \wedge \\ &\qquad [Y(z+1,x) \leftrightarrow (G(z+1,x) \wedge \forall u < g,\ E(z,u,x) \supset Y(z,u)) \vee \\ &\qquad\qquad (\neg G(z+1,x) \wedge \exists u < g,\ E(z,u,x) \wedge Y(z,u))] \qquad (8.15)\end{aligned}$$

It is easy to see that $MCVP$ is equivalent in $\mathbf{V}^0$ to the same axiom with $|Y|$ bounded by $\langle d, g\rangle$.

Theorem 8.44. $\mathbf{TV}^0 = \mathbf{V}^0 + MCVP$.

From this and Theorem 5.75 and Corollary 8.29, since $MCVP$ is equivalent in $\mathbf{V}^0$ to a $\mathbf{\Sigma}_1^B$-formula, we have:

Corollary 8.45. $\mathbf{TV}^0$ *is finitely axiomatizable. The $\mathbf{\Sigma}_1^B$-consequences of $\mathbf{TV}^0$ and of $\mathbf{V}^1$ are each finitely axiomatizable.*

Proof of Theorem 8.44. To show $\mathbf{V}^0 + MCVP \subseteq \mathbf{TV}^0$, it suffices by Corollary 8.41 to show that $\mathbf{V}^0 + \mathbf{\Sigma}_0^B$**-BIT-REC** proves $MCVP$. The axiom $MCVP$ is almost an instance of $\mathbf{\Sigma}_0^B$**-BIT-REC**, but unfortunately using our pairing function the indices of the elements in row $z+1$ of the array Y are not all bigger than the indices of row z. To fix this, one way is to concatenate the rows of Y successively to form a string Z. Thus, since each row of Y has length $\le g$, we can define Z so that

$$Y^{[z]} = Z[zg, (z+1)g]$$

Then (8.15) can be modified to give a definition for Z, and the existence of Z follows from $\mathbf{\Sigma}_0^B$**-BIT-REC**. Finally, Y is easily defined from Z in $\mathbf{V}^0$ by $\mathbf{\Sigma}_0^B$**-COMP**.

To prove the other direction, it suffices (also by Corollary 8.41) to show that the $\mathbf{\Sigma}_0^B$**-BIT-REC** axioms are provable in $\mathbf{V}^0 + MCVP$. Thus for each $\mathbf{\Sigma}_0^B$-formula $\varphi(\vec{w}, y, X, \vec{W})$ we must show

$$\mathbf{V}^0 + MCVP \vdash \exists X \forall z < y,\ X(z) \leftrightarrow \varphi(\vec{w}, z, X^{<z}, \vec{W}) \qquad (8.16)$$

We will show that $\mathbf{V}^0$ proves the existence of a monotone circuit C that computes X by successively computing the bits $X(0), \ldots, X(y-1)$ of X, and also $\neg X(0), \ldots, \neg X(y-1)$. In order to compute $X(z)$ and $\neg X(z)$ we will use a monotone subcircuit C_z whose input array I is

$$X(0), \neg X(0), \ldots, X(z-1), \neg X(z-1) \qquad (8.17)$$

(When $z = 0$ this input array is replaced by the pair of constants $0, 1$.) The subcircuits C_z are specified by parameters g, d, E, G which depend on $\vec{w}, z, \vec{W}$, but not on X. The final row $Y^{[d]}$ in the computed values of C_z is almost the same as its input I, except the values of $X(z)$ and $\neg X(z)$ have been computed as $\varphi(z, X^{<z})$ and $\neg\varphi(z, X^{<z})$, and have been added. That is, $Y^{[d]}$ is

$$X(0), \neg X(0), \ldots, X(z-1), \neg X(z-1), \varphi(z, X^{<z}), \neg\varphi(z, X^{<z})$$

Thus the final row of C_z serves as the input row of C_{z+1}. We show that $\mathbf{V}^0$ proves the existence of g, d, E, G, I satisfying these conditions. From this we can show in $\mathbf{V}^0$ that the subcircuits C_z can be stacked one above the other to form the sequence $C_0, C_1, \ldots, C_{y-1}$ comprising the desired circuit C for computing X.

Actually the final layer of C mixes in negated values of $X(i)$:

$$X(0), \neg X(0), \ldots, X(y-1), \neg X(y-1)$$

so we need a function to extract the positive elements. Thus we define the $\mathbf{AC}^0$ string function *Ext* by

$$\mathit{Ext}(y, Z)(i) \leftrightarrow i < y \wedge Z(2i)$$

Using this we will establish (8.16) by showing

$$\mathbf{V}^0 \vdash \exists g, d, E, G, I\, \forall Y \le \langle d, g\rangle,\ \delta_{MCVP}(g, d, E, G, I, Y) \supset \\ \forall z < y(\mathit{Ext}(y, Y^{[d]})(z) \leftrightarrow \varphi(\vec{w}, z, \mathit{Ext}(y, Y^{[d]})^{<z}, \vec{W})) \qquad (8.18)$$

In constructing the subcircuits C_z we may assume that string equality $Y = Z$ has been removed from φ by using the $\mathbf{V}^0$ axiom **SE** and the equality axioms. Further we can use De Morgan's laws to push negations in so that in both φ and $\neg\varphi$ negations appear only in front of atomic formulas. We proceed to construct the subcircuits C_z by structural induction on the resulting formulas.

For the base case we consider the possible literals

$$s = t, \qquad s \ne t, \qquad s \le t, \qquad t < s, \qquad Z(t), \qquad \neg Z(t) \qquad (8.19)$$

The values of all variables except $|X|$ making up each term t are precomputed from the data $\vec{w}, z, \vec{W}$, so $t = t(|X|)$ is known as a polynomial in $|X|$ before constructing C_z. In general, the value n of a term t is represented in a row of C_z as an array T_t, which satisfies

$$T_t(i) \leftrightarrow i = n,\ 0 \le i \le b$$

for some precomputed upper bound b on t. In case t is $|X|$, this array is computed in C_z from the input (8.17) using the circuits

$$T_{|X|}(i) \equiv X(i-1) \wedge \bigwedge_{j=i}^{z-1} \neg X(j)$$

where the first term $X(i-1)$ is omitted if $i = 0$. In general the sum $s + t$ or product st of two terms is easily computed from s and t using two rows of C_z. For example

$$T_{st}(i) \equiv \bigvee_{i=jk} (T_s(j) \wedge T_t(k))$$

Using these ideas subcircuits C_z for the first four literals in (8.19) are easily constructed. The cases $Z(t)$ and $\neg Z(t)$ are no problem when Z is X, since values for $X(i)$ and $\neg X(i)$ are given as inputs (8.17) to C_z. We can simplify the cases in which Z is a parameter variable W by preprocessing φ so that any occurrence of the form $W(t)$, where t contains $|X|$, is replaced by $\exists x \le s(x = t \wedge W(x))$, where s is a term not involving $|X|$ which is an upper bound for t (and similarly for $\neg W(t)$). Thus for literals $W(t)$ and $\neg W(t)$ we may assume that t is a constant known "at compile time" and hence the truth value of $W(t)$ is known. (The truth values 0 and 1 can be computed by $(X(0) \wedge \neg X(0))$ and $(X(0) \vee \neg X(0))$, respectively.)

For the induction step, the cases φ is $\varphi_1 \wedge \varphi_2$ and φ is $\varphi_1 \vee \varphi_2$ are easy. So it remains to consider the bounded quantifier cases, say

$$\varphi(z, X) \equiv \exists x \le t \psi(x, z, X) \tag{8.20}$$

and replace $\neg\varphi$ by $\forall x \le t \neg\psi(x, z, X)$. We may assume the bounding term t in (8.20) does not contain $|X|$ by replacing t by an upper bound s for t, and adding the conjunct $x \le t$. Hence the value of t is known at compile time. By the induction hypothesis, $\mathbf{V}^0$ proves the existence of subcircuits for $\psi(x, z, X)$. A circuit for $\exists x \le t\psi(x, z, X)$ can be constructed by placing circuits for each of $\psi(0, z, X), \psi(1, z, X), \ldots, \psi(t, z, X)$ side by side so that these formulas are evaluated in parallel. (The second layer for C_z can set up the expected inputs for these circuits.) Then φ can be computed by a single $\vee$ gate from the outputs of these circuits. Similarly for the case $\forall x \le t$.

This completes the description of the subcircuits C_z. Now $\mathbf{V}^0$ proves the second line of (8.18) by induction on z, under the assumption $\delta_{MCVP}(g, d, E, G, I, Y)$, where g, d, E, G, I are defined by our construction for the circuit C. □

8.3 The Theory $\mathbf{V}^1$-HORN

Here we treat the theory $\mathbf{V}^1$**-HORN** [**?**], which is the same as $\mathbf{TV}^0$ but presented with 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 8.46. *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 $\bot$ to $\top$.

Exercise 8.47. *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" 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

$$(x^+ \leftarrow y^+) \wedge (x^+ \leftarrow z^+) \wedge (x^- \leftarrow (y^- \wedge z^-)) \tag{8.21}$$

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 $\mathbf{\Sigma}^B_1$ formula as follows.

Definition 8.48. *A* $\mathbf{\Sigma}^B_1$**-Horn** *formula is an $\mathcal{L}^2_A$-formula of the form*

$$\varphi \;\equiv\; \exists Z_1 \cdots \exists Z_k \forall y_1 \le t_1 \cdots \forall y_m \le t_m \psi \tag{8.22}$$

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 $\mathbf{\Sigma}^B_1$-Horn formulas represent polynomial time relations in their free variables.

Example 8.49 (*Parity*(X)). *This is a $\mathbf{\Sigma}_1^B$-**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.*

$$\begin{aligned}
&\exists Z_{even}\exists Z_{odd}\forall i < |X| \\
&\qquad Z_{even}(0) \wedge \neg Z_{odd}(0) \wedge Z_{odd}(|X|) \\
&\qquad \wedge (\neg Z_{even}(i+1) \vee \neg Z_{odd}(i+1)) \\
&\qquad \wedge (\neg Z_{even}(i) \vee \neg X(i) \vee Z_{odd}(i+1)) \wedge (\neg Z_{odd}(i) \vee \neg X(i) \vee Z_{even}(i+1)) \\
&\qquad \wedge (\neg Z_{even}(i) \vee X(i) \vee Z_{even}(i+1)) \wedge (\neg Z_{odd}(i) \vee X(i) \vee Z_{odd}(i+1))
\end{aligned}$$

Exercise 8.50. *Prove that Parity(X) has the stated property.*

In Section 4.3.2 we showed how the complexity classes $\mathbf{AC}^0$ and the members $\mathbf{\Sigma}_i^P$ of the polynomial hierarchy can be characterized by representation theorems involving the formula classes $\mathbf{\Sigma}_i^B$. Now we state a similar theorem characterizing $\mathbf{P}$.

Theorem 8.51 (Grädel). *A relation $R(\vec{x}, \vec{X})$ is polynomial time iff it is represented by some $\mathbf{\Sigma}_1^B$-**Horn**-formula.*

Proof sketch. $\Longleftarrow$: Suppose that the formula $\varphi(\vec{x}, \vec{X})$ has the form (8.22). 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 $\forall y_1 \le t_1 \cdots \forall y_m \le t_m$ by giving all possible m-tuples of values $(y_1, \cdots, y_m)$ satisfying the bounding terms, and form the conjunction $\Psi(Z_1, \cdots, Z_k)$ 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 $\Psi(Z_1, \cdots, Z_k)$ 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 $\{Z_i(j) \mid i \le k, j \le B\}$. Thus the problem of checking for the existence of $Z_1, \cdots, Z_k$ 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 $\mathbf{\Sigma}_1^B$-**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 8.48 and each clause specifies a local condition on the computation. □

Definition 8.52. *The theory* $\mathbf{V}^1$-**HORN** *has vocabulary* $\mathcal{L}_A^2$ *and axioms those of* $\mathbf{V}^0$ *together with* $\mathbf{\Sigma}_1^B$-**Horn**-**COMP**.

The original definition of $\mathbf{V}^1$-**HORN** in [**?**] was a little different. Recall that $\mathbf{V}^0$ has axioms 2-**BASIC** together with $\mathbf{\Sigma}_0^B$-**COMP** (Definition 5.3). The original definition was essentially $\mathbf{V}^1$-**HORN** = 2-**BASIC** + $\mathbf{\Sigma}_1^B$-**Horn-COMP**. It was shown with some effort that $\mathbf{V}^1$-**HORN** proves $\mathbf{\Sigma}_0^B$-**COMP**, so the two definitions are equivalent.

The next theorem follows from results in [**?**].

Theorem 8.53. $\mathbf{V}^1$-**HORN** $=$ $\mathbf{TV}^0$.

Proof sketch. $\mathbf{V}^1$-**HORN** $\subseteq \mathbf{TV}^0$: It suffices to show $\mathbf{TV}^0 \vdash \mathbf{\Sigma}_1^B$-**Horn-COMP**. Since **VPV** is a conservative extension of $\mathbf{TV}^0$ (Theorem 8.26), it suffices to show $\mathbf{VPV} \vdash \mathbf{\Sigma}_1^B$-**Horn-COMP**. Since $\mathbf{VPV} \vdash \mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$-**COMP** (Lemma 8.7), it suffices to show that for every $\mathbf{\Sigma}_1^B$-**Horn**-formula φ there is a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FP}})$ formula φ' such that $\mathbf{VPV} \vdash \varphi \leftrightarrow \varphi'$.

So let φ be a $\mathbf{\Sigma}_1^B$-**Horn**-formula as in (8.22), where we write $\psi(Z_1,\cdots,Z_k)$ 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

$$\varphi' \equiv \forall y_1 \leq t_1 \cdots \forall y_m \leq t_m \psi(F_1(\vec{x},\vec{X}),\cdots F_k(\vec{x},\vec{X}))$$

To define F_i we refer to the direction $\Longleftarrow$ in the proof of Theorem 8.51. 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

$$F_i(\vec{x},\vec{X})(j) \leftrightarrow j \leq B \wedge \tau(Z_i(j))$$

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. $\mathbf{VPV} \vdash \varphi \supset \varphi'$.

$\mathbf{TV}^0 \subseteq \mathbf{V}^1$-HORN: By Theorem 8.44 it suffices to show that $\mathbf{V}^1$-**HORN** $\vdash$ *MCVP*. We indicated earlier (8.21) how propositional Horn clauses can be used to evaluate circuit gates. Now we show how to use a $\mathbf{\Sigma}_1^B$-**Horn** formula to evaluate the circuit C described by parameters g, d, E, G with input I in Definition 8.43. In essence, the new atoms x^+, x^-, etc. in (8.21) are encoded by the (existentially quantified) string variables Z in the $\mathbf{\Sigma}_1^B$-**Horn** formula. Note that the algorithm outlined on page 192 is for circuits with binary gates, while here the circuit may have unbounded fan-ins.

Thus, we want to define an array $Z(z, x)$ (and its negation $\tilde{Z}(z, x)$) to evaluate gate x at layer z in C (denoted here simply by gate (z, x)). First, for the input gates we have

$$Z(0, x) \leftrightarrow I(x) \qquad \text{and} \qquad \tilde{Z}(0, x) \leftrightarrow \neg I(x) \qquad (\text{for } x < g) \tag{8.23}$$

Next, consider gate $(z+1, x)$. Suppose that this is an $\vee$-gate, i.e., $\neg G(z+1, x)$ holds. Translating the first two conjuncts in (8.21) we get:

$$(\neg G(z+1, x) \wedge E(z, y, x) \wedge Z(z, y)) \supset Z(z+1, x)$$

Translating the last clause of (8.21) is more involved, since now the gate $(z+1, x)$ may have unbounded fan-in. In fact, we formalize a simple algorithm that runs through the inputs of gate $(z+1, x)$ to check if all of them are 0. We use a string variable P — the meaning of $P(z+1, x, y)$ is that all gates (z, u) which are input to $(z+1, x)$, where $u < y$, output 0. The formalization is as follows:

$$\begin{aligned} \neg G(z+1, x) \;&\supset\; P(z+1, x, 0) \\ P(z+1, x, y) \wedge \neg E(z, y, x) \;&\supset\; P(z+1, x, y+1) \\ P(z+1, x, y) \wedge \tilde{Z}(z, y) \;&\supset\; P(z+1, x, y+1) \\ \neg G(z+1, x) \wedge P(z+1, x, g) \;&\supset\; \tilde{Z}(z+1, x) \end{aligned}$$

Let $\psi_\vee$ denote the set of the five clauses described above for the case where the gate $(z+1, x)$ is an $\vee$-gate. Also, let ψ_I be the set of clauses in (8.23). The set $\psi_\wedge$ of clauses for handling the case where $(z+1, x)$ is an $\wedge$-gate is similar to $\psi_\vee$, using an extra variable Q instead of P.

Exercise 8.54. *Give the five clauses of $\psi_\wedge$.*

Now we can show in $\mathbf{V}^0$ that a string Y that is computed by

$$\begin{aligned} Y(z_0, x_0) \leftrightarrow \exists Z \exists \tilde{Z} \exists P \exists Q \forall z < d \forall x < g \forall y < g,& \\ (\neg Z(z, x) \vee \neg \tilde{Z}(z, x)) \wedge \psi_I \wedge \psi_\wedge \wedge \psi_\vee \wedge Z(z_0, x_0)& \end{aligned} \tag{8.24}$$

(for $z_0 \leq d, x_0 < g$) satisfies $\delta_{MCVP}(g, d, E, G, I, Y)$. The following exercise is helpful.

Exercise 8.55. *Let the string variables $Z, \tilde{Z}, P, Q$ satisfy the RHS of* (8.24), *and Y' satisfy $\delta_{MCVP}(g, d, E, G, I, Y')$. Show — by (double) induction on z_0 and x_0 — that for $z_0 \le d, x_0 < g$,*

$$\neg Z(z_0, x_0) \supset \neg Y'(z_0, x_0) \qquad \textit{and} \qquad \neg\tilde{Z}(z_0, x_0) \supset Y'(z_0, x_0)$$

Exercise 8.56. *Prove by number induction that the string Y described above satisfies the recursion in $\delta_{MCVP}(g, d, E, G, I, Y)$.*

Finally, the existence of Y in $MCVP$ follows from the existence of Y that satisfies (8.24); the latter is by $\mathbf{\Sigma}_1^B$**-Horn-COMP**. Notice that although the RHS of (8.24) is a $\mathbf{\Sigma}_1^B$**-Horn** formula, to get a proper instance of $\mathbf{\Sigma}_1^B$**-Horn-COMP** we need a slight modification, i.e.,

$$\exists Y \le \langle g, d\rangle \forall i < \langle g, d\rangle,\ Y(i) \leftrightarrow \varphi(i)$$

where

$$\varphi(i) \equiv \exists Z \exists \tilde{Z} \exists P \exists Q \forall z < d \forall x < g \forall y < g \forall z_0 \le d \forall x_0 < g,$$
$$(\neg Z(z, x) \vee \neg\tilde{Z}(z, x)) \wedge \psi_I \wedge \psi_\wedge \wedge \psi_\vee \wedge (i = \langle z_0, x_0\rangle \supset Z(z_0, x_0))$$

This completes the proof that $\mathbf{TV}^0 \subseteq \mathbf{V}^1$**-HORN**. □

8.4 TV^1 and Polynomial Local Search

It follows from Theorem 8.20 that $\mathbf{V}^1 \subseteq \mathbf{TV}^1$, and hence $\mathbf{TV}^1$ can $\mathbf{\Sigma}_1^B$-define all polynomial time functions. But there is no known nice characterization of the set of *all* functions $\mathbf{\Sigma}_1^B$-definable in $\mathbf{TV}^1$. There is however a nice characterization of the set of all *search problems* $\mathbf{\Sigma}_1^B$-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 8.57. *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 8.58. *A search problem Q_{R_1} is many-one reducible to a search problem Q_{R_2}, written $Q_{R_1} \leq_p Q_{R_2}$, provided there are* **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 8.59. *Show that $\leq_p$ is a transitive relation. Also show that if $Q_{R_1} \leq_p Q_{R_2}$ 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.

Definition 8.60. *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.)

Then

$$Q(\vec{x}, \vec{X}) = \{Z \mid \varphi_Q(\vec{x}, \vec{X}, Z) \wedge N_Q(\vec{x}, \vec{X}, Z) = Z\} \tag{8.25}$$

The problem Q is an **AC**^{0}-**PLS** *problem if φ_Q, N_Q, P_Q are* **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 8.61. *An* **ITERATION** *problem* $Q = Q_F$ *is specified by a poly-time 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* $\vec{x}, \vec{X}$*):*

$$\begin{aligned}\psi_F(Z) \;\equiv\; & Z = \varnothing \wedge F(\varnothing) = \varnothing \;\vee \\ & |Z| \leq t \wedge Z < F(Z) \wedge [t < |F(Z)| \vee F(F(Z)) \leq F(Z)]\end{aligned} \tag{8.26}$$

Then

$$Q_F(\vec{x}, \vec{X}) = \{Z \mid \psi_F(\vec{x}, \vec{X}, Z)\} \tag{8.27}$$

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 8.62. *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:

$$\begin{aligned}&\varphi_Q(Z) \equiv |Z| \leq t \wedge (Z = \varnothing \vee Z < F(Z))\\ &P_Q(Z) = Z\\ &N_Q(Z) = \begin{cases} F(Z) \text{ if } |F(Z)| \leq t \text{ and } Z < F(Z) < F(F(Z)) \\ Z \text{ otherwise}\end{cases}\end{aligned}$$

Then (8.27) follows from (8.25). Notice that if Q_F is an $\mathbf{AC}^0$-**ITERATION** problem then the corresponding problem is an $\mathbf{AC}^0$-**PLS** problem. □

Theorem 8.63. *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 8.60.

We give the following $\mathbf{\Sigma}_0^B$-definition of the concatenation function $X *_z Y$, which is the first z bits of X followed by Y:

$$(X *_z Y)(i) \;\leftrightarrow\; i < z + |Y| \wedge [i < z \wedge X(i) \;\vee\; z \leq i \wedge Y(i \dot{-} z)]$$

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 $\mathcal{L}^2_A$-term large enough so that $|P_Q(N_Q(Z))| \leq u$ for $|Z| \leq t$. 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 $|F(U *_t P_Q(U))| \leq t'$.

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 (8.25), (8.27) and our definitions we see that $G(U *_t V) = N_Q(U)$ is a solution to Q.

Hence by Definition 8.58 we conclude $Q \leq_p Q_F$, 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 8.64. *If* $\mathbf{S}$ *is a set of search problems, then* $\mathbf{C}(\mathbf{S})$ *is the set of search problems many-one reducible to* $\mathbf{S}$.

Theorem 8.65.

$$\mathbf{C}(\mathbf{ITERATION}) = \mathbf{C}(\mathbf{PLS}) = \mathbf{C}(\mathbf{AC}^0\textbf{-ITERATION}) = \mathbf{C}(\mathbf{AC}^0\textbf{-PLS})$$

Proof. The first and last equalities follow from the preceding definition and theorem. The middle equality follows from these and Theorem 8.67 below. □

Definition 8.66. *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 8.67. *The following are equivalent for a search problem* Q*:*

(a) Q *is* $\mathbf{\Sigma}^B_1$*-definable in* $\mathbf{TV}^1$.
(b) Q *is in* $\mathbf{C}(\mathbf{PLS})$.
(c) Q *is in* $\mathbf{C}(\mathbf{AC}^0\textbf{-PLS})$.

Proof. (a) $\Longrightarrow$ (c) follows from Theorem 8.68 below (Witnessing for $\mathbf{TV}^1$) and Lemma 8.62. (c) $\Longrightarrow$ (b) is obvious. Hence it suffices to show (b) $\Longrightarrow$ (a).

By Theorems 8.63 and 8.16 and Corollary 8.27 it suffices to show that every problem in $\mathbf{C}(\mathbf{ITERATION})$ is $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$-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 (8.26) defining Q_F. We may assume that F is an $\mathcal{L}_{\mathbf{FP}}$-function, and hence ψ_F is a $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$-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 $\mathbf{\Sigma}_1^B$-formula (Theorem 8.16), and hence by $\mathbf{\Sigma}_1^B$-**SMAX** (Theorem 8.24), $\mathbf{TV}^1(\mathbf{VPV})$ 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 $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$-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 (supressing $\vec{x}, \vec{X}$)

$$\psi_{R_1}(Z) \equiv \exists W \le t(Z = G(W) \wedge \psi_{R_2}(\vec{f}, \vec{F}, W))$$

where t is the bounding term for Q_{R_2} and ψ_{R_2} is a $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$-formula which defines Q_{R_2} in $\mathbf{TV}^1(\mathbf{VPV})$, and $\vec{f}, \vec{F}, G$ show $Q_{R_1} \le_p Q_{R_2}$ according to Definition 8.58. Then ψ_{R_1} is equivalent to a $\mathbf{\Sigma}_1^B(\mathcal{L}_{\mathbf{FP}})$-formula, and by Definition 8.58

$$\psi_{R_1}(\vec{x}, \vec{X}, Z) \supset R_1(\vec{x}, \vec{X}, Z)$$

Since by assumption $\mathbf{TV}^1(\mathbf{VPV})$ proves $\exists W \le u\, \psi_{R_2}(W)$ (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 8.68 (Witnessing Theorem for $\mathbf{TV}^1$). *Suppose that $\varphi(\vec{x}, \vec{X}, Z)$ is a $\mathbf{\Sigma}_1^1$-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 (8.26) 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 $\mathbf{\Sigma}_0^B$. The proof is similar to the proof of the Witnessing Theorem for $\mathbf{V}^1$ (Section 6.4). 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*-$\mathbf{\Sigma}_1^B$-**SIND** Rule, defined as follows:

Definition 8.69 (The SIND Rule). *For a set Φ of formulas, the* Φ-**SIND** rule *consists of the inferences of the form*

$$\frac{\Gamma, A(\delta) \longrightarrow A(S(\delta)), \Delta}{\Gamma, A(\varnothing) \longrightarrow A(T), \Delta} \tag{8.28}$$

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 $\mathbf{LK}^2$+$\mathbf{IND}$, 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 6.4.2) and for $\mathbf{V}^0$ (Section 5.5.2), with appropriate changes.

Suppose that $\exists Z\varphi(\vec{x},\vec{X},Z)$ is a $\mathbf{\Sigma}^1_1$-theorem of $\mathbf{TV}^1$, where φ is a $\mathbf{\Sigma}^B_0$-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 $\mathbf{\Sigma}^1_1$ formulas, and in fact they are $\mathbf{\Sigma}^B_0$ formulas or *single*-$\mathbf{\Sigma}^1_1$ formulas. As a result, every sequent in π has the form

$$\mathcal{S} = \quad \underbrace{\exists X_i\theta_i(X_i)}_{i=1,\dots,m}, \Gamma \longrightarrow \Delta, \underbrace{\exists Y_j\eta_j(Y_j)}_{j=1,\dots,n} \tag{8.29}$$

for $m,n \ge 0$, where θ_i and η_j and all formulas in Γ and Δ are $\mathbf{\Sigma}^B_0$.

We will prove by induction on the depth in π of the sequent $\mathcal{S}$ that there is an $\mathbf{AC}^0$-$\mathbf{ITERATION}$ problem Q_F with graph ψ_F and for $1 \le i \le n$ there are $\mathcal{L}_{\mathbf{FAC}^0}$-functions G_i such that $\overline{\mathbf{V}}^0$ proves (the semantic equivalent of) the sequent

$$\mathcal{S}' = \quad \underbrace{\theta_i(\beta_i)}_{i=1,\dots,m}, \Gamma, \psi_F(\vec{a},\vec{\alpha},\vec{\beta},\gamma) \longrightarrow \Delta, \underbrace{\eta_j(G_j(\vec{a},\vec{\alpha},\vec{\beta},\gamma))}_{j=1,\dots,n} \tag{8.30}$$

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 $\mathbf{ITERATION}$ problem Q_F. As before, the induction step has a case for $\mathbf{\Sigma}^B_0$-$\mathbf{COMP}$ and for each rule. The argument for $\mathbf{\Sigma}^B_0$-$\mathbf{COMP}$ 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 $\mathbf{\Sigma}^B_1$-$\mathbf{SIND}$ is similar to that for $\mathbf{V}^0$ (Section 5.5.2), and can be obtained using the following lemma, that shows how two $\mathbf{ITERATION}$ problems can be combined into one.

Lemma 8.70 (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 $\mathbf{\Sigma}_0^B$-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 8.63, 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 $\vec{x}, \vec{X}$)

$$F(U{*}V{*}\delta) = \begin{cases} U * V * 2 & \text{if } \psi_{F_1}(U) \wedge \psi_{F_2}(U,V) \wedge \delta \leq 1 \\ \text{else } U * F_2(U,V) * 1 & \text{if } \psi_{F_1}(U) \wedge |V| \leq u \wedge V < F_2(U,V) \wedge \delta \leq 1 \\ \text{else } F_1(U) * \varnothing * \varnothing & \text{if } V = \delta = \varnothing \wedge |U| \leq t \wedge U < F_1(U) \\ \text{else } U * V * \delta & \end{cases}$$

Let the **ITERATION** problem Q_F have bounding term $t + u + 2$.

We claim that

$$\overline{\mathbf{V}}^0(F_1, F_2, F) \vdash \ \psi_F(U * V * \delta) \supset \delta = 2 \wedge \psi_{F_1}(U) \wedge \psi_{F_2}(U, V) \tag{8.31}$$

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 (8.26)

$$U * V * \delta < F(U * V * \delta) = F(F(U * V * \delta))$$

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 (8.31). 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

$$G_1(\vec{x}, \vec{X}, Z) = Z^{<t} \qquad G_2(\vec{x}, \vec{X}, Z) = Z[t, t+u]$$

□

It remains to handle the case in which $\mathcal{S}$ is obtained by an application of the $\mathbf{\Sigma}_1^B$-**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 $\mathbf{\Sigma}_0^B$.

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

$$\mathcal{S}_1' = \quad \Lambda', \eta_1, \psi_F(\delta,\beta,\gamma) \longrightarrow \eta_2, \Pi' \tag{8.32}$$

where

$$\eta_1 \;\equiv\; |\beta| \le r(\delta) \wedge \theta(\delta,\beta) \tag{8.33}$$
$$\eta_2 \;\equiv\; |G(\delta,\beta,\gamma)| \le r(S(\delta)) \wedge \theta(S(\delta), G(\delta,\beta,\gamma)) \tag{8.34}$$

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 $\exists X \le r(T)\theta(T,X)$, given a witness β_0 for $\exists X \le r(\varnothing)\theta(\varnothing,X)$. We want $\overline{\mathbf{V}}^0$ to prove the following sequent $\mathcal{S}'$:

$$\mathcal{S}' = \quad \Lambda', \rho_1, \psi_{F'}(\beta_0,\gamma') \longrightarrow \rho_2, \Pi'' \tag{8.35}$$

where

$$\rho_1 \;\equiv\; |\beta_0| \le r(\varnothing) \wedge \theta(\varnothing,\beta_0) \tag{8.36}$$
$$\rho_2 \;\equiv\; |G'(\beta_0,\gamma')| \le r(T) \wedge \theta(T, G'(\beta_0,\gamma')) \tag{8.37}$$

and Π'' will be given later.

We will use the technique in the proof of Lemma 8.70 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 (8.32), 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 (8.32) 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 (8.32) 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 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 (8.26) 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

$$\beta*\gamma*\delta < F'(\beta*\gamma*\delta) = F'(F'(\beta*\gamma*\delta)) \tag{8.38}$$

Analyzing the definition of F' and our assumptions ($T \neq \varnothing, \rho_1$) shows that the only way that (8.38) 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 (8.38). 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 (8.35) 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 (8.32) 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 (8.35) now follows from the **Claim** and by (8.32) 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'.) □

8.5 KPT Witnessing

8.6 $\mathbf{V}^i$ and $\mathbf{TV}^i$ for $i \geq 2$

8.7 RSUV Isomorphism

Recall the hierarchies of single-sorted theories $\mathbf{S}^i_2$ and $\mathbf{T}^i_2$ (for $i \geq 1$) from Section 3.5. In particular, $\mathbf{S}^1_2$ characterizes the class single-sorted $\mathbf{P}$ in much the same way as $\mathbf{V}^1$ characterizes the class (two-sorted) $\mathbf{P}$ (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 8.7.2 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 8.7.3 and 8.7.4. Finally we sketch a proof of the RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$.

8.7.1 The Theories $\mathbf{S}_2^i$ and $\mathbf{T}_2^i$

For this subsection it might be helpful to revisit Sections 3.1 and 3.5, and Subsection 4.3.2. 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 8.71 (Bounded Formulas of $\mathcal{L}_{\mathbf{S}_2}$). $\mathbf{\Delta}_0^b = \mathbf{\Sigma}_0^b = \mathbf{\Pi}_0^b$ *is the set of formulas whose quantifiers are sharply bounded. For* $i \geq 0$, $\mathbf{\Sigma}_{i+1}^b$ *and* $\mathbf{\Pi}_{i+1}^b$ *are the smallest sets of formulas that satisfy:*

1) $\mathbf{\Pi}_i^b \subseteq \mathbf{\Sigma}_{i+1}^b$, $\mathbf{\Sigma}_i^b \subseteq \mathbf{\Pi}_{i+1}^b$.
2) If $\varphi, \psi \in \mathbf{\Sigma}_{i+1}^b$ *(or* $\mathbf{\Pi}_{i+1}^b$*), then so are* $\varphi \wedge \psi$, $\varphi \vee \psi$.
3) If $\varphi \in \mathbf{\Sigma}_{i+1}^b$ *(resp.* $\varphi \in \mathbf{\Pi}_{i+1}^b$*), then* $\neg\varphi \in \mathbf{\Pi}_{i+1}^b$ *(resp.* $\neg\varphi \in \mathbf{\Sigma}_{i+1}^b$*).*
4) If $\varphi \in \mathbf{\Sigma}_{i+1}^b$ *(resp.* $\varphi \in \mathbf{\Pi}_{i+1}^b$*), then* $\exists x \leq t\ \varphi$ *and* $\forall x \leq |t|\ \varphi$ *are in* $\mathbf{\Sigma}_{i+1}^b$ *(resp.* $\forall x \leq t\ \varphi$ *and* $\exists x \leq |t|\ \varphi$ *are in* $\mathbf{\Pi}_{i+1}^b$*).*

Notice that different from $\mathbf{\Sigma}_i^B$ and $\mathbf{\Pi}_i^B$ (Definition 4.14), here the formulas in $\mathbf{\Sigma}_i^b$ and $\mathbf{\Pi}_i^b$ 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 $\mathbf{\Sigma}_i^P$ if and only if it is represented by a $\mathbf{\Sigma}_i^b$ formula. In particular, a single-sorted relation is in $\mathbf{NP}$ if and only if it is represented by a $\mathbf{\Sigma}_1^b$ formula. (See Definition 4.15 and the $\mathbf{\Sigma}_i^B$ and $\mathbf{\Sigma}_1^1$ Representation Theorem 4.18.)

The set **BASIC** of the defining axioms for symbols in $\mathcal{L}_{\mathbf{S}_2}$ are given in Figure 8.1. 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.

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.

1. $x \leq y \supset Sx \leq Sy$
2. $x \neq Sx$
3. $0 \leq x$
4. $(x \leq y \wedge x \neq y) \leftrightarrow Sx \leq y$
5. $x \neq 0 \supset 2 \cdot x \neq 0$
6. $x \leq y \vee y \leq x$
7. $(x \leq y \wedge y \leq x) \supset x = y$
8. $(x \leq y \wedge y \leq z) \supset x \leq z$
9. $|0| = 0$
10. $|S0| = S0$
11. $x \neq 0 \supset (|2 \cdot x| = S(|x|) \wedge$ $|S(2 \cdot x)| = S(|x|))$
12. $x \leq y \supset |x| \leq |y|$
13. $|x\#y| = S(|x| \cdot |y|)$
14. $0\#x = S0$
15. $x \neq 0 \supset (1\#(2 \cdot x) = 2 \cdot (1\#x)$ $\wedge\, 1\#S(2 \cdot x)) = 2 \cdot (1\#x))$
16. $x\#y = y\#x$
17. $|x| = |y| \supset x\#z = y\#z$
18. $|x| = |u| + |v| \supset$ $x\#y = (u\#y) \cdot (v\#y)$
19. $x \leq x + y$
20. $x \leq y \wedge x \neq y \supset$ $S(2 \cdot x) \leq 2 \cdot y \wedge S(2 \cdot x) \neq 2 \cdot y$
21. $x + y = y + y$
22. $x + 0 = x$
23. $x + Sy = S(x + y)$
24. $(x + y) + z = x + (y + z)$
25. $x + y \leq x + z \leftrightarrow y \leq z$
26. $x \cdot 0 = 0$
27. $x \cdot Sy = (x \cdot y) + x$
28. $x \cdot y = y \cdot x$
29. $x \cdot (y + z) = (x \cdot y) + (x \cdot z)$
30. $1 \leq x \supset (x \cdot y \leq x \cdot z \leftrightarrow y \leq z)$
31. $x \neq 0 \supset |x| = S(|\lfloor \frac{1}{2}x \rfloor|)$
32. $x = \lfloor \frac{1}{2}y \rfloor \leftrightarrow$ $(2 \cdot x = y \vee S(2 \cdot x) = y)$

Figure 8.1: **BASIC**

Definition 8.72 (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*

$$[\varphi(0) \wedge \forall x,\ \varphi(x) \supset \varphi(x+1)] \supset \forall z \varphi(|z|) \tag{8.39}$$

*and Φ-**PIND** is the set of formulas of the form*

$$[\varphi(0) \wedge \forall x,\ \varphi(\lfloor \tfrac{1}{2}x \rfloor) \supset \varphi(x)] \supset \forall z \varphi(z) \tag{8.40}$$

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

Definition 8.73 ($\mathbf{S}_2^i$ and $\mathbf{T}_2^i$). *For $i \geq 1$, $\mathbf{S}_2^i$ is the theory axiomatized by* **BASIC** *and $\mathbf{\Sigma}_i^b$-**PIND**; $\mathbf{T}_2^i$ is the theory axiomatized by* **BASIC** *and $\mathbf{\Sigma}_i^b$-**IND**.*

We leave as an exercise the following interesting results:

Exercise 8.74. *Show that for $i \geq 1$:*

a) $\mathbf{S}_2^i$ *can be axiomatized by* **BASIC** *together with $\mathbf{\Sigma}_i^b$-**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.

8.7.2 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 $\mathit{BIT}(i,x)$ 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 $\mathbf{S}^1_2$. It follows that $\mathbf{S}^1_2(\mathit{BIT})$ is a conservative extension of $\mathbf{S}^1_2$.

Exercise 8.75. *Show that $\mathit{BIT}(i,x)$ is definable in $\mathbf{S}^1_2$, and that*

$$\mathbf{S}^1_2(\mathit{BIT}) \vdash \forall x \forall y,\, x = y \leftrightarrow (|x| = |y| \wedge \forall i \leq |x|,\, \mathit{BIT}(i,x) \leftrightarrow \mathit{BIT}(i,y))$$

Now let $\mathcal{M}$ be a model of $\mathbf{S}^1_2$ with universe U. We can construct from $\mathcal{M}$ a two-sorted $\mathcal{L}^2_A$-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 $\mathcal{L}^2_A$ (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

$$i \in^{\mathcal{N}} x \Leftrightarrow \mathit{BIT}(i,x) \text{ holds in } \mathcal{M}, \qquad \text{for all } i \in U_1, x \in U_2$$

Definition 8.76. *For a model $\mathcal{M}$ of $\mathbf{S}^1_2$, 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 $\mathit{bin}(X)$ (see (4.4)):

$$\mathit{bin}(X) = \sum_i X(i)2^i$$

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 $\mathbf{\Sigma}^1_1$-definable in $\mathbf{V}^1$. Also, using the fact that $\mathit{BIT}(i,x)$ is definable in $\mathbf{I\Delta}_0$ (Subsection 3.3.3) and that $\mathbf{V}^0$ is a conservative extension of $\mathbf{I\Delta}_0$ (Theorem 5.9), we have $\mathit{BIT}(i,x)$ is $\mathbf{\Sigma}^B_0$-definable in $\mathbf{V}^0$:

Corollary 8.77. *The relation $\mathit{BIT}(i,x)$ is $\mathbf{\Sigma}^B_0$-definable in $\mathbf{V}^0$.*

Thus the string function $|X|_2$ whose bit-graph is

$$|X|_2(i) \leftrightarrow (i \leq |X| \wedge \mathit{BIT}(i,|X|))$$

is provably total in $\mathbf{V}^0$.

The string relation $X \leq Y$ is defined in Definition 8.21. The constant 0 is interpreted as the empty set $\varnothing$, which is defined in $\mathbf{V}^0$ by Exercise 5.43. The successor and addition functions on strings are also definable in $\mathbf{V}^0$ (Exercise 5.43). Finally, the functions $X\#Y$ and $\lfloor\frac{1}{2}X\rfloor$ can be defined in $\mathbf{V}^0$ using $\mathbf{\Sigma}_0^B$-**COMP** as follows:

$$(X\#Y)(z) \leftrightarrow z = |X| \cdot |Y|, \qquad \lfloor\tfrac{1}{2}X\rfloor(z) \leftrightarrow z \leq |X| \wedge z+1 \in X$$

Definition 8.78. *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 8.79 (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 $\mathcal{T}_1 \overset{RSUV}{\simeq} \mathcal{T}_2$) 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 $\mathbf{V}^1$. We will come back to this issue in Chapter 9.

The main result of this section is stated below.

Theorem 8.80. *For $i \geq 1$, $\mathbf{S}_2^i$ 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 8.80 (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 2.23.) 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 8.2).

In the next two subsections we define the $^\flat$ and $^\sharp$ translations. The proof of Theorem 8.80 will be given in Subsection 8.7.5.

8.7.3 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

$$\begin{array}{ccc}
\mathbf{S}_2^1 & \overset{\text{RSUV}}{\simeq} & \mathbf{V}^1 \\
\mathcal{M} & \rightharpoonup & \mathcal{M}^\sharp \\
\varphi^\flat & \leftharpoonup & \varphi \\
\hline
\mathcal{N}^\flat & \leftharpoonup & \mathcal{N} \\
\psi & \rightharpoonup & \psi^\sharp
\end{array}$$

Figure 8.2: The RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$.

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 $\mathbf{\Sigma}_1^b$-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 $\mathbf{\Sigma}_0^B$-definable (in $\mathbf{V}^0$) and $\mathbf{\Sigma}_0^B$-bit-definable. For example, the number functions $|x|$ and $x\#y$ are $\mathbf{\Sigma}_0^B$-bit-definable due to the fact that the predicate $BIT(i, x)$ is $\mathbf{\Delta}_0$-definable in $\mathbf{I\Delta}_0$ (Subsection 3.3.3). For the fact that the aforementioned multiplication functions are $\mathbf{\Sigma}_1^B$-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 8.81. $\mathbf{V}^1(\mathcal{L}^+) \vdash \mathbf{g\Sigma}_1^B(\mathcal{L}^+)$-**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 8.82. *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 8.82. 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 8.83. *Let $\psi(\vec{x},\vec{y})$ be an $\mathcal{L}_A^2$-formula.*

a) *Show that if ψ is in $\mathbf{\Sigma}_i^b$ (resp. $\mathbf{\Pi}_i^b$) 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*

$$\mathcal{N}^\flat \models \forall\vec{x}\forall\vec{y}\psi(|\vec{x}|,\vec{y}) \qquad \textit{if and only if} \qquad \mathcal{N}[\mathcal{L}^+] \models \forall\vec{x}\forall\vec{Y}\psi^\sharp(\vec{x},\vec{Y})$$

8.7.4 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 8.82).

Exercise 8.84. *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 $\mathbf{S}_2^1 \vdash t_1(|\vec{x}|,|\vec{y}|) \le |t'(\vec{x},\vec{y})|$.*

We also need the following results, which follows from the fact that BIT is $\mathbf{\Sigma}_1^b$-definable in $\mathbf{S}_2^1$.

Notation Let $\mathcal{L}_{\mathbf{S}_2}^+$ stand for $\mathcal{L}_{\mathbf{S}_2} \cup \{BIT\}$.

Exercise 8.85. *Show that* $\mathbf{S}^1_2(BIT)$ *proves both axiom schemes* $\mathbf{\Sigma}^b_1(BIT)$-**LIND** *and* $\mathbf{\Sigma}^b_1(BIT)$-**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

$$\mathcal{M}^\sharp \models \forall\vec{x}\forall\vec{Y}\varphi(\vec{x},\vec{Y}) \qquad \text{if and only if} \qquad \mathcal{M} \models \forall\vec{x}\forall\vec{y}\varphi^\flat(|\vec{x}|,\vec{y})$$

for any model $\mathcal{M}$ of $\mathbf{S}^1_2(BIT)$.

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 $\varphi^\flat \equiv (\varphi^\flat_1 \wedge \varphi^\flat_2)$ (resp. $(\varphi^\flat_1 \vee \varphi^\flat_2)$, $\neg\varphi^\flat_1$).

Next consider the case where $\varphi(\vec{x},\vec{Y}) \equiv \exists Z \le t(\vec{x},|\vec{Y}|)\,\varphi_1(\vec{x},\vec{Y},Z)$. Let $t'(\vec{x},\vec{y})$ be as in Exercise 8.84. Then

$$\varphi^\flat(\vec{x},\vec{y}) \equiv \exists z \le S0 + t'(\vec{x},\vec{y}),\ |z| \le t(\vec{x},|\vec{y}|) \wedge \varphi^\flat_1(\vec{x},\vec{y},z)$$

Now consider the case where $\varphi(\vec{x},\vec{Y}) \equiv \exists u \le t(\vec{x},|\vec{Y}|)\,\varphi_1(u,\vec{x},\vec{Y})$. Let $t'(\vec{x},\vec{y})$ be as before. Then define

$$\varphi^\flat(\vec{x},\vec{y}) \equiv \exists u \le |t'(\vec{x},\vec{y})|,\ u \le t(\vec{x},|\vec{y}|) \wedge \varphi^\flat_1(u,\vec{x},\vec{y})$$

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 8.86. *Let* $\varphi(\vec{x},\vec{Y})$ *be an* $\mathcal{L}^2_A$*-formula.*

a) *Show that if* φ *is in* $\mathbf{\Sigma}^B_i$ *(resp.* $\mathbf{\Pi}^B_i$*) for some* $i \ge 0$*, then* $\varphi^\flat(|\vec{x}|,\vec{y})$ *is in* $\mathbf{\Sigma}^b_i(BIT)$ *(resp.* $\mathbf{\Pi}^b_i(BIT)$*).*

b) *Let* $\mathcal{M}$ *be a model of* $\mathbf{S}^1_2(BIT)$*. Show that*

$$\mathcal{M}^\sharp \models \forall\vec{x}\forall\vec{Y}\varphi(\vec{x},\vec{Y}) \qquad \textit{if and only if} \qquad \mathcal{M} \models \forall\vec{x}\forall\vec{Y}\varphi^\flat(|\vec{x}|,\vec{y})$$

8.7.5 The RSUV Isomorphism between $\mathbf{S}_2^i$ and $\mathbf{V}^i$

In this subsection we discuss some issues related to the concept of RSUV isomorphism. Then 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 $i \geq 2$ 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 8.87. *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 $\mathcal{T}_1 \overset{RSUV}{\simeq} \mathcal{T}_2$.*

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 8.86 **b**, it follows that $\mathcal{M}^\sharp \models \mathcal{T}_2$. □

Exercise 8.88. *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 8.80 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 8.89. *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 $\mathbf{S}_2^1$ (Exercise 8.75).

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 8.87 above.

Corollary 8.90. *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 $\mathcal{L}_A^2$ such that $\mathcal{T}_1 \overset{RSUV}{\simeq} \mathcal{T}_2$. 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 8.86 **b**, $(\mathcal{M}^\sharp)^\flat \models \varphi^\flat$. Since $(\mathcal{M}^\sharp)^\flat$ and $\mathcal{M}$ are the same structure (Corollary 8.89), it follows that $\mathcal{M} \models \varphi^\flat$. Hence $\mathcal{T}_1 \vdash \varphi^\flat$.

(ii) is proved similarly using Exercise 8.83 **b**. □

Theorem 8.91. *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 $\mathbf{\Sigma}_1^B$-Transformation Lemma 6.25, for each $\mathcal{L}^+$ formula φ there is an $\mathcal{L}_A^2$ 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 8.90 (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 8.86 **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 $\mathbf{\Sigma}_1^B$-Transformation Lemma 6.25. (Here we do not require that $\mathcal{T}$ proves the Replacement axiom scheme.) □

Exercise 8.92. **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 $\mathbf{S}_2^1 \overset{\scriptscriptstyle RSUV}{\simeq} \mathbf{V}^1$. We need to show that $\mathbf{V}^1(\mathcal{L}^+)$ proves the $^\sharp$ translations of the axioms in **BASIC** as well as $\mathbf{\Sigma}_1^b$-**LIND** (see Exercise 8.74). The former is straightforward and is left as an exercise.

Exercise 8.93. *Show that* $\mathbf{V}^1(\mathcal{L}^+)$ *proves the* $^\sharp$ *translations of the* **BASIC** *axioms.*

Now we consider the $\mathbf{\Sigma}_1^b$-**LIND** axiom scheme. We will show that $\mathcal{N}$ satisfies the $^\sharp$ translations of the following *bounded length induction* for $\mathbf{\Sigma}_1^b$ formulas, which logically imply $\mathbf{\Sigma}_1^b$-**LIND**:

$$[\varphi(0) \wedge \forall x \leq |z|,\ \varphi(x) \supset \varphi(x+1)] \supset \forall z \varphi(|z|) \tag{8.41}$$

(where φ is a $\mathbf{\Sigma}_1^b$ formula).

Using Exercise 8.83 **a** it is easy to see that instances of (8.41) translate into $\mathbf{g\Sigma}_1^B(\mathcal{L}^+)$-**IND**. Hence the conclusion follows from Corollary 8.81.

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 $\mathbf{\Sigma}_1^B$-**IND** axioms. The latter translate into $\mathbf{\Sigma}_1^b(BIT)$-**LIND** which is provable in $\mathbf{S}_2^1(BIT)$ by Exercise 8.85. Thus the following simple exercise completes our proof of the RSUV isomorphism between $\mathbf{S}_2^1$ and $\mathbf{V}^1$. □

Exercise 8.94. *Show that* $\mathbf{S}_2^1(BIT)$ *proves the* $^\flat$ *translation of the* 2-**BASIC** *axioms.*

Exercise 8.95. *Complete the proof of Theorem 8.80 by showing that* $\mathbf{S}_2^i \overset{\scriptscriptstyle RSUV}{\simeq} \mathbf{V}^i$ *for* $i \geq 2$.

8.8 Notes

The theory **VPV** defined in Section 8.1 is based on the single-sorted equational theory **PV** [?]. The results in Section 8.1.1 were first proved in single-sorted versions in Chapter 6 of [?].

In Section 8.2 the $\mathbf{TV}^i$ hierarchy for $i \geq 1$ is the two-sorted version of Buss's [?] $\mathbf{T}_2^i$ hierarchy. The theory $\mathbf{TV}^0$ was introduced in [?] where the results of Section 8.2 are outlined, except the presentation in Section 8.2.3 is new.

The theory $\mathbf{V}^1$-**HORN** was introduced in [?], where versions of the results of Section 8.3 are proved.

The **PLS** problems were introduced in [?]. The results in Section 8.4 are mostly two-sorted versions of results from [?]. However our Witnessing Theorem 8.68 is stronger than the one in [?], 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.

Buss [?] 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 $\mathbf{U}_2^1$

and $\mathbf{V}_2^1$ capture **PSPACE** and **EXPTIME**, respectively. (The theories $\mathbf{V}^i$ in this book is sometimes called $\mathbf{V}_1^i$.) The equivalence between $\mathbf{S}_{k+1}^i$ and $\mathbf{V}_k^i$ was first realized in [**?**, **?**]. The name "RSUV isomorphism" was introduced by Takeuti in [**?**], where he also introduced the hierarchies $\mathbf{R}_k$, and proved the equivalences between $\mathbf{R}_{k+1}^i$ and $\mathbf{U}_k^i$ and between $\mathbf{S}_{k+1}^i$ and $\mathbf{V}_k^i$. The $\mathbf{S} - \mathbf{V}$ equivalence was also proved in [**?**]. The syntactic translations $^\flat$ and $^\sharp$ are called *interpretations* in [**?**, **?**] (the symbols $^\flat$ and $^\sharp$ were introduced in [**?**]).

Chapter 9

Theories for Small Classes

In this chapter we present subtheories of $\mathbf{TV}^0$ which are associated with several subclasses of $\mathbf{P}$. These classes include the sequence

$$\mathbf{AC}^0(p) \subsetneq \mathbf{TC}^0 \subseteq \mathbf{NC}^1 \subseteq \mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{NC} \subseteq \mathbf{P}$$

We present a generic method for developing the theories. As an accompanying example, we will treat in detail the theory $\mathbf{VTC}^0$, which is associated with the class $\mathbf{TC}^0$. In general, each class $\mathbf{C}$ has a complete (under $\mathbf{AC}^0$ reduction) problem P. The associated theory $\mathbf{VC}$ is axiomatized by $\mathbf{V}^0$ and an axiom that encodes a suitable algorithm that solves P. These theories are finitely axiomatizable, because $\mathbf{V}^0$ is. To show that $\mathbf{VC}$ characterizes the corresponding class, we introduce a universal theory $\overline{\mathbf{VC}}$ in the style of $\overline{\mathbf{V}}^0$ (Section 5.6) and show that it is a conservative extension of $\mathbf{VC}$. As for $\overline{\mathbf{I\Delta}}_0$ and $\overline{\mathbf{V}}^0$, the idea is to introduce function symbols for all functions in $\mathbf{FC}$ (see Definition 5.16). The main task is to show that these functions are provably total in $\mathbf{VC}$.

Our universal theories are "minimal" theories for the corresponding complexity classes in the sense that the axioms consist of straightforward definitions for the functions and predicates in the classes. For example, $\overline{\mathbf{VTC}}^0$ satisfies this condition, and since it is a conservative extension of $\mathbf{VTC}^0$, the latter is also a minimal theory for $\mathbf{TC}^0$.

The chapter is organized as follows. First, we define the notion of $\mathbf{AC}^0$ reduction. Then, in Section 9.2, we present the theory $\mathbf{VTC}^0$ and its universal counterpart $\overline{\mathbf{VTC}}^0$, and show that $\overline{\mathbf{VTC}}^0$ is a conservative extension of $\mathbf{VTC}^0$. This proves the Definability Theorem for $\mathbf{VTC}^0$ (Corollary 9.10). We will in fact prove general results which lead to establishing theories for other classes. These theories will be presented in the following sections. Finally, in Section 9.7 we prove interesting recursion-theoretic characterization of several subclasses of $\mathbf{FL}$ using the number recursion scheme.

9.1 $\mathbf{AC}^0$ Reductions

The classes that we consider in this chapter are nice in the sense that they are closed under $\mathbf{AC}^0$ reductions in the Turing style (as opposed to the more restricted many-one style). We will formalize this notion. The idea is (see for example [**?**]) that 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. We will also 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 $\mathbf{\Sigma}_0^B$-definable from $\mathcal{L}$ if it is polynomially bounded, and its bit graph (resp. graph) is represented by a $\mathbf{\Sigma}_0^B(\mathcal{L})$ formula (Definition 5.37). The following definition generalizes the notion of $\mathbf{\Sigma}_0^B$-definability.

Definition 9.1 ($\mathbf{AC}^0$ Reductions). *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*

$$F_i \textit{ is } \mathbf{\Sigma}_0^B\textit{-definable from } \mathcal{L} \cup \{F_1, \ldots, F_{i-1}\}, \textit{ for } i = 1, \ldots, n; \tag{9.1}$$

and that F (resp. f) is $\mathbf{\Sigma}_0^B$-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 $\mathbf{\Sigma}_0^B(\mathcal{L} \cup \{F_1, \ldots, F_n\})$ formula.

If in the above definition $\mathcal{L}$ consists only of functions in $\mathbf{FAC}^0$, then a single interation $(n = 1)$ 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 $\mathbf{TC}^0$. As far as we know there is no bound on the number of iterations needed, because (as far as we know) there is no fixed d such that every member of $\mathbf{TC}^0$ can be defined by a polynomial size family of circuits of depth d.

Definition 9.2 ($\mathbf{FAC}^0$ 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 $\mathbf{\Sigma}_0^B$-definability. For the case of $\mathbf{FAC}^0$, this follows from Corollary 5.41.

Corollary 9.3. *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 5.16). The following lemma is straightforward consequence of the definitions involved.

Lemma 9.4. *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. 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.

Definition 9.5 (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*

$$F(y) = \{f(x) : x \le y\}$$

Note that if f is polynomially bounded, then so is F.

For example, recall that the $\mathbf{\Sigma}_0^B$ formula $\varphi_{parity}(X, Y)$ (5.32) asserts that for $0 \le i < |X|$, 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:

$$f(x, X) = \begin{cases} x & \text{if } x > 0 \text{ and the number of 1 bits in } X(0), \ldots, X(x-1) \text{ is odd} \\ 0 & \text{otherwise} \end{cases}$$

Theorem 9.6. *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 $\mathbf{\Sigma}_0^B$-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

$$F(y)(z) \leftrightarrow z < t \wedge \exists x \le y \; z = f(x)$$

where t is the bounding term for F. Hence F is also $\mathbf{\Sigma}_0^B$-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 $\mathbf{\Sigma}_0^B$-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 $\mathbf{\Sigma}_0^B(\mathcal{L})$ 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 $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-formula ψ, and (by structural induction on φ) it is clear that for every $\mathbf{\Sigma}_0^B(\mathcal{L})$-formula $\varphi(\vec{z}, \vec{Z})$ there is a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$-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 $\mathbf{\Sigma}_0^B$-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 $\mathbf{\Sigma}_0^B(\mathcal{L})$ 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

$$F(\vec{z}, \vec{X}) = \mathit{Cut}(t, G(t, \vec{z}, \vec{X}))$$

where $G(y, \vec{z}, \vec{X})$ is the string comprehension of $f(x, \vec{z}, \vec{X})$, and Cut (see (6.5) on page 127) is the $\mathbf{FAC}^0$ function defined by

$$\mathit{Cut}(x, X)(z) \leftrightarrow z < x \wedge X(z)$$

It remains to show that if a number function f is $\mathbf{\Sigma}_0^B$-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 $\mathbf{\Sigma}_0^B(\mathcal{L})$ 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 $g(x, \vec{z}, \vec{X})$. □

9.2 The Theory $\mathbf{VTC}^0$

The class nonuniform $\mathbf{TC}^0$ (or $\mathbf{TC}^0/poly$) is defined similarly to nonuniform $\mathbf{AC}^0$ (Section 4.1), but now the circuits may contain *majority* gates, i.e., gates with unbounded fan-in, which output 1 if and only if the number of 1 inputs is more than the number of 0 inputs. Thus, a language is in nonuniform $\mathbf{TC}^0$ if it is accepted by a family of polynomial size, bounded depth circuits $\langle C_n \rangle$ of this type. In the uniform version, the circuits are specified in a uniform way. We use **FO** uniformity, i.e., $\langle C_n \rangle$ is required to be in **FO**, so $\mathbf{TC}^0$ refers to **FO**-uniform $\mathbf{TC}^0$.

In this section we develop the theory $\mathbf{VTC}^0$, whose $\mathbf{\Sigma}_1^1$-definable functions are precisely functions in $\mathbf{FTC}^0$. $\mathbf{VTC}^0$ is obtained from $\mathbf{V}^0$ by adding a formula that formalizes a polytime computation of the counting function *numones*, where $numones(x, X)$ is the number of elements of X that are $< x$ (page 136). (Intuitively, majority gates can be equivalently replaced by counting gates, i.e., gates which output the number of 1 inputs.)

In order to show that the provably total functions of $\mathbf{VTC}^0$ are precisely the $\mathbf{TC}^0$ functions, we introduce the universal theory $\overline{\mathbf{VTC}}^0$, whose language $\mathcal{L}_{\mathbf{FTC}^0}$ consists of all $\mathbf{TC}^0$ functions. The theory $\overline{\mathbf{VTC}}^0$ is developed in the style of $\overline{\mathbf{V}}^0$ (Section 5.6), and the main task here is to show that $\overline{\mathbf{VTC}}^0$ is a conservative extension of $\mathbf{VTC}^0$ (see the alternative proof of the Witnessing Theorem for $\mathbf{V}^0$ in Subsection 5.6.1). Because of the additional function *numones*, proving this conservativity turns out to be more involved than proving the conservativity of $\overline{\mathbf{V}}^0$ over $\mathbf{V}^0$ (Theorem 5.71). We will prove the conservativity of this type in a more general setting; the general conservativity result proved here enables us to develop theories for a number of other classes in the subsequent sections. At the end of this section, we will present a proof of the Pigeonhole Principle in $\mathbf{VTC}^0$.

In Subsection 9.7.2, we will provide another characterization of $\mathbf{FTC}^0$ using the *number summation* operation. It is possible to develop an universal theory over $\mathcal{L}_{\mathbf{FTC}^0}$ using this operation in the style of **VPV** and use this universal theory to obtain the desired characterization of $\mathbf{TC}^0$ by $\mathbf{VTC}^0$. It is also possible to show that this universal theory is equivalent to $\overline{\mathbf{VTC}}^0$. We will not go into further detail here. [1]

[1]OR SHOULD WE ?

9.2.1 $\mathbf{TC}^0$ and $\mathbf{VTC}^0$

As for $\mathbf{AC}^0$ (review Section 4.1), there are several equivalent ways of defining uniform $\mathbf{TC}^0$. The descriptive characterization of $\mathbf{TC}^0$ is $\mathbf{FO}(M)$ (i.e., **FO** augmented with the majority quantifier), or $\mathbf{FO}(COUNT)$ (i.e., **FO** with the counting quantifier). Here we use the characterization of $\mathbf{TC}^0$ which is based on the notion of $\mathbf{AC}^0$-reducibility defined in Section 9.1. In particular, we use the fact that the function $numones(x, X)$ (page 136) is complete for the class $\mathbf{TC}^0$.

Definition 9.7. *$\mathbf{TC}^0$ is the $\mathbf{AC}^0$ closure of numones. $\mathbf{FTC}^0$ is the $\mathbf{FAC}^0$ closure of numones.*

The theory $\mathbf{VTC}^0$ is axiomatized by $\mathbf{V}^0$ together with the axiom *NUMONES*, which is essentially a $\mathbf{\Sigma}_1^B$ defining axiom for $numones(x, X)$ (see formula (6.14) on page 136).

Below we abbreviate part of the $\mathbf{\Sigma}_1^1$-defining axiom for *numones* (6.14) by introducing the $\mathbf{\Sigma}_0^B$ formula $\delta_{NUM}(x, X, Y)$, which states that Y is a "counting array" for X, i.e., for each $z \le x$, $Y(z, y)$ holds if and only if $numones(z, X) = y$. Recall that $(Y)^z$ denotes $seq(z, Y)$, the z-th element of the bounded sequence of numbers coded by Y (Definition 5.55).

$$\begin{aligned}\delta_{NUM}(x, X, Y) \equiv\ & (Y)^0 = 0 \wedge \\ & \forall z < x,\ (X(z) \supset (Y)^{z+1} = (Y)^z + 1) \wedge (\neg X(z) \supset (Y)^{z+1} = (Y)^z) \end{aligned} \quad (9.2)$$

Definition 9.8 ($\mathbf{VTC}^0$). *Let NUMONES denote $\forall X \forall x \exists Y \delta_{NUM}(x, X, Y)$. The theory $\mathbf{VTC}^0$ is axiomatized by $\mathbf{V}^0$ and NUMONES.*

In $\mathbf{V}^0$, *NUMONES* is equivalent to the same axiom with $\exists Y$ replaced by the bounded quantifier $\exists Y \le 1 + \langle x, x \rangle$. Hence, $\mathbf{VTC}^0$ is a polynomial-bounded theory.

The following fact are easily verified.

Proposition 9.9. *The function numones is provably total in $\mathbf{VTC}^0$. $\mathbf{VTC}^0 \subseteq \mathbf{TV}^0$. $\mathbf{VTC}^0$ is finitely axiomatizable.*

Our goal for the rest of this section is to prove the following theorem:

Theorem 9.10 (Definability Theorem for $\mathbf{VTC}^0$). **a)** *The $\mathbf{\Sigma}_1^1$-definable (and $\mathbf{\Sigma}_1^B$-definable) functions in $\mathbf{VTC}^0$ are precisely those in $\mathbf{FTC}^0$.*

b) *The $\mathbf{\Delta}_1^1$-definable (and $\mathbf{\Delta}_1^B$-definable) predicates in $\mathbf{VTC}^0$ are precisely those in $\mathbf{TC}^0$.*

Corollary 9.11. *$\mathbf{VTC}^0$ is a proper extension of $\mathbf{V}^0$. In fact, $\mathbf{VTC}^0$ is not $\mathbf{\Sigma}_0^B$-conservative over $\mathbf{V}^0$.*

Proof. The first part follows from the theorem and the fact that the number function $parity(X)$, which is the parity of the total number of elements in X (Subsection 5.5.1), is not in $\mathbf{FAC}^0$, but *parity* is in $\mathbf{FTC}^0$, since it can be easily computed using *numones*. The second part of the theorem holds because $\mathbf{VTC}^0$ proves the Pigeonhole Principle (Subsection 9.2.6 below), while this principle is not provable in $\mathbf{V}^0$ (Corollary 7.21). □

Outline of the Proof of the Definability Theorem for $\mathbf{VTC}^0$. For part **a**, the statement for $\mathbf{\Sigma}_1^B$-definable functions follows from that of $\mathbf{\Sigma}_1^1$-definable functions and Parikh's Theorem (see Corollary 5.29). Part **b** of the theorem follows from **a** and Theorem 5.59. The proof of part **a** spans over Subsections 9.2.2 – 9.2.5 (Corollaries 9.24 and 9.25). □

9.2.2 The Theory $\overline{\mathbf{VTC}}^0$

For this subsection, it is useful to review the theory $\overline{\mathbf{V}}^0$ introduced in Section 5.6. The theory $\overline{\mathbf{VTC}}^0$ is defined similarly to $\overline{\mathbf{V}}^0$, with the addition of *numones*. We first specify the vocabulary $\mathcal{L}_{\mathbf{FTC}^0}$ of $\overline{\mathbf{VTC}}^0$. The purpose is that the symbols in $\mathcal{L}_{\mathbf{FTC}^0}$ represent precisely the functions of $\mathbf{FTC}^0$ and that every function of $\mathbf{FTC}^0$ has a quantifier-free defining axiom. Consider the following quantifier-free defining axioms for *numones*:

$$numones(0, X) = 0 \tag{9.3}$$
$$X(z) \supset numones(z + 1, X) = numones(z, X) + 1 \tag{9.4}$$
$$\neg X(z) \supset numones(z + 1, X) = numones(z, X). \tag{9.5}$$

The language $\mathcal{L}_{\mathbf{FTC}^0}$ is defined in the same way as $\mathcal{L}_{\mathbf{FAC}^0}$ (Definition 5.67). Recall the definitions of pd, $f_{\mathbf{SE}}$, $F_{\varphi,t}$ and $f_{\varphi,t}$ from Section 5.6.

Definition 9.12. $\mathcal{L}_{\mathbf{FTC}^0}$ *is the smallest set that satisfies*

1) $\mathcal{L}_{\mathbf{FTC}^0}$ *includes* $\mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}, numones\}$
2) *For each open formula* $\varphi(z, \vec{x}, \vec{X})$ *over* $\mathcal{L}_{\mathbf{FTC}^0}$ *and term* $t = t(\vec{x}, \vec{X})$ *of* $\mathcal{L}_A^2$, *there is a string function* $F_{\varphi,t}$ *and a number function* $f_{\varphi,t}$ *in* $\mathcal{L}_{\mathbf{FTC}^0}$.

Definition 9.13. $\overline{\mathbf{VTC}}^0$ *is the theory over* $\mathcal{L}_{\mathbf{FTC}^0}$ *with the following quantifier-free axioms:* **B1**–**B11**, **L1**, **L2** *(Figure 5.1),* **B12**$'$, **B12**$''$ (5.40), **SE**$'$ (5.41), *the defining axioms* (9.3), (9.4) *and* (9.5) *for numones, and* (5.37) *for each function* $F_{\varphi,t}$ *and* (5.38) *for each function* $f_{\varphi,t}$ *of* $\mathcal{L}_{\mathbf{FTC}^0}$.

Lemma 9.14. **a)** *For every* $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FTC}^0})$ *formula* φ *there is an open formula* ψ *of* $\mathcal{L}_{\mathbf{FTC}^0}$ *such that* $\overline{\mathbf{VTC}}^0$ *proves* $(\varphi \leftrightarrow \psi)$.

b) *The functions in* $\mathcal{L}_{\mathbf{FTC}^0}$ *represent precisely* $\mathbf{FTC}^0$. *A relation is in* $\mathbf{TC}^0$ *if and only if it is represented by some open* $\mathcal{L}_{\mathbf{FTC}^0}$ *formula.*

c) $\overline{\mathbf{VTC}}^0$ *proves the* $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FTC}^0})$-**COMP** *axiom scheme.*

Proof. Part **a** is proved in the same way as Lemma 3.44. (See also Lemma 5.69.)

Part **b** follows from **a** and Definition 9.7.

For **c**, let $\varphi(z, \vec{x}, \vec{Y})$ be a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FTC}^0})$ formula. By **a**, there is a quantifier-free formula ψ so that $\overline{\mathbf{VTC}}^0 \vdash \varphi(z, \vec{x}, \vec{Y}) \leftrightarrow \psi(z, \vec{x}, \vec{Y})$. The function $F_{\psi,y}$ (5.37) satisfies

$$\forall z < y\ F_{\psi,y}(\vec{x}, \vec{Y})(z) \leftrightarrow \psi(z, \vec{x}, \vec{Y})$$

Therefore, the string X in the comprehension axiom (5.1) for φ can be taken to be $F_{\psi,y}$. In other words, $\overline{\mathbf{VTC}}^0$ proves the comprehension axiom for φ. □

Corollary 9.15. $\overline{\mathbf{VTC}}^0$ *extends* $\mathbf{VTC}^0$.

Proof. Lemma 9.14 **c** shows that $\overline{\mathbf{VTC}}^0$ extends $\mathbf{V}^0$. By Lemma 5.49, $\overline{\mathbf{VTC}}^0$ proves Multiple Comprehension for $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FTC}^0})$ formulas. Using this axiom scheme, $\overline{\mathbf{VTC}}^0$ proves

$$\exists Y \le (\langle x, x\rangle + 1)\, \forall z \le x \forall y \le x\, (Y(z, y) \leftrightarrow y = \mathit{numones}(z, X))$$

From the defining axioms (9.3), (9.4) and (9.5) for *numones*, we can show that this Y satisfies $\delta_{NUM}(x, X, Y)$ (9.2). Consequently, $\overline{\mathbf{VTC}}^0$ proves *NUMONES*.
□

9.2.3 Aggregate Functions and Conservative Extensions

Now we set out to prove that $\overline{\mathbf{VTC}}^0$ is conservative over $\mathbf{VTC}^0$. Recall the similar result that $\overline{\mathbf{V}}^0$ is conservative over $\mathbf{V}^0$ (Theorem 5.71), whose proof relies essentially on the fact that every $\mathcal{L}_{\mathbf{FAC}^0}$ function has a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ defining axiom (Lemma 5.40). Unfortunately, the analog (i.e., every $\mathcal{L}_{\mathbf{FTC}^0}$ function has a $\mathbf{\Sigma}_0^B(\mathit{numones})$ defining axiom) does not appear to hold. This is because, in general, an open formula of $\mathcal{L}_{\mathbf{FTC}^0}$ is not equivalent to a $\mathbf{\Sigma}_0^B(\mathit{numones})$ formula, for the same reason that a $\mathbf{TC}^0$ circuit involving nested threshold gates cannot be made polynomially equivalent to a circuit with unnested threshold gates. Hence we must work harder to prove that $\overline{\mathbf{VTC}}^0$ is conservative over $\mathbf{VTC}^0$.

Notice that $\overline{\mathbf{VTC}}^0$ is defined inductively, and since $\overline{\mathbf{VTC}}^0$ extends $\mathbf{VTC}^0$, the starting point can be taken to be $\mathbf{VTC}^0$. In other words, $\overline{\mathbf{VTC}}^0$ is obtained from $\mathbf{VTC}^0$ by a series of extensions. Our goal is to show that each successive extension is conservative over the preceeding one. It will then follow from Corollary 3.31 that $\overline{\mathbf{VTC}}^0$ is a conservative extension of $\mathbf{VTC}^0$.

More formally, the inductive definition of $\mathcal{L}_{\mathbf{FTC}^0}$ (Definition 9.12) gives rise to a sequence of vocabularies: $\mathcal{L}_0 = \mathcal{L}_A^2$, $\mathcal{L}_1 = \mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}, \mathit{numones}\}$ and each $\mathcal{L}_{n+1}$ (for $n \ge 1$) extends $\mathcal{L}_n$ by either $f_{\varphi,t}$ or $F_{\varphi,t}$, where φ is an open formula over $\mathcal{L}_n$, and t is a $\mathcal{L}_A^2$ term. Let $\mathbf{VTC}^0(\mathcal{L}_n)$ be the extension of $\mathbf{VTC}^0$ obtained by adding the functions in $\mathcal{L}_n$ and their defining axioms. We will show that $\mathbf{VTC}^0(\mathcal{L}_{n+1})$ is conservative over $\mathbf{VTC}^0(\mathcal{L}_n)$ by showing that the new function in $\mathcal{L}_{n+1}$ is provably total in $\mathbf{VTC}^0(\mathcal{L}_n)$. Since this new function is already $\mathbf{\Sigma}_0^B$-definable from $\mathcal{L}_n$ (see Definition 5.37) — in fact, it is definable from $\mathcal{L}_n$ by a quantifier-free formula — it suffices to show that

$$\mathbf{VTC}^0(\mathcal{L}_n) \vdash \mathbf{\Sigma}_0^B(\mathcal{L}_n)\textbf{-COMP} \tag{9.6}$$

(see Corollary 5.38 and Corollary 5.39).

We will prove (9.6) by induction on n. It turns out that we need a slightly stronger induction hypothesis, which is stated using the notion of *aggregate functions* defined below. Informally, for a string funtion F (or a number function f), the aggregate function of F (resp. f), denoted by $F^\star$ (resp. $f^\star$), is the string function that gathers different values of F (resp. f). Recall the functions *Row* and *seq* from Section 5.4 (Definition 5.50 and Definition 5.55).

Definition 9.16 (Aggregate Function). *Suppose that $F(x_1, \ldots, x_k, X_1, \ldots, X_n)$ is a polynomially bounded string function, i.e., for some $\mathcal{L}_A^2$ term t,*

$$|F(\vec{x}, \vec{X})| \le t(\vec{x}, |\vec{X}|)$$

Then $F^\star(b, Z_1, \ldots, Z_k, X_1, \ldots, X_n)$ is the polynomially bounded string function that satisfies

$$|F^\star(b, \vec{Z}, \vec{X})| \le \langle b, t(|\vec{Z}|, |\vec{X}|)\rangle$$

and

$$F^\star(b, \vec{Z}, \vec{X})(w) \leftrightarrow \exists u < b \exists v < w,\ w = \langle u, v\rangle \wedge F((Z_1)^u, \ldots, (Z_k)^u, X_1^{[u]}, \ldots, X_n^{[u]})(v) \tag{9.7}$$

Similarly, suppose that $f(x_1, \ldots, x_k, X_1, \ldots, X_n)$ is a polynomially bounded number function, i.e., for some $\mathcal{L}_A^2$ term t,

$$f(\vec{x}, \vec{X}) \le t(\vec{x}, |\vec{X}|)$$

Then $f^\star(b, \vec{Z}, \vec{X})$ is the polynomially bounded string *function that satisfies*

$$|f^\star(b, \vec{Z}, \vec{X})| \le \langle b, 1 + t\rangle$$

and

$$f^\star(b, \vec{Z}, \vec{X})(w) \leftrightarrow \exists u < b,\ w = \langle u, f((Z_1)^u, \ldots, (Z_k)^u, X_1^{[u]}, \ldots, X_n^{[u]})\rangle \tag{9.8}$$

Notice that, by (9.7), for $u < b$,

$$F^\star(b, \vec{Z}, \vec{X})^{[u]} = F((Z_1)^u, \ldots, (Z_k)^u, X_1^{[u]}, \ldots, X_n^{[u]})$$

Also, by (9.8), for $u < b$,

$$(f^\star(b, \vec{Z}, \vec{X}))^u = f((Z_1)^u, \ldots, (Z_k)^u, X_1^{[u]}, \ldots, X_n^{[u]})$$

Example 9.17 (*numones**)**.**

$$\begin{aligned}&numones^\star(b,Z,X)=Y \ \leftrightarrow\ (|Y|\le\langle b,1+|X|\rangle\wedge\\&\quad\forall w<\langle b,1+|X|\rangle, Y(w)\leftrightarrow\exists u<b,\ w=\langle u, numones((Z)^u,X^{[u]})\rangle) \qquad (9.9)\end{aligned}$$

In Lemma 9.23, we will show that numones* *is provably total in* $\mathbf{VTC}^0$.

The next lemma strengthens Corollary 5.38 when the underlying vocabulary contains *Row* and *seq* (see also the Extension by Bit-Definition Lemma 5.35). Its proof is left as an exercise.

Lemma 9.18. *Suppose that the vocabulary $\mathcal{L}$ contains Row and seq, and that $\mathcal{T}$ is a theory over $\mathcal{L}$ that extends $\mathbf{V}^0$ and proves the $\mathbf{\Sigma}_0^B(\mathcal{L})$-***COMP** *axiom scheme. Suppose that a function F (or f) is $\mathbf{\Sigma}_0^B$-definable from $\mathcal{L}$. Then the function $F^\star$ (or $f^\star$) is $\mathbf{\Sigma}_0^B(\mathcal{L})$-definable from $\mathcal{L}$. In addition, $F^\star$ (resp. $f^\star$) is $\mathbf{\Sigma}_0^B(\mathcal{L})$-definable, and hence provably total, in $\mathcal{T}$.*

Exercise 9.19. *Prove the lemma.*

Our goal now is to prove Theorem 9.21 below, which is important in the proof (by induction) of (9.6). First we prove the next result, which gives sufficient conditions for $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP** to continue to hold in a theory after the theory is extended by a $\mathbf{\Sigma}_1^1$-definable function. Instead of $\mathbf{\Sigma}_1^1$-definable, we state this theorem for the more general notion of *definable* function (Definition 5.27).

Theorem 9.20. *Let $\mathcal{T}$ be an extension of $\mathbf{V}^0$ with vocabulary $\mathcal{L}$, where $\mathcal{L}$ contains the functions Row and seq. Suppose that $\mathcal{T}$ proves $\mathbf{\Sigma}_0^B(\mathcal{L})$-***COMP***. Let F be a definable string function of $\mathcal{T}$ such that the function $F^\star$ is also definable in $\mathcal{T}$. Then $\mathcal{T}(F)$ proves $\mathbf{\Sigma}_0^B(\mathcal{L}\cup\{F\})$-***COMP***. The same is true for a number function f definable in $\mathcal{T}$ for which $f^\star$ is definable in $\mathcal{T}$.*

Proof. We will consider the case of extending $\mathcal{L}$ by a string function F. The case where $\mathcal{L}$ is extended by a number function is handled similarly by using number variables w_i instead of the string variables W_i in the argument below.

First, since $\mathcal{T}$ proves $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP**, by Lemma 5.49 it proves the Multiple Comprehension axioms for $\mathbf{\Sigma}_0^B(\mathcal{L})$ formulas.

Claim For any $\mathcal{L}$-terms $\vec{s},\vec{T}$ that contain variables $\vec{z}$, $\mathcal{T}(F)$ proves

$$\exists Y\forall z_1<b_1\dots\forall z_m<b_m Y^{[\vec{z}]}=F(\vec{s},\vec{T}) \qquad (9.10)$$

Proof of the Claim. Since $\mathcal{T}$ proves the Multiple Comprehension axiom scheme for $\mathbf{\Sigma}_0^B(\mathcal{L})$ 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 existtence of Z_i such that $(Z)^{\langle\vec{z}\rangle}=s_i$, for $1\le i\le k$. Now the value of Y that satisfies (9.10) is just $F^\star(\langle\vec{b}\rangle,\vec{Z},\vec{X})$. $\square$

Let $\mathcal{L}' = \mathcal{L} \cup \{F\}$. We show by induction on the quantifier depth of a $\mathbf{\Sigma}_0^B(\mathcal{L}')$ formula ψ that $\mathcal{T}(F)$ proves

$$\exists Z \le \langle b_1, \dots, b_m \rangle \forall z_1 < b_1 \dots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \psi(\vec{z}) \tag{9.11}$$

where $\vec{z}$ are all free number variables of ψ. It follows that $\mathcal{T}(F) \vdash \mathbf{\Sigma}_0^B(\mathcal{L}')$-**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 $\mathbf{\Sigma}_0^B(\mathcal{L})$, 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 $\vec{s}_1, \vec{T}_1$ 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 claim above, for $1 \le i \le k$, $\mathcal{T}(F)$ proves the existence of W_i such that

$$\forall z_1 < b_1 \dots \forall z_m < b_m,\ W_i^{[\vec{z}]} = F(\overrightarrow{s'_i}, \overrightarrow{T'_i}) \tag{9.12}$$

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, by Multiple Comprehension for $\mathbf{\Sigma}_0^B(\mathcal{L})$ and the fact that $\mathcal{L}$ contains *Row*,

$$\mathcal{T} \vdash \exists Z \le \langle b_1, \dots, b_m \rangle \forall z_1 < b_1 \dots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \psi'(\vec{z}, W_1, \dots, W_k).$$

Such Z satisfies (9.11) when each W_i is defined by (9.12).

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, by Lemma 5.49

$$\mathbf{V}^0 \vdash \exists Z \forall z_1 < b_1 \dots \forall z_m < b_m,\ Z(\vec{z}) \leftrightarrow \forall x < t Z'(\vec{z}, x).$$

Thus $\mathcal{T}(F) \vdash \exists Z \forall \vec{z} < \vec{b}\, Z(\vec{z}) \leftrightarrow \psi(\vec{z})$. $\square$

Now we extend the above theorem for the case where the new functions F/f are actually provably total (i.e., $\mathbf{\Sigma}_1^1$-definable) in $\mathcal{T}$. The most important application of this strengthening is the proof of (9.6), which, as discussed before, implies that $\mathbf{VTC}^0$ is conservative over $\mathbf{VTC}^0$.

Theorem 9.21. *Let $\mathcal{T}$ be an extension of $\mathbf{V}^0$ with the vocabulary $\mathcal{L}$, where $\mathcal{L}$ contains Row and seq. Suppose that $\mathcal{T}$ and $\mathcal{L}$ satisfy*

a) *$\mathcal{T}$ proves the $\mathbf{\Sigma}_0^B(\mathcal{L})$-**COMP**, and*

b) *For each $\mathbf{\Sigma}_0^B(\mathcal{L})$ formula θ there is a $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$ formula η such that $\mathcal{T} \vdash \theta \leftrightarrow \eta$.*

Let F (or f) be a provably total string function in $\mathcal{T}$ such that the function $F^\star$ (or $f^\star$) is also provably total in $\mathcal{T}$. Then **a** *and* **b** *hold with $\mathcal{T}(F)$ (resp. $\mathcal{T}(f)$) replacing $\mathcal{T}$, and $\mathcal{L} \cup \{F\}$ (resp. $\mathcal{L} \cup \{f\}$) replacing $\mathcal{L}$.*

In our applications of this theorem, the theory $\mathcal{T}$ is always polynomial-bounded; therefore, the formula η in **b** can be taken to be $\mathbf{\Sigma}_1^B(\mathcal{L}_A^2)$ (as opposed to $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$). Furthermore, $\overline{\mathbf{VTC}}^0$, as well as most of our other universal theories, are obtained by inductively adding new functions F_{n+1} or f_{n+1}, which are $\mathbf{\Sigma}_0^B$-definable from the current language $\mathcal{L}_n$. Corollary 5.38 and Lemma 9.18 now become handy, since they show that F and $F^\star$ (resp. f and $f^\star$) are provably total in $\mathbf{VTC}^0(\mathcal{L}_n)$.

Proof of Theorem 9.21. We prove for the case of the string function F. The case for the number function f is similar. First, $\mathcal{T}(F)$ and $\mathcal{L} \cup \{F\}$ satisfy **a** by Theorem 9.20. We show that they also satisfy **b**. Suppose that

$$\theta \equiv \mathrm{Q}_1 z_1 < r_1 \ldots \mathrm{Q}_n z_n < r_n \psi(\vec{z})$$

is a $\mathbf{\Sigma}_0^B(\mathcal{L} \cup \{F\})$ formula, where ψ is quantifier-free. Let $\vec{s}_i, \vec{T}_i, \overrightarrow{s_i'}, \overrightarrow{T_i'}$ and $\psi'(\vec{z}, W_1, \ldots, W_k)$ be as described in the proof of Theorem 9.20. Define

$$\theta'(W_1, ..., W_k) \equiv \mathrm{Q}_1 z_1 < r_1 \ldots \mathrm{Q}_n z_n < r_n \psi'(\vec{z}, W_1, ..., W_k).$$

For $1 \le i \le k$ let λ_i be the formula

$$\forall z_1 < b_1 \ldots \forall z_m < b_m \varphi(\overrightarrow{s_i'}, \overrightarrow{T_i'}, W_i^{[\vec{z}]})$$

Then, θ is equivalent in $\mathcal{T}(F)$ to

$$\exists W_1 \ldots \exists W_k,\ ((\bigwedge \lambda_i) \wedge \theta'(W_1, \ldots, W_k))$$

By property **b** for $\mathcal{T}$ and $\mathcal{L}$, we may replace the first conjunct in the scope of the string quantifiers by a $\mathbf{\Sigma}_1^1$ formula, and thus obtain the required $\mathbf{\Sigma}_1^1$ formula η in **b** for $\mathcal{T}(F)$ and $\mathcal{L} \cup \{F\}$. □

The next corollary summarizes our discussion so far.

Corollary 9.22. *Suppose that $\mathcal{T}_0$ and $\mathcal{L}_0$ satisfy the hypotheses* **a** *and* **b** *of Theorem 9.21. 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 the defining axiom for a function F or f that is $\mathbf{\Sigma}_0^B$-definable from $\mathcal{L}_i$, the vocabulary of $\mathcal{T}_i$. Let*

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

and let $\mathcal{L}_\infty$ be the vocabulary of $\mathcal{T}_\infty$. Then $\mathcal{T}_\infty$ is a conservative extension of $\mathcal{T}_0$, and the additional functions in $\mathcal{T}_\infty$ are $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$-definable in $\mathcal{T}_0$. Also $\mathcal{T}_\infty$ and $\mathcal{L}_\infty$ satisfy the hypotheses **a** *and* **b** *of Theorem 9.21.*

Proof. First, by the hypothesis that $\mathcal{T}_0$ proves $\mathbf{\Sigma}_0^B(\mathcal{L}_0)$-**COMP**, it is easy to prove by induction on i, using Corollary 5.38, Lemma 9.18 and Theorem 9.21, that $\mathcal{T}_i$ and $\mathcal{L}_i$ satisfy the properties **a**, **b** of Theorem 9.21.

It then follows from Corollary 5.38 that the new function in $\mathcal{T}_{i+1}$ is provably total in $\mathcal{T}_i$, and therefore $\mathcal{T}_{i+1}$ is a conservative extension of $\mathcal{T}_i$. As a result, $\mathcal{T}_\infty$ is conservative over $\mathcal{T}_0$ (Corollary 3.31).

Finally, the graph of each function in $\mathcal{T}_\infty$ has a $\mathbf{\Sigma}_0^B(\mathcal{L}_i)$ definition for some i. This definition is provably equivalent in $\mathcal{T}_i$ to a $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$ formula, by the property stated in **b** of Theorem 9.21. Consequently, every function in $\mathcal{T}_\infty$ is $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$-definable in $\mathcal{T}_0$, since $\mathcal{T}_i$ is conservative over $\mathcal{T}_0$. □

9.2.4 The Conservativity of $\overline{\mathbf{VTC}}^0$ over $\mathbf{VTC}^0$

The theory $\overline{\mathbf{VTC}}^0$ is obtained from $\mathcal{T}_0 = \mathbf{VTC}^0(\mathit{numones}, \mathit{Row}, \mathit{seq})$ by a series of extension just as described in Corollary 9.22. The final step of proving the conservativity of $\overline{\mathbf{VTC}}^0$ over $\mathbf{VTC}^0$ is to show that $\mathcal{T}_0$ satisfies the hypotheses **a** and **b** of Theorem 9.21. For this we apply the same theorem for $\mathcal{T} = \mathbf{VTC}^0(\mathit{Row}, \mathit{seq}, \mathit{left}, \mathit{right})$. We need the following lemma.

Lemma 9.23. *The function $\mathit{numones}^\star$ is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{VTC}^0(\mathit{Row}, \mathit{seq})$.*

Proof. It suffices to show that for any number term $t(u)$ over $\mathcal{L}_A^2 \cup \{\mathit{Row}, \mathit{seq}\}$,

$$\mathbf{VTC}^0(\mathit{Row}, \mathit{seq}) \vdash \exists Y \forall u < b\ \delta_{NUM}(t(u), X^{[u]}, Y^{[u]})$$

To prove this, the idea is to construct Y using $\mathbf{\Sigma}_0^B(\mathit{Row})$-**COMP** from the counting array Y' for a "big" string X', where X' is the concatenation of the initial segments of the rows $X^{[0]}, \ldots, X^{[b-1]}$ of X. Formally, let s be an $\mathcal{L}_A^2$ number term that dominates $t(u)$, for all $u < b$. Let X' be defined by [2]

$$X'(us + z) \leftrightarrow z < t(u) \wedge X^{[u]}(z), \quad \text{for } z < s, u < b.$$

In other words, the bit string

$$X'(us)\ \ldots\ X'(us + t(u) - 1)$$

is a copy of

$$X^{[u]}(0)\ \ldots\ X^{[u]}(t(u) - 1)$$

and $X'(us + t(u)), \ldots, X'((u+1)s - 1)$ are all 0. Therefore, for $z \leq t(u)$,

$$\mathit{numones}(z, X^{[u]}) = \mathit{numones}(us + z, X') - \mathit{numones}(us, X').$$

[2]USE SUBSTRING FUNCTION FROM CHAPTER 8 HERE

Let Y' be the counting array for X', i.e., $Y'(z,y) \leftrightarrow \mathit{numones}(z,X') = y$. Then,

$$Y^{[u]}(z,y) \leftrightarrow y + \mathit{numones}(us,X') = \mathit{numones}(us+z,X')$$

Hence,

$$Y^{[u]}(z,y) \leftrightarrow \exists y_1, y_2 \le |X'|,\ Y'(us,y_1) \wedge Y'(us+z,y_2) \wedge y + y_1 = y_2$$

Consequently, Y exists in $\mathbf{V}^0$ by $\mathbf{\Sigma}_0^B$ Multiple Comprehension. $\square$

The complete proof of the conservativity of $\overline{\mathbf{VTC}}^0$ over $\mathbf{VTC}^0$ is presented in the next corollary.

Corollary 9.24. **a)** *$\overline{\mathbf{VTC}}^0$ is a conservative extension of $\mathbf{VTC}^0$.*
b) *Every function in $\mathcal{L}_{\mathbf{FTC}^0}$ is $\mathbf{\Sigma}_1^1(\mathcal{L}_A^2)$-definable in $\mathbf{VTC}^0$.*

Proof. Let $\mathcal{T}_0 = \mathbf{VTC}^0(\mathit{Row}, \mathit{seq}, \mathit{numones})$. It suffices to show that $\mathcal{T}_0$ satisfies the hypotheses **a**, **b** of Theorem 9.21. Then $\overline{\mathbf{VTC}}^0$ is the theory $\mathcal{T}_\infty$ obtained from $\mathcal{T}_0$ as in Corollary 9.22. Consequently, both parts **a** and **b** of Corollary 9.24 follow from Corollary 9.22 and the fact that $\mathcal{T}_0$ is a conservative extension of $\mathbf{VTC}^0$ (because the functions *Row*, *seq* and *numones* are definable in $\mathbf{VTC}^0$).

Now we show that $\mathcal{T}_0$ satisfies the hypotheses **a** and **b** of Theorem 9.21. Let $\mathcal{T} = \mathbf{VTC}^0(\mathit{Row}, \mathit{seq})$. Then $\mathcal{T}_0 = \mathcal{T}(\mathit{numones})$. Since *Row* and *seq* are $\mathbf{AC}^0$ functions and $\mathcal{T}$ extends $\mathbf{V}^0$, $\mathcal{T}$ satisfies hypotheses **a** and **b** of Theorem 9.21 (see the $\mathbf{FAC}^0$ Elimination Lemma 5.73). Also, it follows from Lemma 9.23 that $\mathit{numones}^\star$ (9.9) is $\mathbf{\Sigma}_1^1$-definable in $\mathcal{T}$. Thus, by Theorem 9.21, $\mathcal{T}_0$ also satisfies the hypotheses **a**, **b** there. $\square$

The above corollary proves one direction of (part **a** of) the Definability Theorem for $\mathbf{VTC}^0$ 9.10. The other direction is the Witnessing Theorem for $\mathbf{VTC}^0$, which is stated and proved in the next Subsection.

9.2.5 The Witnessing Theorem for $\mathbf{VTC}^0$

Recall that each string function $F \in \mathbf{FTC}^0$ has a defining axiom according to our construction of $\mathcal{L}_{\mathbf{FTC}^0}$ (see Definition 9.12 and Lemma 9.14 **b**). In fact, there is a finite sequence of $\mathbf{FTC}^0$ functions $F_1 = \mathit{Row}, \dots, F_n$ that are involved in defining F. Let $\mathcal{L}_F$ denote this sequence of functions (including F). Similar to Corollary 9.24, for each $F \in \mathbf{FTC}^0$, $\overline{\mathbf{VTC}}^0$ is a conservative extension of the theory $\mathbf{VTC}^0(\mathcal{L}_F)$.

Corollary 9.25 (Witnessing Theorems for $\mathbf{VTC}^0$). *For each theorem $\exists \vec{Y} \varphi(\vec{x}, \vec{X}, \vec{Y})$ of $\mathbf{VTC}^0$, where φ is $\mathbf{\Sigma}_0^B$, there is a string function $\vec{F} \in \mathbf{FTC}^0$ such that*

$$\mathbf{VTC}^0(\mathcal{L}_F) \vdash \varphi(\vec{x}, \vec{X}, \vec{F}(\vec{x}, \vec{X})).$$

Proof. This is an application of the Herbrand Theorem 4.32, using the fact that φ is equivalent in $\overline{\mathbf{V}}^0$ to a quantifier-free $\mathcal{L}_{\mathbf{FAC}^0}$ formula (Lemma 5.69), and that the symbols in $\mathcal{L}_{\mathbf{FTC}^0}$ represent precisely the functions of $\mathbf{FTC}^0$ (Lemma 9.14 **b**). The proof is similar to the proof of the second proof of Parikh's Theorem (page 52). $\square$

9.2.6 Proving the Pigeonhole Principle in $\mathbf{VTC}^0$

In this subsection we present a proof of the Pigeonhole Principle (Subsection 7.1.2) in $\mathbf{VTC}^0$. From this and the independence of PHP from $\mathbf{V}^0$ (Corollary 7.21), it follows that $\mathbf{VTC}^0$ is a proper extension of $\mathbf{V}^0$ (see the proof of Corollary 9.11). (Although this also follows from the fact that *numones* is not an $\mathbf{AC}^0$ function.) In the next chapter we will show that each $\mathbf{\Sigma}_0^B$ theorem of $\mathbf{VTC}^0$ translates into a family of tautologies having polysize $\mathbf{TC}^0$**-Frege** proofs. It will follow that the family **PHP** (Definition 7.12) has polysize $\mathbf{TC}^0$**-Frege** proofs. This separates **bPK** from $\mathbf{TC}^0$**-Frege**. On the other hand, we will show (Subsection 9.5.3) that $\mathbf{VNC}^1$ extends $\mathbf{VTC}^0$. Therefore PHP is 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. This subsection is independent of the remaining of this chapter.

Recall (Example 7.18) that $\mathbf{PHP}(a, X)$ is the following formula

$$\forall x \le a \exists y < a X(x,y) \supset \exists x \le a \exists z \le a \exists y < a (x \ne z \wedge X(x,y) \wedge X(z,y))$$

Theorem 9.26. $\mathbf{VTC}^0 \vdash \mathbf{PHP}(a, X)$.

Proof. We will actually prove that $\mathbf{VTC}^0 \vdash \mathbf{PHP'}(a, X)$, where $\mathbf{PHP'}(a, X)$ is the formula

$$\forall x \le a \exists y < a X(y,x) \supset \exists x \le a \exists z \le a \exists y < a (x \ne z \wedge X(y,x) \wedge X(y,z)) \quad (9.13)$$

This will show that $\mathbf{VTC}^0 \vdash \mathbf{PHP}(a, X)$, since $\mathbf{PHP}(a, X)$ is just $\mathbf{PHP'}(a, X^t)$, where X^t is the "transpose" of X:

$$X^t(y,z) \leftrightarrow X(z,y) \qquad \text{for } y < a, z \le a$$

Intuitively, the premise of (9.13) implies that the total number of elements in all a rows of X is at least $(a+1)$, because the union of these rows contains all number that are $\le a$. We show that there is a row of X that has at least two elements, which implies the conclusion of (9.13). Below, we will formalize the argument that if every row of X contains at most one element, then the total number of elements in all a rows X is at most a, which is a contradiction. We need the following functions, all except the last one are in fact in $\mathbf{FAC}^0$:

- Union: $X \cup Y$

$$(X \cup Y)(z) \leftrightarrow z < |X| + |Y| \wedge (X(z) \vee Y(z))$$

- Multiple Union: We interpret X as an array of a rows. Then $\mathit{MultUnion}(a, X)$ is the union of the rows $X^{[x]}$, for $x < a$:

$$\mathit{MultUnion}(a, X)(z) \leftrightarrow z < |X| \wedge \exists x < aX^{[x]}(z)$$

- Concatenation: Suppose that X codes an array of a rows. Then the function $\mathit{Concat}(a, b, X)$ is the concatenation of the initial segments bounded by b of the rows $X^{[x]}$ of X, for $x < a$:

$$\mathit{Concat}(a, b, X)(bx + y) \leftrightarrow x < a \wedge y < b \wedge X^{[x]}(y)$$

- Total Number of Bits in an Array: Again, X is viewed as coding an array of a rows. Then $\mathit{totNum}(a, b, X)$ is the total number of elements of the initial segments (bounded by b) of the rows $X^{[x]}$ of X, for $x < a$:

$$\mathit{totNum}(a, b, X) = \mathit{numones}(ab, \mathit{Concat}(a, b, X))$$

Exercise 9.27. *Show that the following are theorems of* $\overline{\mathbf{VTC}}^0$*:*

a) $\mathit{numones}(b, X \cup Y) \leq \mathit{numones}(b, X) + \mathit{numones}(b, Y)$.

b) $\mathit{totNum}(a + 1, b, X) = \mathit{totNum}(a, b, X) + \mathit{numones}(b, X^{[a]})$.

c) $\mathit{numones}(b, \mathit{MultUnion}(a, X)) \leq \mathit{totNum}(a, b, X)$.

d) $\forall x < a \ \mathit{numones}(b, X^{[x]}) \leq u \ \supset \ \mathit{totNum}(a, b, X) \leq au$.

Now the total number of elements that are $\leq a$ in all a rows of X is $\mathit{totNum}(a, a + 1, X)$. Suppose, by way of contradiction, that

$$\forall x < a \ \mathit{numones}(a + 1, X^{[x]}) \leq 1$$

Then, by **d** of the exercise above, we have

$$\mathit{totNum}(a, a + 1, X) \leq a$$

It follows from **c** that

$$\mathit{numones}(a + 1, \mathit{MultUnion}(a, X)) \leq a$$

However, it is obvious that $\forall z \leq a \ \mathit{MultUnion}(a, X)(z)$. By a simple induction argument, this implies

$$\mathit{numones}(a + 1, \mathit{MultUnion}(a, X)) = a + 1$$

a contradiction. □

9.3 Theories for Other Subclasses of P

In this section, we show how to develop finitely axiomatizable theories for a number of other uniform subclasses of **P** in the style of $\mathbf{VTC}^0$. Consider a polytime function F, and let **C** be the class of two-sorted relations which are $\mathbf{AC}^0$-reducible to F. The class **FC** (Definition 5.16) can be equivalently defined as the $\mathbf{FAC}^0$ closure of F (Definition 9.2). In the case of $\mathbf{TC}^0$, F is essentially the string function computing the "counting array" Y in (9.2) (page 222).

The theory **VC** defined in this section is similar to $\mathbf{VTC}^0$ in the sense that **VC** is axiomatized by $\mathbf{V}^0$ together with a single axiom $AXIOM_F$ that formalizes a polytime algorithm that computes F. To show that the functions in **FC** are precisely the provably total functions of **VC** we will proceed just as in Section 9.2: here we will introduce the universal theory $\overline{\mathbf{VC}}$, whose vocabulary consists precisely of functions of **FC**, and show that $\overline{\mathbf{VC}}$ is a conservative extension of **VC**.

As in the case of $\mathbf{VTC}^0$ and $\overline{\mathbf{VTC}}^0$, the main task here is to show that $\overline{\mathbf{VC}}$ is conservative over **C**. We will use the results from Subsection 9.2.3. In particular, we will need the aggregate function $F^\star$ of F to be provably total in **VC**. Thus, in general, $AXIOM_F$ is indeed a defining axiom for $F^\star$ instead of F.

Another instance of **VC** is the finite axiomatization of $\mathbf{TV}^0$ presented in Subsection 8.2.3, where $\mathbf{TV}^0$ is shown to be equal to $\mathbf{V}^0 + MCVP$. In this case, the function F can be viewed as the string function that evaluates the gate values of a monotone circuit on a given input (a complete problem for **P**). Notice that $MCVP$ defines just F, but it possible to define the function $F^\star$ in $\mathbf{V}^0 + MCVP$. The same is true for $\mathbf{VTC}^0$ — we show in Lemma 9.23 that $numones^\star$ is provably total in $\mathbf{VTC}^0$. In fact, for each class that we consider in the following sections, the additional axiom is essentially a defining axiom for F, and thus is simpler than the axiom $AXIOM_F$ defined in this section. In each case, we are able to show that $F^\star$ is definable in the corresponding theory. Each proof is, however, rather *ad hoc*.

9.3.1 The Theories VC and $\overline{\mathbf{VC}}$

The quantifier-free defining axiom for F is obtained from Cobham's recursion theoretic characterization of the polytime functions (Theorem 6.16). The proof of that theorem actually shows that each polytime function can be obtained from $\mathbf{AC}^0$ functions by composition and *at most* one application of the limited recursion operation (Definition 6.15). In each complexity class of interest it turns out that a suitable function F complete for the class can be defined by such a recursion of the form (e.g., $Conf_{\mathsf{M}}$ in the proof of Theorem 6.16)

$$F(0,X) = Cut(t(0,|X|), Init(X)) \tag{9.14}$$

$$F(x+1,X) = Cut(t(x+1,|X|), Next(x,X,F(x,X))) \tag{9.15}$$

where Cut (6.5), $Init(X)$ and $Next(x,X,Y)$ are $\mathbf{AC}^0$ functions, $t(x,y)$ is a polynomial. (For example, $F(x,X)$ is the configuration at time x of a polytime

Turing machine that solves some complete problem for a given class.) Notice that the above defining axioms for F are quantifier-free in $\mathcal{L}_{\mathbf{FAC}^0}$.

Now $\mathcal{L}_{\mathbf{FC}}$ and $\overline{\mathbf{VC}}$ are defined in the same way as $\mathcal{L}_{\mathbf{FTC}^0}$ and $\overline{\mathbf{VTC}}^0$ (Definition 9.12 and Definition 9.13). Recall the definitions of $F_{\varphi,t}$ and $f_{\varphi,t}$ from Section 5.6.

Definition 9.28 ($\mathcal{L}_{\mathbf{FC}}$ and $\overline{\mathbf{VC}}$). *The language $\mathcal{L}_{\mathbf{FC}}$ is the smallest set that satisfies*

1) *$\mathcal{L}_{\mathbf{FC}}$ includes $\mathcal{L}_{\mathbf{FAC}^0} \cup \{F\}$*
2) *For each open formula $\varphi(z, \vec{x}, \vec{X})$ over $\mathcal{L}_{\mathbf{FC}}$ and term $t = t(\vec{x}, \vec{X})$ of $\mathcal{L}_A^2$, there is a string function $F_{\varphi,t}$ and a number function $f_{\varphi,t}$ in $\mathcal{L}_{\mathbf{FC}}$.*

The theory $\overline{\mathbf{VC}}$ is the extension of $\mathbf{V}^0$ where the additional axioms include: the defining axioms (9.14), (9.15) *for F, and* (5.37)/(5.38) *for each (new) function $F_{\varphi,t}/f_{\varphi,t}$ of $\mathcal{L}_{\mathbf{FC}}$.*

The next lemma is analogous to Lemma 9.14, and its proof is left as an exercise.

Lemma 9.29. **a)** *For every $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FC}})$ formula φ there is a quantifier-free formula ψ of $\mathcal{L}_{\mathbf{FC}}$ such that $\overline{\mathbf{VC}} \vdash \varphi \leftrightarrow \psi$.*

b) *The functions in $\mathcal{L}_{\mathbf{FC}}$ represent precisely* **FC**. *A relation is in* **C** *if and only if it is represented by some open $\mathcal{L}_{\mathbf{FC}}$ formula.*

c) *$\overline{\mathbf{VC}}$ proves the $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FC}})$-**COMP** axiom scheme.*

Exercise 9.30. *Prove the lemma.*

Now we define $AXIOM_F$. This axiom specifies the (polytime) computation of multiple (i.e., polynomially many) values of $F(x, X)$. In particular, let $\delta_F(a, b, X, Y)$ be the formula stating that Y encodes simultaneously the b recursive computations of $F(a, X^{[0]}), \ldots, F(a, X^{[b-1]})$: $Y^{[u,x]} = F(x, X^{[u]})$ for all $x \le a, u < b$. More precisely,

$$\delta_F(a, b, X, Y) \equiv \forall u < b,\ Y^{[u,0]} = \mathit{Cut}(t(0, |X^{[u]}|), \mathit{Init}(X^{[u]})) \wedge \\ \forall x < a,\ Y^{[u,x+1]} = \mathit{Cut}(t(x+1, |X^{[u]}|), \mathit{Next}(x, X^{[u]}, Y^{[u,x]})) \quad (9.16)$$

Here we do not introduce new functions *Cut*, *Init*, *Next*, but tacitly use their $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ bit definitions instead (see the $\mathbf{FAC}^0$ Elimination Lemma 5.73).

Definition 9.31. *Let $AXIOM_F$ be $\forall a \forall b \forall X \exists Y\, \delta_F(a, b, X, Y)$. The theory* **VC** *has vocabulary $\mathcal{L}_A^2$ and is axiomatized by $\mathbf{V}^0$ and the axiom $AXIOM_F$.*

Again, $AXIOM_F$ is equivalent in $\mathbf{V}^0$ to the same axiom with $|Y|$ bounded by $\langle b, t(a, |X|)\rangle$. Also, since $\mathbf{V}^0$ is finitely axiomatizable, so is **VC**. The following proposition is immediate from definition.

Proposition 9.32. **a)** $\mathbf{VC} \subseteq \mathbf{TV}^0$.

b) *The function $F^\star$ is $\mathbf{\Sigma}^1_1$-definable in* **VC**.

Corollary 9.33. **a)** *$\overline{\mathbf{VC}}$ is a conservative extension of* **VC**.

b) *Every function in $\mathcal{L}_{\mathbf{FC}}$ is $\mathbf{\Sigma}^1_1(\mathcal{L}^2_A)$ definable in* **VC**.

Proof. First we show that $\overline{\mathbf{VC}}$ extends **VC**. Since $\overline{\mathbf{VC}}$ proves $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FC}})$-**COMP** (Lemma 9.29 **c**), it also proves the Multiple Comprehension for $\mathbf{\Sigma}^B_0(\mathcal{L}_{\mathbf{FC}})$ formulas (Lemma 5.49). Hence $\overline{\mathbf{VC}}$ proves $AXIOM_F$, i.e., $\overline{\mathbf{VC}}$ extends **VC**.

The remaining parts of the corollary are proved in almost the same way as Corollary 9.24. First, we apply Theorem 9.21, using the fact that $F^\star$ is $\mathbf{\Sigma}^1_1$-definable in $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$ (Proposition 9.32 above). This shows that $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0}\cup\{F\})$ is conservative over $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$ and satisfies the hypotheses of Theorem 9.21. Next, we apply Corollary 9.22 for $\mathcal{T}_0 = \mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0}\cup\{F\})$ and $\mathcal{T}_\infty = \overline{\mathbf{VC}}$. It follows from both steps that $\overline{\mathbf{VC}}$ is conservative over $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$ and every function in $\mathcal{L}_{\mathbf{FC}}$ is $\mathbf{\Sigma}^1_1(\mathcal{L}^2_A)$ definable in $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$. Finally, the conclusions follow from the fact that $\mathbf{VC}(\mathcal{L}_{\mathbf{FAC}^0})$ is conservative over **VC**. □

Similar to the Witnessing Theorem for $\mathbf{VTC}^0$ 9.25 we have:

Corollary 9.34 (Witnessing Theorem for $\overline{\mathbf{VC}}$). *For each theorem $\exists\vec{Z}\varphi(\vec{a},\vec{\alpha},\vec{Z})$ of $\overline{\mathbf{VC}}$, where φ is a $\mathbf{\Sigma}^B_0$ formula, there are functions $\vec{F}$ of $\mathcal{L}_{\mathbf{FC}}$ such that*

$$\overline{\mathbf{VC}} \vdash \forall\vec{x}\forall\vec{X}\ \varphi(\vec{x},\vec{X},\vec{F}(\vec{x},\vec{X})).$$

We summarize the characterization of **C** by **VC** in the next corollary.

Corollary 9.35 (Definability Theorem for VC). **a)** *The $\mathbf{\Sigma}^1_1$-definable (and $\mathbf{\Sigma}^B_1$-definable) functions in* **VC** *are precisely those in $\mathcal{L}_{\mathbf{FC}}$.*

b) *The $\mathbf{\Delta}^1_1$-definable (and $\mathbf{\Delta}^B_1$-definable) predicates in* **VC** *are precisely those in* **C**.

9.3.2 The $\mathbf{\Sigma}^B_0$ Replacement Rule and Axiom in VC

In this subsection we discuss the $\mathbf{\Sigma}^B_0$ replacement rule. In general, if a theory $\mathcal{T}$ admits this rule, then the string functions $f^\star/F^\star$ are provably total in $\mathcal{T}$, given that f/F are provably total in $\mathcal{T}$.

Definition 9.36 (Replacement Rule). *Suppose that $\mathcal{T}$ is a theory over $\mathcal{L}$, where either $\mathcal{L} = \mathcal{L}^2_A$ or $\mathcal{L}^2_A \cup \{Row\} \subseteq \mathcal{L}$. Let Φ be a set of $\mathcal{L}$-formulas. Then $\mathcal{T}$ is said to* admit the Φ replacement rule *if whenever $\mathcal{T}$ proves*

$$\forall z < b\exists Y\ \varphi(z,Y) \tag{9.17}$$

for a formula $\varphi \in \Phi$, φ may contain other free variables, then $\mathcal{T}$ also proves

$$\exists Y\forall z < b\ \varphi(z,Y^{[z]}) \tag{9.18}$$

Note that $\mathcal{T}$ admits the Φ replacement rule whenever it proves the Φ-**REPL** axiom scheme (Definition 6.18). In general the converse is not true. For example, $\mathbf{V}^0$ admits the $\mathbf{\Sigma}_0^B$ replacement rule as shown in the lemma below, but it does not proves $\mathbf{\Sigma}_0^B$-**REPL**.

We are mainly interested in the case where $\Phi = \mathbf{\Sigma}_0^B(\mathcal{L})$. Then a for theory $\mathcal{T}$ which extends $\mathbf{V}^0$, "$\mathcal{T}$ admits the $\mathbf{\Sigma}_0^B(\mathcal{L})$ replacement rule" implies that $f^\star/F^\star$ are also $\mathbf{\Sigma}_1^1$-definable in $\mathcal{T}$, for $\mathbf{\Sigma}_1^1$-definable functions f/F. The converse is true when $\mathcal{T}$ is a universal theory. We prove this for $\overline{\mathbf{VC}}$. It will follow also that **VC** admits the $\mathbf{\Sigma}_0^B$ replacement rule.

Lemma 9.37. **a)** *The theory* $\overline{\mathbf{VC}}$ *admits the* $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FC}})$ *replacement rule.*
b) **VC** *admits the* $\mathbf{\Sigma}_0^B$ *replacement rule.*

Proof. Part **b** follows from **a** and the fact that $\overline{\mathbf{VC}}$ is a conservative extension of **VC**. We prove **a**. Suppose that $\overline{\mathbf{VC}}$ proves (9.17), where φ is a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FC}})$ formula. Then by the Witnessing Theorem for $\overline{\mathbf{VC}}$ 9.34, there is a function $F(z)$ of $\mathcal{L}_{\mathbf{FC}}$ such that

$$\overline{\mathbf{VC}} \vdash \varphi(z, F(z))$$

The existence of Y in (9.18) is witnessed by the function $G(b)$, where $G(b)^{[z]} = F(z)$, for $z < b$. The function $G(b)$ has the $\mathbf{\Sigma}_0^B$-bit-definition:

$$G(b)(z,u) \leftrightarrow z < b \wedge u < t(z) \wedge F(z)(u)$$

where t is a bounding term for $F(z)$. Thus $\overline{\mathbf{VC}}$ also proves (9.18). □

9.4 Theories for $\mathbf{AC}^0(m)$ 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.

In descriptive complexity, uniform $\mathbf{AC}^0(m)$ (or just $\mathbf{AC}^0(m)$) can be characterized using the *mod*(m) quantifier [**?**]. Here we define $\mathbf{AC}^0(m)$ using the number function $mod_m(x, X)$, where

$$mod_m(x,X) = numones(x,X) \mod m$$

Definition 9.38. $\mathbf{AC}^0(m)$ *is the class of relations that are* $\mathbf{AC}^0$*-reducible to* $mod_m(x,X)$ *and* $\mathbf{FAC}^0(m)$ *is the class of functions which are* $\mathbf{AC}^0$*-reducible to* mod_m*. Also,*

$$\mathbf{ACC} = \bigcup_{i\geq 2} \mathbf{AC}^0(m), \qquad \mathbf{FACC} = \bigcup_{i\geq 2} \mathbf{FAC}^0(m)$$

In this section we will define the theory $\mathbf{V}^0(m)$ that characterizes $\mathbf{AC}^0(m)$. (Then $\mathbf{VACC} = \bigcup \mathbf{V}^0(m)$.) Following the discussion in Section 9.3, we will first define the universal theory $\overline{\mathbf{V}}^0(m)$. Here, we use the following quantifier-free defining axioms for mod_m (we identify the natural number m with the corresponding numeral $\underline{m}$):

$$mod_m(0, X) = 0 \quad (9.19)$$
$$X(x) \wedge mod_m(x, X) + 1 < m \supset mod_m(x + 1, X) = mod_m(x, X) + 1 \quad (9.20)$$
$$X(x) \wedge mod_m(x, X) + 1 = m \supset mod_m(x + 1, X) = 0 \quad (9.21)$$
$$\neg X(x) \supset mod_m(x + 1, X) = mod_m(x, X) \quad (9.22)$$

Definition 9.39. *For each $m \geq 2$, $\mathcal{L}_{\mathbf{FAC}^0(m)}$ is the smallest set that satisfies*

1) *$\mathcal{L}_{\mathbf{FAC}^0(m)}$ includes $\mathcal{L}_A^2 \cup \{pd, f_{\mathbf{SE}}, mod_m\}$*
2) *For each open formula $\varphi(z, \vec{x}, \vec{X})$ over $\mathcal{L}_{\mathbf{FAC}^0(m)}$ and term $t = t(\vec{x}, \vec{X})$ of $\mathcal{L}_A^2$, there is a string function $F_{\varphi,t}$ and a number function $f_{\varphi,t}$ (see Equations* (5.37) *and* (5.38) *in Section 5.6) in $\mathcal{L}_{\mathbf{FAC}^0(m)}$.*

The universal theory $\overline{\mathbf{V}}^0(m)$ is axiomatized by the axioms **B1**–**B11**, **L1**, **L2** *(Figure 5.1),* **B12**$'$, **B12**$''$ (5.40), **SE**$'$ (5.41) *and* (9.19) – (9.22) *Also, $\mathcal{L}_{\mathbf{FACC}} = \bigcup\{\mathcal{L}_{\mathbf{FAC}^0(m)} \mid m \geq 2\}$, and $\overline{\mathbf{VACC}} = \bigcup\{\overline{\mathbf{V}}^0(m) \mid m \geq 2\}$.*

Proposition 9.40. *The symbols in $\mathcal{L}_{\mathbf{FAC}^0(m)}$ and $\mathcal{L}_{\mathbf{FACC}}$ represent precisely the functions of $\mathbf{FAC}^0(m)$ and $\mathbf{FACC}$, respectively. A relation is in $\mathbf{AC}^0(m)$ or $\mathbf{ACC}$ if and only if it is represented by an open formula in $\mathcal{L}_{\mathbf{FAC}^0(m)}$ or $\mathcal{L}_{\mathbf{FACC}}$, respectively.*

For $m \geq 2$, the theory $\mathbf{V}^0(m)$ is defined using the formula $\delta_{\mathbf{MOD}_m}(x, X, Y)$, which states that Y is a "counting modulo m" array for X:

$$\delta_{\mathbf{MOD}_m}(x, X, Y) \equiv Y(0, 0) \ \wedge \ \forall z < x,$$
$$(X(z) \supset (Y)^{z+1} = ((Y)^z + 1) \mod m)) \wedge (\neg X(z) \supset (Y)^{z+1} = (Y)^z).$$

Since $(y \mod m)$ is an $\mathbf{AC}^0$ number function, $\delta_{\mathbf{MOD}_m}(x, X, Y)$ is equivalent to a $\mathbf{\Sigma}_0^B(\mathcal{L}_A^2)$ formula (by $\mathbf{FAC}^0$ Elimination Lemma 5.73). Indeed, if $\varphi(y)$ is a $\mathbf{\Sigma}_0^B$ formula, we can take $\varphi(y \mod m)$ as an abbreviation for the $\mathbf{\Sigma}_0^B$ formula

$$\exists r < m, \exists q \leq y, \ y = qm + r \wedge \varphi(r).$$

Definition 9.41. *For each $m \geq 2$, let $\mathbf{MOD}_m \equiv \forall X \forall x \exists Y \delta_{\mathbf{MOD}_m}(x, X, Y)$. Then $\mathbf{V}^0(m)$ has vocabulary $\mathcal{L}_A^2$ and is axiomatized by $\mathbf{V}^0$ and the axiom $\mathbf{MOD}_m$. Also, $\mathbf{VACC} = \bigcup\{\mathbf{V}^0(m) \mid m \geq 2\}$.*

Note that the string Y in $\mathbf{MOD}_m$ can be bounded by $|Y| \leq 1 + \langle x, m\rangle$.

The next excercise is to show that $mod_m^\star$ is provably total in $\mathbf{V}^0(m)$; it can be proved in the same way as Lemma 9.23.

Exercise 9.42. *Let $\mathcal{T}$ be $\mathbf{V}^0(m)$ extended by the defining axioms for Row, seq, left and right. Show that $\mathcal{T}$ proves the following general form of $\mathbf{MOD}_m$ (cf. Lemma 9.23):*

$$\exists Y \forall u < b \; \delta_{\mathbf{MOD}_m}(t(u), X^{[u]}, Y^{[u]})$$

for any number term $t(u)$ over $\mathcal{L}_A^2 \cup \{Row, seq, left, right\}$.

The characterization of $\mathbf{AC}^0(m)$ by $\mathbf{V}^0(m)$ is left as an exercise.

Exercise 9.43. *For each $m \geq 2$:*

a) *$\overline{\mathbf{V}}^0(m)$ is a conservative extension of $\mathbf{V}^0(m)$.*

b) *The $\mathbf{\Sigma}_1^1$-definable (and $\mathbf{\Sigma}_1^B$-definable) functions in $\mathbf{V}^0(m)$ are precisely those in $\mathbf{FAC}^0(m)$.*

c) *The $\mathbf{\Delta}_1^1$-definable (and $\mathbf{\Delta}_1^B$-definable) predicate in $\mathbf{V}^0(m)$ are precisely those in $\mathbf{AC}^0(m)$.*

Exercise 9.44. **a)** *The $\mathbf{\Sigma}_1^1$-definable (and $\mathbf{\Sigma}_1^B$-definable) functions of $\mathbf{VACC}$ are precisely the functions in $\mathbf{FACC}$. The $\mathbf{\Delta}_1^1$-definable (and $\mathbf{\Delta}_1^B$-definable) predicate of $\mathbf{VACC}$ are precisely the predicate in $\mathbf{ACC}$.*

b) *If $\mathbf{VACC}$ is finitely axiomatizable, then $\mathbf{ACC} = \mathbf{AC}^0(m)$, for some m.*

c) *If $\mathbf{VACC} \vdash NUMONES$, then $\mathbf{TC}^0 = \mathbf{AC}^0(m)$, for some m.*

d) *$\overline{\mathbf{VACC}}$ is a conservative extension of $\mathbf{VACC}$.*

e) *$\mathbf{VACC} \subseteq \mathbf{VTC}^0$.*

9.5 Theories for $\mathbf{NC}^k$ and $\mathbf{AC}^k$

Recall the definition of $\mathbf{AC}^0$ using circuit families in Section 4.1. 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 the same, but now the gates in C_n must have bounded fan-in. The corresponding function classes $\mathbf{FAC}^k$ and $\mathbf{FNC}^k$ are defined using circuits with multiple outputs. Also,

$$\mathbf{NC} = \bigcup_{k \geq 1} \mathbf{NC}^k = \bigcup_{k \geq 1} \mathbf{AC}^k, \qquad \mathbf{FNC} = \bigcup_{k \geq 1} \mathbf{FNC}^k = \bigcup_{k \geq 1} \mathbf{FAC}^k$$

Note that

$$\mathbf{NC}^0 \subsetneq \mathbf{AC}^0 \subsetneq \mathbf{NC}^1 \subseteq \mathbf{AC}^1 \subseteq \mathbf{NC}^2 \subseteq \ldots$$

Note also that $\mathbf{NC}^1$ coincides with **Alogtime**, the class of languages computable by alternating Turing machines in logarithmic time. Buss [**?**] shows that the Boolean sentence value problem is complete for **Alogtime**. Here we use the fact that the problem of evaluating a nicely encoded log-depth monotone Boolean circuit — i.e. a "balanced" formula, given its input, is complete for $\mathbf{NC}^1$ under $\mathbf{AC}^0$ reduction.

In general, for $k \geq 1$, $\mathbf{AC}^k$ is the class of relations which are $\mathbf{AC}^0$-reducible to the Monotone Circuit Value Problem, where the circuit has unbounded fan-in and depth bounded by $(\log n)^k$. Here, the circuit can be encoded as described in Subsection 8.2.3. For $k \geq 2$, $\mathbf{NC}^k$ has the same characterization, but the circuit has bounded fan-in.

Notice that the encoding of a circuit, as presented in Subsection 8.2.3, does not explicitly specify the connection (i.e., "wires") in the circuit. It is not known whether a log-depth circuit can be evaluated in $\mathbf{NC}^1$ when it is so encoded. Thus, it is not known whether the definition of $\mathbf{VNC}^k$ for $k \geq 2$ below will give $\mathbf{VNC}^1$ if we simply replace k by 1.

This section is organized as follows. First we introduce the theory $\mathbf{VNC}^1$, and then show that it extends the theory $\mathbf{VTC}^0$. Finally we present the theories $\mathbf{VAC}^k$, for $k \geq 1$, and $\mathbf{VNC}^k$, for $k \geq 2$.

9.5.1 The Theory $\mathbf{VNC}^1$

Consider a log depth, bounded fan-in, monotone Boolean circuit C whose underlying graph is a binary tree. Then C can be encoded "nicely" as follows. Suppose that C has $(2a-1)$ gates, where the a input gates are numbered $a, \dots, 2a-1$, and the output gate is numbered 1. Furthermore, the inputs to an internal gate x (where $x < a$) are from gates numbered $2x$ and $2x+1$. Thus, to fully specify C we need only to specify the type of each internal gate. This specification is given by a string variable G: if $G(x)$ holds then gate x is an $\wedge$-gate, otherwise it is an $\vee$-gate.

We formalize a polytime algorithm that computes the output of C, given inputs $I(0), \dots, I(a-1)$. We use the polytime algorithm presented in Subsection 8.2.3, but note that here the circuits is specified in a special way. In the following formula, $Y(x)$ is the value of gate x, for $x < 2a$. Define

$$\begin{aligned}\delta_{MFVP}(a, G, I, Y) \equiv\ & \forall x < a,\ Y(x+a) \leftrightarrow I(x) \wedge 0 < x \supset \\ & Y(x) \leftrightarrow [(G(x) \wedge Y(2x) \wedge Y(2x+1)) \vee (\neg G(x) \wedge (Y(2x) \vee Y(2x+1)))]\end{aligned}$$

Figure 9.1 depicts a computation of (the bits of) Y for $a = 6$. Here the "inputs" $I(0), \dots, I(5)$ are assigned to $Y(6), \dots, Y(11)$, respectively. Also $Y(1), \dots, Y(5)$ are computed by "gates" specified by $G(1), \dots, G(5)$ (not shown).

Definition 9.45 ($\mathbf{VNC}^1$). *Let*

$$MFVP \equiv \forall a \forall G \forall I \exists Y\, \delta_{MFVP}(a, G, I, Y) \tag{9.23}$$

The theory $\mathbf{VNC}^1$ *has vocabulary* $\mathcal{L}_A^2$ *and is axiomatized by* $\mathbf{V}^0$ *and the axiom* *MFVP*.

In $\mathbf{V}^0$, *MFVP* is equivalent to the same axiom with $|Y|$ bounded by $2a$.

In order to define the universal theory $\overline{\mathbf{VNC}}^1$ in the style of $\overline{\mathbf{VTC}}^0$, we use the function *Fval* that computes Y in the axiom *MFVP*.

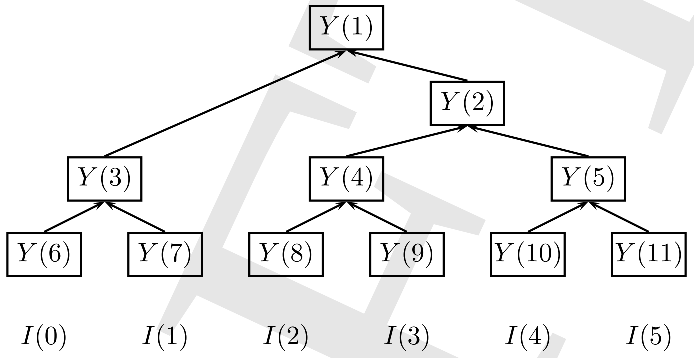

Figure 9.1: Computing Y which satisfies $\delta_{MFVP}(a,G,I,Y)$ for $a=6$

Exercise 9.46. *Let $Fval(x,G,I)$ be defined by*

$$Fval(x,G,I) = Y \leftrightarrow |Y| \leq 2a \wedge \delta_{MFVP}(x,G,I,Y)$$

Give a quantifier-free defining axiom for Fval in the vocabulary $\mathcal{L}_{\mathbf{FAC}^0}$.

The function $Fval^\star$ is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{VNC}^1$ (Exercise 9.56). It follows that $\mathbf{VNC}^1$ is precisely the theory $\mathbf{VC}$ as described in Subsection 9.3.1, where $F = Fval$ and $\mathbf{C}$ is the $\mathbf{AC}^0$ closure of $Fval$, i.e., $\mathbf{C} = \mathbf{NC}^1$. We obtain:

Corollary 9.47. *Let $\mathcal{L}_{\mathbf{FNC}^1}$ and $\overline{\mathbf{VNC}}^1$ be defined as in Definition 9.28 but using Fval instead of F.*

- **a)** *The functions in $\mathcal{L}_{\mathbf{FNC}^1}$ are precisely functions in $\mathbf{FNC}^1$. A relation is in $\mathbf{NC}^1$ iff it is represented by a $\mathbf{\Sigma}_0^B(\mathcal{L}_{\mathbf{FNC}^1})$ formula.*
- **b)** *$\overline{\mathbf{VNC}}^1$ is a conservative extension of $\mathbf{VNC}^1$.*
- **c)** *The $\mathbf{\Sigma}_1^1$-definable (and $\mathbf{\Sigma}_1^B$-definable) functions in $\mathbf{VNC}^1$ are precisely those in $\mathbf{FNC}^1$.*
- **d)** *The $\mathbf{\Delta}_1^1$-definable (and $\mathbf{\Delta}_1^B$-definable) predicate in $\mathbf{VNC}^1$ are precisely those in $\mathbf{NC}^1$.*

The theory $\mathbf{VNC}^1$ was first defined in [**?**, **?**] as a two-sorted version of Arai's single-sorted theory **AID** [**?**]. It was defined to be axiomatized by $\mathbf{V}^0$ and the axiom scheme $\mathbf{\Sigma}_0^B$-*TreeRec*, which is the set of formulas of the form

$$\exists Y \forall x < a,$$
$$(Y(x+a) \leftrightarrow \psi(x)) \wedge (0 < x \supset (Y(x) \leftrightarrow \varphi(x)[Y(2x), Y(2x+1)])) \quad (9.24)$$

where $\psi(x)$ is a $\mathbf{\Sigma}_0^B$ formula, $\varphi(x)[p,q]$ is a $\mathbf{\Sigma}_0^B$ formula which contains two Boolean variables p and q, and Y does not occur in ψ and φ. We will show that the two definitions of $\mathbf{VNC}^1$ are in fact equivalent.

START

Theorem 9.48. $\mathbf{VNC}^1$ *can be alternatively axiomatized by* $\mathbf{V}^0$ *together with the* $\mathbf{\Sigma}_0^B$*-TreeRec axiom scheme.*

Proof Sketch. First, it is easy to see that *MFVP* is an instance of the $\mathbf{\Sigma}_0^B$-*TreeRec* axiom scheme. Therefore it remains to show that $\mathbf{VNC}^1$ proves this axiom scheme.

The idea is to construct a large tree (a', G, I) so that from $Fval(a', G, I)$ we can extract Y satisfying (9.24) using $\mathbf{\Sigma}_0^B$**-COMP**. The key part of this construction is to show that $Y(x)$ can be computed uniformly using binary subtrees of constant depth. Since the desired subtrees do not contain $\neg$-gates, we will also construct subtrees that compute $\neg Y(x)$.

Formally, there are in total 16 Boolean functions

$$\beta_1, \ldots, \beta_8, \beta_9 \equiv \neg\beta_1, \ldots, \beta_{16} \equiv \neg\beta_8$$

in two variables p, q. Each β_i can be computed by a binary and-or tree of depth 2 with inputs among 0, 1, p, q, $\neg p$, $\neg q$. For $1 \le i \le 16$, 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 trees that provide inputs 0, 1, $X_i(x)$ $(1 \le i \le 16)$. □

9.5.2 The Theories $\mathbf{VNC}^k$ and $\mathbf{VAC}^k$

In general, for $k \ge 1$ to develop a theory for $\mathbf{AC}^k$ we formalize the polytime algorithm which solves the Monotone Circuit Value Problem just as in Subsection 8.2.3, but now the input circuit is required to have depth bounded by $(\log n)^k$. (Recall that the function $\log x$, or also $|x|$, is definable in $\mathbf{I\Delta}_0$, see Chapter 3.) Recall the formula δ_{MCVP} in Definition 8.43. Now we require that $d \le |a|^k$. Also, for $\mathbf{VNC}^k$ we need an extra condition on C, i.e., C has fan-in at most 2. This is expressed by the following formula:

$$Fanin2(a,d,E) \equiv \forall z < d \forall x < a \exists u_1, u_2 < a \forall v < a,\ E(z,v,x) \supset v = u_1 \vee v = u_2$$

Definition 9.49 ($\mathbf{VAC}^k$ and $\mathbf{VNC}^k$). *For* $k \ge 1$ *the theory* $\mathbf{VAC}^k$ *has vocabulary* $\mathcal{L}_A^2 \cup \{\log\}$ *and is axiomatized by* $\mathbf{V}^0$ *and the axiom*

$$\forall a \forall E \forall G \forall I \exists Y\ \delta_{MCVP}(a, |a|^k, E, G, I, Y) \tag{9.25}$$

For $k \geq 2$ the theory $\mathbf{VNC}^k$ has vocabulary $\mathcal{L}_A^2 \cup \{\log\}$ and is axiomatized by $\mathbf{V}^0$ and the axiom

$$\forall a \forall E \forall G \forall I (\mathit{Fanin2}(a,d,E) \supset \exists Y\, \delta_{MCVP}(a, |a|^k, E, G, I, Y)) \tag{9.26}$$

Now let $\mathit{Cval}(a,d,E,G,I)$ be the function that witnesses Y in (9.25) above. The next exercise is to show that $\mathit{Cval}^\star(a,|a|^k,E,G,I)$ is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{VAC}^k$, and for E such that $\mathit{Fanin2}(a,|a|^k,E)$ holds, then it is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{VNC}^k$. This can be used to show the characterization of $\mathbf{AC}^k/\mathbf{NC}^k$ by $\mathbf{VAC}^k/\mathbf{VNC}^k$.

Exercise 9.50. *Let $t(u)$ be a number term over $\mathcal{L}_A^2 \cup \{\mathit{Row}, \mathit{seq}, \mathit{left}, \mathit{right}, \log\}$.*

a) *For $k \geq 1$ show that $\mathbf{VAC}^k(\mathit{Row}, \mathit{seq}, \mathit{left}, \mathit{right}, \log)$ proves*

$$\exists Y \forall u < b\, \delta_{MCVP}(t(u), |t(u)|^k, E^{[u]}, G^{[u]}, I^{[u]}, Y^{[u]})$$

b) *For $k \geq 2$ show that $\mathbf{VNC}^k(\mathit{Row}, \mathit{seq}, \mathit{left}, \mathit{right}, \log)$ proves*

$$\forall u < b\, \mathit{Fanin2}(t(u), |t(u)|^k, E^{[u]}) \supset$$
$$\exists Y \forall u < b\, \delta_{MCVP}(t(u), |t(u)|^k, E^{[u]}, G^{[u]}, I^{[u]}, Y^{[u]})$$

It is straightforward that MFVP (9.23) is provable in the theory $\mathbf{V}^0$ extended by the axiom (9.26) for $k = 1$. The exercise below is to show that we could have obtained $\mathbf{VNC}^1$ just as other theories $\mathbf{VNC}^k$.

Exercise 9.51. *Show that*

$$\mathbf{VNC}^1(\log) \vdash \mathit{Fanin2}(a, |a|, E) \supset \exists Y\, \delta_{MCVP}(a, |a|, E, G, I, Y)$$

9.5.3 $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$

In this subsection we show that $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$. We have shown that $\mathbf{VTC}^0$ proves the Pigeonhole Principle in Subsection 9.2.6. Consequently $\mathbf{VNC}^1$ proves PHP. In the next chapter we will show that $\mathbf{\Sigma}_0^B$ theorems of $\mathbf{VNC}^1$ translate into family of tautologies having polysize **Frege** proofs. Thus we obtain as a corollary a theorem of Buss [**?**] that the family **PHP** has polysize **Frege** proofs.

Theorem 9.52. $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$.

Proof. We formalize the proof that the total number of bits in a string can be computed in $\mathbf{NC}^1$, i.e., that *numones* is $\mathbf{\Sigma}_1^1$ definable in $\mathbf{VNC}^1$. Informally, to count the number of bits in a string X of length n (i.e., $\mathit{numones}(n,X)$) we can use the divide-and-conquer technique: Let $c_{i+n} = X(i)$ for $0 \leq i < n$ (here we use $X(i)$ for the characteristic function of the relation $i \in X$). Also, for $1 \leq i < n$, $c_i = c_{2i} + c_{2i+1}$. Then $c_1 = \mathit{numones}(n,X)$. (See Figure 9.2 for an example.)

We need to formalize the computation of c_1 in $\mathbf{VNC}^1$. In fact the next theorem shows that we can formalize the same computation but for strings in

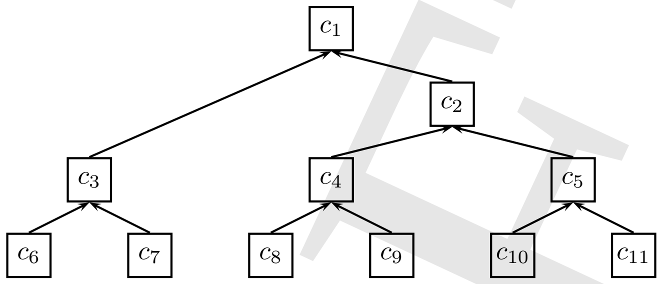

Figure 9.2: Computing $numones(6, X)$ in $\mathbf{NC}^1$

place of numbers and string addition in place of (number) addition. This is more general since converting a number into its binary representation can be done in $\mathbf{V}^0$ (Subsection 3.3.3). □

Theorem 9.53. $\mathbf{VNC}^1$ *proves*

$$\exists Z \forall x < a,\ Z^{[a+x]} = I^{[x]} \wedge x > 0 \supset Z^{[x]} = Z^{[2x]} + Z^{[2x+1]}$$

This theorem can be proved by using the notion of *ambiguous arithmetic notation* together with the generalization of Theorem 9.48 given in Theorem 9.55 below. Here we will formalize the another argument, from [**?**].

Proof. We compute $Z^{[x]}$ recursively as in Figure 9.2, but now the nodes contain $Z^{[x]}$ instead of c_x. We need to be careful here, for if we perform the string addition $Z^{[2x]} + Z^{[2x+1]}$ by $\mathbf{AC}^0$ circuits, then we will end up with an $\mathbf{AC}^1$ circuit. By the idea from [**?**], we encode each $Z^{[x]}$ by a pair of strings $(C^{[x]}, S^{[x]})$ so that $C^{[x]} + S^{[x]} = Z^{[x]}$. Essentially, $C^{[x]}$ contains the carries and $S^{[x]}$ contains the bit-wise sums while adding $C^{[2x]}, S^{[2x]}, C^{[2x+1]}, S^{[2x+1]}$.

Formally we define $\mathbf{NC}^0$ string functions $F_1(X, Y, Z, W)$ and $F_2(X, Y, Z, W)$ so that

$$X + Y + Z + W = F_1(X, Y, Z, W) + F_2(X, Y, Z, W)$$

First, we define for $0 \le z < |X| + |Y| + |Z|$:

$$\begin{aligned} F_1'(X, Y, Z)(z) &\leftrightarrow X(z) \oplus Y(z) \oplus Z(z) \\ F_2'(X, Y, Z)(0) &\leftrightarrow \bot \\ F_2'(X, Y, Z)(z+1) &\leftrightarrow ((X(z) \wedge Y(z)) \vee (X(z) \wedge Z(z)) \vee (Y(z) \wedge Z(z))) \end{aligned}$$

Then define

$$F_1(X, Y, Z, W) = F_1'(W, U, V), \qquad F_2(X, Y, Z, W) = F_2'(W, U, V)$$

where $U = F_1'(X, Y, Z)$ and $V = F_2'(X, Y, Z)$.

Lemma 9.54. $\mathbf{V}^0 \vdash F_1(X,Y,Z,W) + F_2(X,Y,Z,W) = X + Y + Z + W$.

Proof Sketch. By Exercise 5.43, $\mathbf{V}^0$ proves basic properties of string addition. Thus it suffices to show that

$$\mathbf{V}^0 \vdash F_1'(X,Y,Z) + F_2'(X,Y,Z) = X + Y + Z \tag{9.27}$$

For $i = 1, 2$, let $F_i''(x,X,Y,Z) = F_i'(\mathit{Cut}(x,X), \mathit{Cut}(x,Y), \mathit{Cut}(x,Z))$ (see (6.5) on page 127 for the definition of *Cut*). Then we can prove by induction on x that

$$\mathit{Cut}(x,X) + \mathit{Cut}(x,Y) + \mathit{Cut}(x,Z) = F_1''(x,X,Y,Z) + F_2''(x,X,Y,Z)$$

Both the base case and the induction step require case analysis and are straightforward. From this we can easily get (9.27). □

It remains to show that $\mathbf{VNC}^1$ proves the existence of C and S such that

$$\forall x < a, S^{[x+a]} = I^{[x]} \wedge C^{[x+a]} = \varnothing \wedge 0 < x \supset$$
$$C^{[x]} = F_1(C^{[2x]}, S^{[2x]}, C^{[2x+1]}, S^{[2x+1]}) \wedge S^{[x]} = F_2(C^{[2x]}, S^{[2x]}, C^{[2x+1]}, S^{[2x+1]})$$

Notice that the bits $C^{[x]}(z), S^{[x]}(z)$ are computed from

$$\{C^{[2x]}(y), C^{[2x+1]}(y), S^{[2x]}(y), S^{[2x+1]}(y) : z - 2 \le y \le z\}$$

(where implicitly $C^{[2x]}(y) \equiv \bot$ if $y < 0$, etc.). This is not in the form of the $\mathbf{\Sigma}_0^B$-*TreeRec* axioms, so first we transform C, S by defining their transposition C' and S':

$$C'^{[y]}(x) \leftrightarrow C^{[x]}(y), \qquad S'^{[y]}(x) \leftrightarrow S^{[x]}(y)$$

Then $C'^{[z]}(x)$ and $S'^{[z]}(x)$ are computed from

$$\{C'^{[y]}(2x), C'^{[y]}(2x+1), S'^{[y]}(2x), S'^{[y]}(2x+1) : z - 2 \le y \le z\}$$

Now the existence of C' and S' in $\mathbf{VNC}^1$ follows easily from Theorem 9.55 below. □

In the next theorem we show that $\mathbf{VNC}^1$ proves a generalization of the $\mathbf{\Sigma}_0^B$-*TreeRec* axiom scheme. Part **a** is the first step, but its proof technique can be used again to prove **b**.

Theorem 9.55. **a)** *Suppose that $2 \le k \in \mathbb{N}$, and $\psi(x)$ and $\varphi(x)[p_0, \ldots, p_{k-1}]$ are $\mathbf{\Sigma}_0^B$ formulas. Then $\mathbf{VNC}^1$ proves*

$$\exists Y, \forall x < ka\, a \le x \supset Y(x) \leftrightarrow \psi(x) \wedge$$
$$\forall x < a\, Y(x) \leftrightarrow \varphi(x)[Y(kx), \ldots, Y(kx+k-1)] \tag{9.28}$$

b) *Suppose that* $1 \le k, \ell \in \mathbb{N}$, *and* $\psi_i(x,y)$ *and* $\varphi_i(x,y)[p_1, q_1, \ldots, p_{k\ell}, q_{k\ell}]$ *are* $\mathbf{\Sigma}_0^B$ *formulas for* $1 \le i \le k$, *where* $\vec{p}, \vec{q}$ *are the Boolean variables. Then* $\mathbf{VNC}^1$ *proves the existences of* $Z_1, \ldots, Z_k$ *such that*

$$\forall z < c \forall x < a \bigwedge_{i=1}^{k} (Z_i^{[z]}(x+a) \leftrightarrow \psi_i(z,x) \wedge 0 < x \supset Z_i^{[z]}(x) \leftrightarrow \varphi_i(z,x)[\ldots])$$

where $[\ldots]$ *is the list of* $Z_i^{[z+j]}(2x), Z_i^{[z+j]}(2x+1)$, $1 \le i \le k, 0 \le j < \ell$. *Also,* $Z_i^{[z]}(y)$ *implicitly is* $\perp$ *if* $z < 0$.

Proof Sketch. **a)** We prove for the case $k = 4$. Similar arguments work for other cases. Using Theorem 9.48 we will define a', ψ', φ' so that from Y' that satisfies (9.24) (page 240) for a', ψ' and φ' we can obtain Y that satisfies (9.28) above.

Informally, consider Y in (9.28) as a tree whose nodes are labelled with $Y(x)$, $x < |Y|$, then Y has branching factor of 4 (since $k = 4$). Thus we need two layers of the tree in (9.24) to simulate one layer of Y in (9.28).

Formally, we will define injective map f so that $Y(x) \leftrightarrow Y'(f(x))$. Since the subtrees rooted at $Y(1)$, $Y(2)$ and $Y(3)$ form a partition of Y, we define f so that $f(1) = 4$, $f(2) = 5$ and $f(3) = 6$. (The subtrees rooted at $Y'(4)$, $Y'(5)$ and $Y'(6)$ of the binary tree Y' are disjoint.)

We 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 $\mathbf{\Sigma}_0^B$ with at most 2 Boolean variables, for $1 \le i \le 3$. Define

$$\begin{aligned}
\psi'(4^{m+1} + y) &\equiv \psi(4^m + y) && \text{for } y < 3 \cdot 4^m \\
\varphi'(4^{m+1} + y)[p,q] &\equiv \varphi_1(4^m + y)[p,q] && \text{for } y < 3 \cdot 4^m \\
\varphi'(2 \cdot 4^{m+1} + 2y)[p,q] &\equiv \varphi_2(4^m + y)[p,q] && \text{for } y < 3 \cdot 4^m/2 \\
\varphi'(2 \cdot 4^{m+1} + 2y + 1)[p,q] &\equiv \varphi_3(4^m + y)[p,q] && \text{for } y < 3 \cdot 4^m/2
\end{aligned}$$

Let f be the mapping

$$f(4^m + y) = 4^{m+1} + y \qquad \text{for } 0 \le y < 3 \cdot 4^m$$

By the results in Chapter 3, f is provably total in $\mathbf{I\Delta}_0$, and hence also $\mathbf{V}^0$.

Finally, let Y' satisfies (9.24) for $a' = f(a)$, ψ' and φ' above. Let Y be define as $Y(x) \leftrightarrow Y'(f(x))$. It is easy to verify that Y satisfies (9.28).

b) The proof is similar to the proof of **a**. Consider for example $k = 2, \ell = 2$. Using part **a** (for $k' = 8$), the idea is to construct a' and $\mathbf{\Sigma}_0^B$ formulas $\psi'(c,x)$ and $\varphi'(c,x)[p_0, \ldots, p_7]$ so that from the set Y that satisfies (9.28) (for a', ψ' and φ') we can obtain Z_1, Z_2.

The idea is to define partial, onto maps $f, g : \mathbb{N} \to \mathbb{N}$ and $h : \mathbb{N} \to \{1,2\}$ so that $Y(x) \leftrightarrow Z_{h(x)}^{[g(x)]}(f(x))$. (Then $\psi'(c,x) \leftrightarrow \psi_{h(x)}(g(x), f(x))$ and $\varphi'(c,x) \leftrightarrow \varphi_{h(x)}(g(x), f(x))$.) As in **a**, these maps can be computed easily using the binary representation of x.

In particular, assume w.l.o.g. that $c \geq 1$. Then the subtrees rooted at $Y(c)$, $\ldots$, $Y(3c-1)$ of the octree Y' are disjoint. It suffices to define f, g and h so that these subtrees are identical to the (overlapping) trees $Z_1^{[0]}$, $Z_2^{[0]}$, $\ldots$, $Z_1^{[c-1]}$, $Z_2^{[c-1]}$. Thus we can have

$$f(c+z) = 1, \qquad g(c+z) = \lfloor z/2 \rfloor, \qquad h(c+z) = 1 + (z \mod 2)$$

for $0 \leq z < 2c$. Similarly, for $0 \leq z < 2 \cdot 8^m c$ we can define f, g and h at $8^m c + z$ using the base 8 notation for z. □

Exercise 9.56. *Using part* **b** *of Theorem 9.55, show that the function* $Fval^\star$ *is* $\mathbf{\Sigma}_1^1$*-definable in* $\mathbf{VNC}^1$*.*

9.6 Theories for NL and L

NL (resp. **L**) is the class of problems solvable in a nondeterministic (resp. deterministic) Turing machine in space $O(\log n)$. An important result in complexity theory is that **NL** is closed under complementation [**?**, **?**]. This shows that **NL** and **FNL** are closed under (Cook-Turing) $\mathbf{AC}^0$-reduction.

First, we present the theory **VNL** that characterizes **NL** in the same way that $\mathbf{VTC}^0$ characterizes $\mathbf{TC}^0$. The theory **VNL** is an instance of **VC** obtained by adding to $\mathbf{V}^0$ an axiom that formalizes a polytime algorithm for the problem *PATH* defined below. Then we show that the relations in **NL** are precisely those represented by $\mathbf{\Sigma}_1^1$**-Krom** formulas, a subclass of $\mathbf{\Sigma}_1^1$ (indeed, $\mathbf{\Sigma}_1^B$). We also present the theory $\mathbf{V}^1$**-KROM**, and show that it is equivalent to **VNL**. Finally we define **VL**, a theory which characterizes **L** and show that it is the same as $\mathbf{\Sigma}_0^B$**-Rec** [**?**].

9.6.1 The Theory VNL

We use the fact that **NL** is the class of the problems that are $\mathbf{AC}^0$-reducible to the *PATH* problem. This is the problem of given a directed graph G and two designated verteces s and t, deciding whether there is a path from s to t. The complete function for **NL** that we use arises from the polytime algorithm which solves *PATH* by inductively computing all verteces in G that have distance from s at most $0, 1, \ldots, n$, where n is the number of verteces in G.

Formally, suppose that the verteces of G are numbered $0, \ldots, (a-1)$ $(a \geq 1)$, the designated vertex s is numbered 0, and that E is a 2-dimensional array coding G: $E(x, y)$ holds if and only if there is a directed edge from x to y in G (for $x, y < a$). The function $Conn(z, a, E)$ plays the role of F in the discussion in Section 9.3, where $Conn(z, a, E)$ is the set of all verteces of G that have distance from 0 at most z.

Proposition 9.57. **NL** *is the* $\mathbf{AC}^0$ *closure of Conn,* **FNL** *is the* $\mathbf{FAC}^0$ *closure of Conn.*

To obtain the universal theory $\overline{\mathbf{VNL}}$, consider the following quantifier-free defining axioms for *Conn*:

$$Conn(0, a, E) = \{0\} \tag{9.29}$$
$$Conn(z+1, a, E) = \\ Conn(z, a, E) \cup \{x < a \mid Conn(z, a, E) \cap Nr(x, a, E) \neq \varnothing\} \tag{9.30}$$

Here $\cup$, $\cap$ and Nr are $\mathbf{AC}^0$ functions: $X \cup Y$ (resp. $X \cap Y$) is the union (resp. intersection) of the sets X and Y, and $Nr(x, a, E)$ is the set of all neighbors y of x, $y < a$, such that $\langle y, x\rangle \in E$. Notice that these functions, together with the string constants $\varnothing$ and $\{0\}$, are in $\mathbf{FAC}^0$. Therefore the above defining axioms for *Conn* are quantifier-free formulas over $\mathcal{L}_{\mathbf{FAC}^0}$.

Definition 9.58 ($\overline{\mathbf{VNL}}$). *The vocabulary $\mathcal{L}_{\mathbf{FNL}}$ is defined just as $\mathcal{L}_{\mathbf{FC}}$ in Definition 9.28, with Conn replacing F. The theory $\overline{\mathbf{VNL}}$ has vocabulary $\mathcal{L}_{\mathbf{FNL}}$ and is defined to be $\overline{\mathbf{VC}}$ in that definition, with Conn (and its defining axioms* (9.29) *and* (9.30) *above) and $\mathcal{L}_{\mathbf{FNL}}$ replacing respectively F and $\mathcal{L}_{\mathbf{FC}}$.*

Proposition 9.59. *The functions in $\mathcal{L}_{\mathbf{FNL}}$ represents precisely the functions of* **FNL**. *A relation is in* **NL** *if and only if it is representable by an open (and therefore $\mathbf{\Sigma}_0^B$) formula of $\mathcal{L}_{\mathbf{FNL}}$.*

Now we define **VNL**. The additional axiom in **VNL** formalizes a computation of *Conn*, i.e., a polytime algorithm that solves *PATH*. In the formula $\delta_{CONN}(a, E, Y)$ below, $Y(z, x)$ holds if and only if there is a path from 0 to x of length at most z, i.e., $Y^{[z]} = Conn(z, a, E)$. (See also δ_F in (9.16).)

$$\delta_{CONN}(a, E, Y) \equiv Y(0,0) \wedge \forall x < a(x \neq 0 \supset \neg Y(0, x)) \wedge \\ \forall z < a \forall x < a,\ Y(z+1, x) \leftrightarrow (Y(z, x) \vee \exists y < a,\ Y(z, y) \wedge E(y, x)). \tag{9.31}$$

Definition 9.60 (The Theory VNL). *Define CONN to be the formula $\forall a \forall E \exists Y\, \delta_{CONN}(a, E, Y)$. The theory* **VNL** *has vocabulary $\mathcal{L}_A^2$ and is axiomatized by $\mathbf{V}^0$ together with the axiom CONN.*

It is easy to see that $\mathbf{V}^0$ proves that *CONN* is equivalent to the same axiom with $|Y|$ bounded by $(1 + \langle a, a\rangle)$. Hence **VNL** is a polynomial-bounded theory. The following exercise is an analogue of Lemma 9.23. It shows that in **VNL** we can simultaneously solve the *PATH* problem in multiple graphs, and it follows that we can also check the connectivity in a graph.

Exercise 9.61. *Show that for any $\mathcal{L}_A^2 \cup \{Row, seq, left, right\}$ number term $t(u)$,*

$$\mathbf{VNL}(Row, seq, left, right) \vdash \forall b \forall E \exists Z \forall u < b\, \delta_{CONN}(t(u), E^{[u]}, Z^{[u]}).$$

Deduce that the function $Conn^\star$ is provably total in **VNL**.

Corollary 9.62. $\overline{\mathbf{VNL}}$ *is a conservative extension of* **VNL**. *The class of $\mathbf{\Sigma}_1^1$-definable (and $\mathbf{\Sigma}_1^B$-definable) functions in* **VNL** *is precisely* **FNL**. *The class of $\mathbf{\Delta}_1^1$-definable (and $\mathbf{\Delta}_1^B$-definable) predicates in* **VNL** *is precisely* **NL**.

9.6.2 Representing NL by $\mathbf{\Sigma}_1^1$-Krom Formulas

In this subsection we present a characterization of **NL** relations by the subclass $\mathbf{\Sigma}_1^1$-**Krom** of the $\mathbf{\Sigma}_1^1$ (indeed, $\mathbf{\Sigma}_1^B$) formulas. This is based on Grädel's (second-order) descriptive characterization of *co*-**NL** [**?**] (which is the same as **NL** by Immerman-Szelepcsényi Theorem). This characterization of is derived from the fact that the Satisfiability Problem for propositional Krom formulas (i.e., 2-CNF formulas: formulas in conjunctive normal form where each clause contains at most two literals) is complete for *co*-**NL**. First, we define the notion of a Krom and a $\mathbf{\Sigma}_1^1$-**Krom** formula in bounded arithmetic.

Definition 9.63 (Krom and $\mathbf{\Sigma}_1^1$-Krom Formula). *A $\mathbf{\Sigma}_0^B$ formula $\varphi(\vec{z}, \vec{P})$ (which may contain other free variables) is a Krom formula* with respect to *the free variables $\vec{P}$ if it is a conjunction $\bigwedge_{i=1}^m C_i$, where for $1 \le i \le m$, C_i is a disjunction of (i) at most two* literals *of the form $P_j(\vec{z})$ or $\neg P_j(\vec{z})$, and (ii) a quantifier-free formula $\psi(\vec{z})$ which does not contain any of the variables $\vec{P}$.*

A $\mathbf{\Sigma}_1^1$-**Krom** *formula is a $\mathbf{\Sigma}_1^B$ formula of the form:*

$$\exists P_1 \dots \exists P_k \forall z_1 \le t_1 \dots \forall z_m \le t_m \varphi(\vec{z}, \vec{P}) \tag{9.32}$$

where $\varphi(\vec{z}, \vec{P})$ is a Krom formula with respect to $\vec{P}$.

Notice that $\mathbf{\Sigma}_0^B \not\subseteq \mathbf{\Sigma}_1^1$-**Krom**, although we will show later (Theorem 9.67) that for each $\mathbf{\Sigma}_0^B$ formula is equivalent (in the theory **VNL**) to a $\mathbf{\Sigma}_1^1$-**Krom** formula.

Example 9.64 (Transitive Closure in Graphs). *Suppose that a graph G is coded by (a, E) as before (page 246). Then 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:*

$$\begin{aligned} ContainTC(a, E, P) \equiv \forall x < a \forall y < a \forall z < a \\ (E(x, y) \supset P(x, y)) \wedge (P(x, y) \wedge E(y, z) \supset P(x, z)) \end{aligned}$$

It is easy to see that $ContainTC$ is equivalent to a quantifier-free Krom formula w.r.t. P. The following formula, stating that there is no *path from x_1 to x_2 in G, is equivalent to a* $\mathbf{\Sigma}_1^1$-**Krom** *formula:*

$$\varphi_{\neg Reach}(x_1, x_2, a, E) \equiv \exists P,\ ContainTC(a, E, P) \wedge \neg P(x_1, x_2) \tag{9.33}$$

The set Y that satisfies the comprehension for $\varphi_{\neg Reach}$:

$$|Y| \le a \wedge \forall y < a\, Y(y) \leftrightarrow \varphi_{\neg Reach}(x, y, a, E)$$

is the set of all verteces that are not *reachable from vertex x.*

The result in descriptive complexity and Immerman-Szelepcsényi Theorem give us:

Theorem 9.65 ($\mathbf{\Sigma}_1^1$-Krom Representation Theorem). **a)** *A relation is in co-***NL** *if and only if it is representable by a* $\mathbf{\Sigma}_1^1$-**Krom** *formula.*

b) *A relation is in* **NL** *if and only if it is representable by a* $\mathbf{\Sigma}_1^1$-**Krom** *formula.*

Proof Sketch. Part **b** follows from **a** and Immerman-Szelepcsényi Theorem A.14. Part **a** can be proved as follows. For the IF direction, consider a relation $R(\vec{x}, \vec{X})$ that is represented by the $\mathbf{\Sigma}_1^1$-**Krom** formula given in (9.32). For given inputs $(\vec{x}, \vec{X})$, $(\vec{x}, \vec{X}) \not\in R$ iff (9.32) is false, i.e., the following (polynomially long) propositional 2-CNF formula is unsatifiable:

$$\bigwedge_{z_1=0}^{v_1} \dots \bigwedge_{z_m=0}^{v_m} A_{z_1,\dots,z_m} \tag{9.34}$$

where

- v_i is the value of t_i;
- $A_{z_1,\dots,z_m}$ is the evaluation of $\varphi(\vec{z}, \vec{P}, \vec{x}, \vec{X})$ using the given values of $\vec{x}, \vec{X}$ and $\vec{z}$. We treat each atom $P_i(\vec{z})$ as a propositional variable. (Thus each $A_{z_1,\dots,z_m}$ is a 2-CNF formula, where each clause contains at most two propositional variables $P_i(\vec{z})$, $1 \le i \le m$.) Note that evaluating the $\mathbf{\Sigma}_0^B$ disjunct ψ in each clause C_i (as in Definition 9.63) can be done in $\mathbf{AC}^0$.

Assume w.l.o.g. that each clause in $A_{z_1,\dots,z_m}$ contains exactly two variables from $P_i(\vec{z})$. Then in general, (9.34) is unsatifiable iff its conjuncts contain a sequence of the form:

$$\ell_0 \supset \ell_1, \qquad \dots, \qquad \ell_k \supset \neg\ell_0$$

where ℓ_i are the literals in (9.34). The existence of such sequence can be checked by a NTM working in logspace. Thus $R \in$ *co*-**NL**.

For the ONLY IF direction, recall (Example 9.64) that the $\mathbf{\Sigma}_1^1$-**Krom** formula $\varphi_{\neg Reach}(x_1, x_2, a, E)$ states that in a graph encoded by $\langle a, E\rangle$, there is *no* path from x_1 to x_2. Suppose that R is an *co*-**NL** relation. Let M be a polytime logspace NTM that accepts inputs $(\vec{x}, \vec{X})$ iff $(\vec{x}, \vec{X}) \not\in R$. The configurations of M (without work tape content) form a polynomial (in $\vec{x}, |\vec{X}|$) size graph. Assume a suitable method of coding the configurations of M by numbers $\le t$, for some polynomial t of $(\vec{x}, |\vec{X}|)$, then such graph can be encoded by $\langle t, \varphi_{\mathsf{M}}\rangle$, where φ_{M} is a $\mathbf{\Sigma}_0^B$ formula such that

$$\varphi_{\mathsf{M}}(y, z, \vec{x}, \vec{X}) \text{ holds iff } z \text{ is a next configuration of } y$$

The $\mathbf{\Sigma}_1^1$-**Krom** formula that represents R is $\varphi_{\neg Reach}(x_1, x_2, t, \varphi_{\mathsf{M}})$, where x_1 and x_2 are respectively the number terms coding the initial and accepting configurations of M, and we have replaced E in (9.33) by φ_{M}. □

9.6.3 The Theory $\mathbf{V}^1$-KROM

The theory $\mathbf{V}^1$**-KROM** is developed in [**?**, **?**] based on the characterization of **NL** relations discussed above, and has been shown to characterize **NL**. In this subsection we present $\mathbf{V}^1$**-KROM**, and outline a proof of $\mathbf{V}^1$**-KROM** = **VNL**. It follows that $\mathbf{V}^1$**-KROM** is finitely axiomatizable.

Definition 9.66. *The theory* $\mathbf{V}^1$**-KROM** *is axiomatized by* 2-**BASIC** *(Figure 5.1) and the comprehension axiom scheme over all* $\mathbf{\Sigma}_1^1$**-Krom** *formulas.*

Although syntactically $\mathbf{\Sigma}_0^B \not\subseteq \mathbf{\Sigma}_1^1$**-Krom**, we will show that $\mathbf{V}^1$**-KROM** extends $\mathbf{V}^0$:

Lemma 9.67. $\mathbf{V}^0 \subseteq \mathbf{V}^1$**-KROM**.

Proof. First we prove:

Claim $\mathbf{V}^1$**-KROM** proves the multiple comprehension axioms (Lemma 5.49) for quantifier-free formulas, i.e., $\mathbf{V}^1$**-KROM** proves

$$\exists X \le \langle y_1, \ldots, y_k\rangle \forall z_1 < y_1 \ldots \forall z_k < y_k (X(z_1, \ldots, z_k) \leftrightarrow \varphi(z_1, \ldots, z_k)) \quad (9.35)$$

for any quantifier-free formula φ.

Recall that $X(z_1, \ldots, z_k)$ stands for $X(\langle z_1, \ldots, z_k\rangle)$. A first attempt to prove this claim might be that

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X \le \langle \vec{y}\rangle \forall z < \langle \vec{y}\rangle X(z) \leftrightarrow \exists \vec{z} < \vec{y},\, z = \langle \vec{z}\rangle \wedge \varphi(\vec{z})$$

However $\exists z_1 < y_1 \ldots \exists z_k < y_k (\ldots)$ is not a $\mathbf{\Sigma}_1^1$**-Krom** formula. Here (9.35) is proved using the following instance of the $\mathbf{\Sigma}_1^1$**-Krom-COMP**:

$$\exists X \le \langle \vec{y}\rangle \forall z < \langle \vec{y}\rangle X(z) \leftrightarrow (\exists P \forall \vec{z} < \langle \vec{y}\rangle,\ (P(\vec{z}) \leftrightarrow \varphi(\vec{z})) \wedge P(z))$$

Now we prove the lemma by showing that $\mathbf{V}^1$**-KROM** proves the multiple comprehension axiom for any $\mathbf{\Sigma}_0^B$ formula φ. The proof is by structural induction on φ. Assume w.l.o.g. that φ is in prenex form. The base case, φ is a quantifier-free formula, follows from the claim above. For the induction step, first consider the case where $\varphi(\vec{x}) \leftrightarrow \forall \vec{z} < \vec{a}\, \psi(\vec{x}, \vec{z})$. By the induction hypothesis for ψ,

$$\mathbf{V}^1\text{-}\mathbf{KROM} \vdash \exists X' \le \langle \vec{y}, \vec{a}\rangle \forall \vec{x} < \vec{y} \forall \vec{z} < \vec{a},\ X(\vec{x}, \vec{z}) \leftrightarrow \psi(\vec{x}, \vec{z})$$

Now we can apply the comprehension axiom for the $\mathbf{\Sigma}_1^1$**-Krom** formula $\forall \vec{z} < \vec{a}\, X'(\vec{x}, \vec{z})$.

Finally, the case where $\varphi(\vec{x}) \leftrightarrow \exists \vec{z} < \vec{a}\, \psi(\vec{x}, \vec{z})$ can be reduced to the previous case using the fact that if Y satisfies the multiple comprehension for a formula θ:

$$|Y| \le \langle \vec{b}\rangle \wedge \forall \vec{u} < \vec{b}\, Y(\vec{u}) \leftrightarrow \theta(\vec{u})$$

then Y' satisfies the multiple comprehension axiom for $\neg\theta$, where Y' is obtained from Y using multiple comprehension over the quantifier-free formula $\neg Y(\vec{z})$. $\square$

It is interesting to note that each $\mathbf{\Sigma}_0^B$ formula is equivalent to a $\mathbf{\Sigma}_1^1$-**Krom** formula. Moreover, this is provable in $\mathbf{V}^0$. We will outline a proof of the following theorem.

Theorem 9.68. *For each $\mathbf{\Sigma}_0^B$ formula φ, there is a $\mathbf{\Sigma}_1^1$-**Krom** formula φ' so that $\mathbf{V}^0 \vdash \varphi \leftrightarrow \varphi'$.*

Proof. Suppose that

$$\varphi \equiv \exists x_1 < a \forall y_1 < a \exists x_2 < a \forall y_2 < a \ldots \exists x_k < a \forall y_k < a \psi(x_1, y_1, \ldots, x_k, y_k)$$

The $\mathbf{\Sigma}_1^1$-**Krom** formula φ' states the existence of an array (of dimension $(2k-1)$) $S(x_1, y_1, \ldots, x_{k-1}, y_{k-1}, x_k)$ which decribes a search for the witnesses $x_1, \ldots, x_k$. Thus we perform a depth-first-search for "true" nodes on the or-and tree which results from φ by expanding the bounded quantifiers to finite disjunctions and conjunctions (Figure 9.3). If a node is an $\vee$-node, then each of its childrens 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 childrens are tested in parallel using universal quantifier. Also, if we encounter a "false" child of an $\wedge$-node, then we backtrack and continue to the next sibling of the parent of this node (see (9.38) below). At the leaves the of the tree (which are $\wedge$-nodes), we backtrack when ψ is false (see (9.37)).

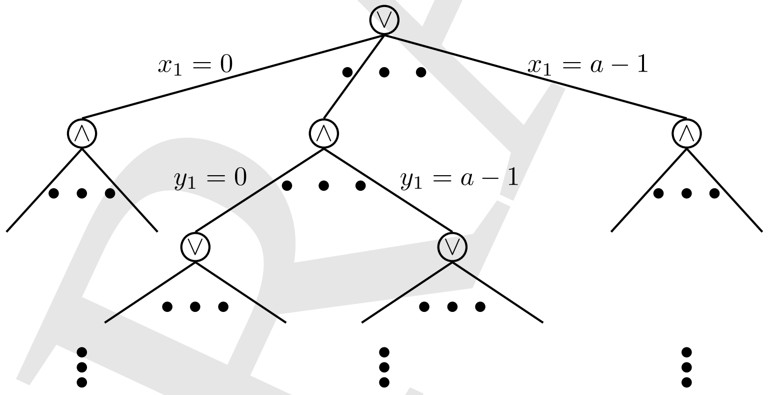

Figure 9.3: The $\vee$-$\wedge$ tree

Notation Since all number quantifiers here are bounded by a, we simply write $\exists x$ and $\forall x$ for $\exists x < a$ and $\forall x < a$, respectively.

Every $\wedge$-node is specified by a tuple $\langle x_1, y_1, \ldots, y_{j-1}, x_j \rangle$. For such a node, the fact that our search visit that node is indicated by

$$\forall y_j \ldots \forall y_{k-1} S(x_1, y_1, \ldots, y_{j-1}, x_j, y_j, 0, y_{j+1}, 0, \ldots, 0, y_{k-1})$$

Also, the fact that at least one child of the $\vee$-node specified by $\langle x_1, y_1, \ldots, y_{j-1} \rangle$ is true is coded by

$$\neg S(x_1, y_1, \ldots, y_{j-1}, x_j, y_j, a, 0, 0, \ldots, 0)$$

Now let ψ' be the Krom formula stating that S describes a successful search which starts at the leftmost child of the root:

$$\forall\vec{x}\forall\vec{y}\forall\overrightarrow{y'},\ (S(0,y_1,0,y_2,\ldots,0,y_{k-1},0)\wedge\neg S(a,0,0,\ldots,0)\wedge \tag{9.36}$$
$$((S(x_1,y_1,\ldots,x_k)\wedge\neg\psi(x_1,y_1,\ldots,x_k,y_k))\supset S(x_1,y_1,\ldots,x_k+1))\wedge \tag{9.37}$$
$$\bigwedge_{i=1}^{k-1}(S(x_1,y_1,\ldots,x_i,y_i,a,\vec{0})\supset S(x_1,y_1,\ldots,x_i+1,y'_i,0,y'_{i+1},\ldots,y'_k,0))) \tag{9.38}$$

The $\mathbf{\Sigma}^1_1$**-Krom** formula φ' is defined to be $\exists S\psi'$. We will show that $\varphi'\leftrightarrow\varphi$.

First we prove the direction $\varphi'\supset\varphi$. Thus assume the existence of S. For $1\le i\le k$ let

$$\begin{aligned}\rho_i(x_1,y_1,\ldots,x_{i-1},y_{i-1})\equiv\ &\forall\overrightarrow{y'}\,S(x_1,\ldots,y_{i-1},0,y'_i,0,\ldots,0,y'_{k-1},0)\wedge\\ &\neg\forall\overrightarrow{y'}\,S(x_1,\ldots,y_{i-1},a,y'_i,0,\ldots,0,y'_{k-1},0)\end{aligned}$$

Exercise 9.69. *Prove by induction on i, $i=k,\ldots,1$ that*

$$\rho_i(x_1,\ldots,y_{i-1})\supset\exists x_i\forall y_i\ldots\exists x_k\forall y_k\,\psi(x_1,\ldots,y_k)$$

*(Hint: You may need to use X-**MIN**, which is provable in $\mathbf{V}^1$-**KROM** by Lemma 9.67 above.)*

It follows that φ is true since (9.36) gives us $\rho_1()$.

Next we show that $\varphi\supset\varphi'$. Assume φ. We need to show the existence of a set S that satisfies φ'. Define the functions $f_1()$, $f_2(y_1)$, …, $f_k(y_1,\ldots,y_{k-1})$ as follows:

$$\begin{aligned}f_1()&=\min\{x_1:\forall y_1\ldots\exists x_k\forall y_k\,\psi(x_1,y_1\ldots,x_k,y_k)\}\\ f_2(y_1)&=\min\{x_2:\forall y_1\ldots\exists x_k\forall y_k\,\psi(f_1(),y_1,x_2,\ldots,x_k,y_k)\}\\ &\ldots\\ f_k(y_1,\ldots,y_{k-1})&=\min\{x_k:\forall y_k\,\psi(f_1(),y_1,f_2(y_1),\ldots,y_{k-1},x_k,y_k)\}\end{aligned}$$

These functions are in $\mathbf{FAC}^0$ and hence are definable in $\mathbf{V}^1$-**KROM** because $\mathbf{V}^1$-**KROM** extends $\mathbf{V}^0$. Now we define the set S. For a tuple $\langle x_1,y_1,\ldots,x_k,y_k\rangle$, let

$$i=\min\{j\le k:x_j\ne f_j(\ldots,y_{j-1})\}$$

Then

$$S(x_1,y_1,\ldots,x_k,y_k)=\begin{cases}\top & \text{if } x_i\le f_i(\ldots,y_{i-1})\\ \bot & \text{otherwise}\end{cases}$$

The remaining of the proof is left as an exercise.

Exercise 9.70. *Show that the set S defined above satisfies* (9.36) – (9.38).

□

The $\mathbf{\Sigma}^1_1$**-Krom** Representation Theorem shows that for every $\mathbf{\Sigma}^1_1$**-Krom** formula φ, there is another $\mathbf{\Sigma}^1_1$**-Krom** formula φ' so that $\varphi \leftrightarrow \neg\varphi'$ is true. However, it is a significant task to show that such equivalence can be established in $\mathbf{V}^1$**-KROM**. In fact, this requires formalizing the proof of Immerman-Szelepcsényi Theorem in $\mathbf{V}^1$**-KROM**. We will not do this here, but note that the fact that $\mathbf{V}^1$**-KROM** proves such equivalence shows that the **NL** relations are $\mathbf{\Delta}^1_1$-definable in $\mathbf{V}^1$**-KROM**, and that the **FNL** functions are $\mathbf{\Sigma}^1_1$-definable in $\mathbf{V}^1$**-KROM**. Here these characterizations follow from Corollary 9.62 and the next theorem.

Theorem 9.71 ([?]). $\mathbf{V}^1$**-KROM** = **VNL**.

Proof Sketch. First we show that **VNL** $\subseteq$ $\mathbf{V}^1$**-KROM**. Since $\mathbf{\Sigma}^B_0 \subset \mathbf{\Sigma}^1_1$**-Krom**, $\mathbf{V}^1$**-KROM** is an extension of $\mathbf{V}^0$. It remains to show that $\mathbf{V}^1$**-KROM** proves the axiom *CONN* (Definition 9.60). The following claim follows from definitions and the fact that $\mathbf{V}^1$**-KROM** extends $\mathbf{V}^0$:

Claim $\mathbf{V}^1$**-KROM** proves the multiple comprehension axiom scheme (Lemma 5.49) for $\mathbf{\Sigma}^1_1$**-Krom** formulas. For each $\mathbf{\Sigma}^1_1$**-Krom** formula φ, $\mathbf{V}^1$**-KROM** proves the comprehension for $\neg\varphi$.

Recall that in the formula $\delta_{CONN}(a, E, Y)$ in (9.31), $Y(z, x)$ holds iff in the graph G coded by (a, E) there is a path from 0 to x of length $\leq z$. The following formula (which is equivalent to a $\mathbf{\Sigma}^1_1$**-Krom** formula) states that there is *no* path from x_1 to x_2 in G of length $\leq z$:

$$\begin{aligned}\varphi_{\neg Dist}(x_1, x_2, z, a, E) \equiv \exists P,\ &\neg P(z, x_2) \wedge P(0, x_1) \wedge \forall y < a(y \neq x_1 \supset \neg P(0, y)) \\ &\wedge \forall u < z \forall x < a \forall y < a(P(u, x) \wedge E(x, y) \supset P(u + 1, y))\end{aligned}$$

By the claim above, $\mathbf{V}^1$**-KROM** proves the existence of Y that encodes the distances of all verteces in G from the vertex 0:

$$\forall z < a \forall x < a,\ Y(z, x) \leftrightarrow \neg\varphi_{\neg Dist}(0, x, z, a, E)$$

Thus $Y(z, x)$ holds iff the distance from 0 to x is at most z. Hence this set Y also satisfies $\delta_{CONN}(a, E, Y)$ in (9.31).

Now we show that $\mathbf{V}^1$**-KROM** $\subseteq$ **VNL**. Let $\exists\vec{P}\forall\vec{z} \leq \vec{t}\varphi(\vec{z}, x, \vec{P})$ be a $\mathbf{\Sigma}^1_1$**-Krom** formula (which may contain other free variables). We need to show that

$$\mathbf{VNL} \vdash \exists X \leq b \forall x < b,\ X(x) \leftrightarrow \exists\vec{P}\forall\vec{z} \leq \vec{t}\varphi(\vec{z}, x, \vec{P})$$

Essentially, the idea is to formalize in **VNL** the IF direction in the proof of part **a** of the $\mathbf{\Sigma}^1_1$**-Krom** Representation Theorem 9.65. In other words, for each value of $x < a$, we construct a graph G_x (coded by $(t(x), E^{[x]})$ for some term t and string variable E) whose verteces are labelled with the "literals" $P_i(\vec{x})$ and $\neg P_i(\vec{x})$, for $\vec{x} \leq \vec{t}$. For literals ℓ_1 and ℓ_2, there is an edge from ℓ_1 to ℓ_2 if $\ell_1 \supset \ell_2$

occurs in the CNF formula (9.34). (We can assume that each clause in (9.34) contains exactly two literals — duplicating the literals if necessary.)

Now $\exists \vec{P} \forall \vec{z} \le \vec{t} \varphi(\vec{z}, x, \vec{P})$ is *false* iff in G_x there is a path from some literal ℓ to its negation $\neg\ell$ and a path from $\neg\ell$ to ℓ. Although *CONN* only specify the verteces that are reachable from a single vertex 0, by Exercise 9.61, we can check in **VNL** simultaneously if $\neg\ell$ is reachable from ℓ, for all literals ℓ. □

9.6.4 The Theory VL

The theory **VL** is obtained from the fact that the restriction of the *PATH* problem to directed graphs with outdegree at most 1 is complete for **L**. We will define **VL** and $\overline{\mathbf{VL}}$ as discussed in Section 9.3, and point out that **VL** is equivalent to the theory $\mathbf{\Sigma}_0^B$**-Rec** from [**?**]. First, the function $\mathit{UniConn}(z, a, E)$ is defined to be the same as $\mathit{Conn}(z, a, E)$ when (a, E) codes such a graph, and the default value $\varnothing$ otherwise.

Proposition 9.72. **L** *is the* $\mathbf{AC}^0$ *closure of UniConn,* **FL** *is the* $\mathbf{FAC}^0$ *closure of UniConn.*

(In Subsection 9.7.1 we prove a recursion characterization of **FL** which originated from [**?**].) Now we give the defining axiom for *UniConn*. First, the following formula states that (a, E) codes a graph of outdegree ≤ 1:

$$OUT^{\le 1}(a, E) \equiv \forall x < a \exists y < a \forall z < a\, E(x, z) \supset y = z$$

Note that $OUT^{\le 1}$ is equivalent to a quantifier-free formula over $\mathcal{L}_{\mathbf{FAC}^0}$. The universal defining axioms for *UniConn* are (the quantifier-free equivalence over $\mathcal{L}_{\mathbf{FAC}^0}$ of):

$$(\neg OUT^{\le 1}(a, E) \vee z = 0) \supset \mathit{UniConn}(z, a, E) = \{0\} \tag{9.39}$$

$$\begin{aligned} &OUT^{\le 1}(a, E) \supset \mathit{UniConn}(z + 1, a, E) = \\ &\qquad \mathit{UniConn}(z, a, E) \cup \{x < a \mid \mathit{UniConn}(z, a, E) \cap \mathit{Nr}(x, a, E) \ne \varnothing\} \end{aligned} \tag{9.40}$$

Definition 9.73 ($\overline{\mathbf{VL}}$). $\mathcal{L}_{\mathbf{FL}}$ *is defined as* $\mathcal{L}_{\mathbf{FC}}$ *in Definition 9.28 with UniConn replacing F. The theory* $\overline{\mathbf{VL}}$ *is the instance of* $\overline{\mathbf{VC}}$*, defined using UniConn (with defining axioms* (9.39) *and* (9.40) *above) and* $\mathcal{L}_{\mathbf{FL}}$ *instead of F and* $\mathcal{L}_{\mathbf{FC}}$*.*

Proposition 9.74. *The functions in* $\mathcal{L}_{\mathbf{FL}}$ *represents precisely the functions of* **FL**. *A relation is in* **L** *if and only if it is representable by an open (and therefore* $\mathbf{\Sigma}_0^B$*) formula of* $\mathcal{L}_{\mathbf{FL}}$*.*

Definition 9.75 (The Theory VL). *Define UniCONN to be the formula*

$$\forall a \forall E,\ OUT^{\le 1}(a, E) \supset \exists Y\, \delta_{CONN}(a, E, Y)$$

The theory **VL** *has vocabulary* $\mathcal{L}_A^2$ *and is axiomatized by* $\mathbf{V}^0$ *and UniCONN.*

The following exercise is analogous to Lemma 9.23, and is used to show that $\overline{\mathbf{VL}}$ is conservative over **VL**. It also shows that **VL** is an instance of **VC** that characterizes **L**.

Exercise 9.76. *Show that for any $\mathcal{L}_A^2 \cup \{Row, seq, left, right\}$ number term $t(u)$, $\mathbf{VL}(Row, seq, left, right)$ proves*

$$\forall u < b\, OUT^{\leq 1}(t(u), E^{[u]}) \supset \exists Y \forall u < b\, \delta_{CONN}(t(u), E^{[u]}, Y^{[u]}).$$

Derive that the function $UniConn^\star$ is $\mathbf{\Sigma}_1^1$ definable in **VL**.

Corollary 9.77. $\overline{\mathbf{VL}}$ *is a conservative extension of* **VL**. *The class of $\mathbf{\Sigma}_1^1$-definable (and $\mathbf{\Sigma}_1^B$-definable) functions in* **VL** *is precisely* **FL**. *The class of $\mathbf{\Delta}_1^1$-definable (and $\mathbf{\Delta}_1^B$-definable) predicates in* **VL** *is precisely* **L**.

Zambella [**?**] introduced the theory $\mathbf{\Sigma}_0^B$-**Rec** which is essentially $\mathbf{V}^0$ together with the following axiom scheme:

$$\forall x < a \exists y < a\, \varphi(x, y) \supset \exists Z,\, (Z)^0 = 0 \wedge \forall x < a\, \varphi((Z)^x, (Z)^{x+1}).$$

where φ is a $\mathbf{\Sigma}_0^B$ formula not involving Z.

Exercise 9.78. *Show that* **VL** *is the same as* $\mathbf{\Sigma}_0^B$-**Rec**.

Theorem 9.79. $\mathbf{VNC}^1 \subseteq \mathbf{VL}$.

Proof. It suffices to show that **VL** proves the *MFVP* axiom (9.23) on page 239. Given a, G, I, we construct a graph encoded by (a', E) so that the truth value of $Fval(a, G, I)(1)$ can be obtained from $UniConn(a', a', E)$. (More generally, for each x, $1 \leq x < 2a$ we can construct a graph $(t(x), E^{[x]})$ so that $Fval(a, G, I)(x)$ can be computed from $UniConn(t(x), t(x), E^{[x]})$. Exercise 9.76 **b** can be used to find $Fval(a, G, I)$.)

The graph (a', E) describes a depth-first traversal in the circuits G to verify that it outputs $\top$. The verteces are

$$\{0\} \cup \{\langle x, \mathsf{d}, 0\rangle : 1 \leq x < 2a\} \cup \{\langle x, \mathsf{u}, v\rangle : 1 \leq x < 2a, 0 \leq v \leq 1\}$$

where $\mathsf{d} = 1$ (down) and $\mathsf{u} = 2$ (up) are used to to indicate direction of the traversal, and v is the value of the evaluation of node x in the circuit. The edges of this graph are specified by E as follows:

$$\begin{array}{rl}
E(\langle 0,0,0\rangle, \langle 1,\mathsf{d},0\rangle) & \text{(note that } \langle 0,0,0\rangle = 0) \\
E(\langle x,\mathsf{d},0\rangle, \langle 2x,\mathsf{d},0\rangle) & \text{for } 1 \leq x < a \\
E(\langle x+a,\mathsf{d},0\rangle, \langle x+a,\mathsf{u},0\rangle) & \text{if } \neg I(x)\text{, for } 0 \leq x < a \\
E(\langle x+a,\mathsf{d},0\rangle, \langle x+a,\mathsf{u},1\rangle) & \text{if } I(x)\text{, for } 0 \leq x < a \\
E(\langle x,\mathsf{u},1\rangle, \langle \lfloor x/2\rfloor,\mathsf{u},1\rangle) & \text{for } 1 \leq x < a \\
E(\langle 2x,\mathsf{u},0\rangle, \langle x,\mathsf{u},0\rangle) & \text{if } G(x) \text{ is an } \wedge\text{-gate, for } 1 \leq x < a \\
E(\langle 2x,\mathsf{u},0\rangle, \langle 2x+1,\mathsf{d},0\rangle) & \text{if } G(x) \text{ is an } \vee\text{-gate, for } 1 \leq x < a
\end{array}$$

Let $a' = \langle 2a-1, 2, 1\rangle$. It is easy to see that (a', E) encodes a graph of outdegree ≤ 1. Now we can prove that if a node of G is visited then our traversal evaluates it correctly:

$$\varphi(x,a,G,I) \equiv \exists z\, \mathit{UniConn}(z,a',E)(\langle x,\mathsf{d},0\rangle) \supset \\ (\mathit{Fval}(a,G,I)(x) \leftrightarrow \exists z'\, \mathit{UniConn}(z',a',E)(\langle x,\mathsf{u},1\rangle))$$

To prove $\forall x < 2a\, \varphi(x,a,G,I)$, we prove $\forall y \leq (1+|a|)\, \psi(y,a,G,I)$ by induction on y, where

$$\psi(y,a,G,I) \equiv \forall x < 2a(|x| = 1+|a|-y \supset \varphi(x,a,G,I))$$

Now from $E(\langle 0,0,0\rangle, \langle 1,\mathsf{d},0\rangle)$ we have $\mathit{UniConn}(1,a',E)(\langle 1,\mathsf{d},0\rangle)$. Then by $\varphi(1,a,G,I)$ we have $\mathit{Fval}(a,G,I)(1) \leftrightarrow \exists z'\, \mathit{UniConn}(z',a',E)(\langle 1,\mathsf{u},1\rangle)$. $\square$

9.7 The Number Recursion Operation

We define the number recursion operation which produces a new number function from existing number functions. This operation is similar to *limited recursion* (but the latter defines new *string functions* from existing string functions — Definition 6.15). It is useful in characterizing **FL** and a number of its subclasses. For a class **FC**, the recursion-theoretic characterization that we present here can be used to obtain universal extension of **VC** which is equivalent to $\overline{\mathbf{VC}}$. We will not prove these equivalences here, but point out that each recursion scheme can be proved in the corresponding theory **VC**.

Definition 9.80 (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*

$$f(0,\vec{x},\vec{X}) = g(\vec{x},\vec{X}) \tag{9.41}$$

$$f(y+1,\vec{x},\vec{X}) = h(y, f(y,\vec{x},\vec{X}), \vec{x}, \vec{X}) \tag{9.42}$$

If further $f(y,\vec{x},\vec{X}) \leq t(y,\vec{x},\vec{X})$, then we also say that f is obtained by t-bounded number recursion 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.

In the following subsections, first we characterize **FL** using polynomial-bounded number recursion. Then we define the number summation operation, an special case of the number recursion operation, and show that $\mathbf{FTC}^0$ is closed under this operation. In Subsection 9.7.3 we prove some properties of the k-bounded number recursion operation which will be used in the characterizations of $\mathbf{FAC}^0(2)$ and $\mathbf{FAC}^0(6)$ in Subsection 9.7.4. Finally we present an interesting recursion theoretic characterization of $\mathbf{FNC}^1$ using 4-bounded number recursion. This characterization of $\mathbf{FNC}^1$ is based on Barrington's result that the word problem for the permutation group S_5 is complete for $\mathbf{NC}^1$.

9.7.1 Lind's Characterization of FL

We prove the two-sorted version of Lind's characterization of **FL**, which is similar to Cobham's recursion characterization of **FP**.

Theorem 9.81 (Lind's Characterization of FL). **FL** *is the same as the class of functions obtained from* $\mathbf{FAC}^0$ *by finitely many applications of composition, string comprehension, and polynomial-bounded number recursion.*

Proof Sketch. The proof is similar to the proof of Cobham's Characterization of **FP** (Theorem 6.16). The $\supseteq$ direction follows from the fact that $\mathbf{FAC}^0$ is contained in **FL**, and that **FL** is closed under composition, string comprehension, and polynomial-bounded number recursion.

For the $\subseteq$ direction, it suffices to consider logspace polytime Turing machines. Let M be such a machine working on input $\vec{x}, \vec{X}$. Recall that since M works in logspace, a configuration of M does not include the content of its input tape. Assume a reasonable encoding of the configurations of M by numbers which are > 0 (see also Exercise 6.13). There are a polynomial $t(\vec{x}, |\vec{X}|)$ bounding the running time of M, and $\mathbf{AC}^0$ functions $init_{\mathsf{M}}(\vec{x}, \vec{X})$ and $next_{\mathsf{M}}(z, \vec{x}, \vec{X})$ such that:

- $init_{\mathsf{M}}(\vec{x}, \vec{X})$ is the initial configuration of M, $init_{\mathsf{M}}(\vec{x}, \vec{X}) \le t(\vec{x}, |\vec{X}|)$;
- $z' = next_{\mathsf{M}}(z, \vec{x}, \vec{X})$ if z and z' code two consecutive configuration of M, or $z' = z$ if z codes a final configuration of M, or $z' = 0$ if z does not code a configuration of M.

Then the function $conf_{\mathsf{M}}(y, \vec{x}, \vec{X})$, which is the configuration of M at time y, satisfies (9.41) and (9.42) with $init_{\mathsf{M}}$ replacing g and $conf_{\mathsf{M}}$ replacing h.

Suppose for example that M computes a string function $F(\vec{x}, \vec{X})$, then the bits of F can be computed by looking at the times when M writes to its output tape.

Exercise 9.82. *Define using composition and polynomial-bounded number recursion 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 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 $conf_{\mathsf{M}}(write_{\mathsf{M}}(y, \vec{x}, \vec{X}), \vec{x}, \vec{X})$. Now it is easy to see that F can be obtained from the above functions using composition and string comprehension. $\square$

9.7.2 Number Summation

The number summation operation is an instance of number recursion. In this subsection we present a characterization of $\mathbf{FTC}^0$ using this operation.

Definition 9.83 (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.

Notice that sum_f can be obtained from f by number recursion.

Theorem 9.84. *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.*

Proof. For the ONLY IF direction, by Definition 9.7 and Theorem 9.6, we need to show only that *numones* can be obtained by number summation from $\mathbf{AC}^0$ function. But this is clear, since

$$numones(x, X) = \sum_{y=0}^{x} X(y)$$

where we write 0 for $\bot$ and 1 for $\top$.

For the other direction, also by Definition 9.7 and Theorem 9.6, it suffices to show that if $f(x)$ is a $\mathbf{TC}^0$ function, then the function $\mathsf{sum}_f(y)$ is also in $\mathbf{FTC}^0$. In fact we will show that $\mathsf{sum}_f(x)$ is $\mathbf{\Sigma}_1^1$-definable in $\mathbf{VTC}^0$.

First, $f^\star$ is also a $\mathbf{TC}^0$ function, and is $\mathbf{\Sigma}_1^1$ definable in $\mathbf{FTC}^0$. By definition,

$$(f^\star(Z))^u = f((Z)^u), \qquad \text{for } u \le y$$

Let Z be such that $(Z)^u = u$, for all $u \le y$. Let $Y = f^\star(Z)$. Then

$$\mathsf{sum}_f(y) = \sum_{x \le y} (Y)^x$$

We calculate $\mathsf{sum}_f(y)$ using *numones*. Let W be defined (using $\mathbf{\Sigma}_0^B$-$\mathbf{COMP}$) such that

$$W(x|Y| + v) \qquad \text{holds iff} \qquad x \le y, v < (Y)^x$$

Then it is easy to verify that $\mathsf{sum}_f(y) = numones((y+1)|Y|, W)$. □

9.7.3 k-Bounded Number Recursion

In the next subsection we will present recursion theoretic characterizations of $\mathbf{FAC}^0(2)$, $\mathbf{FAC}^0(6)$ and $\mathbf{FNC}^1$ using the k-bounded number recursion (Definition 9.80) for suitable $k \in \mathbb{N}$. In this subsection we prove some properties of this recursion scheme, which will be useful later. For a constant $k \in \mathbb{N}$, suppose that the number function f is obtained from g and h using k-bounded number recursion: $f(x) \le k$ for all x, and

$$\begin{aligned} f(0) &= g \\ f(x+1) &= h(x, f(x)) \qquad \text{for } x \ge 0 \end{aligned}$$

(f, g and h may contain other parameters). Without loss of generality, we may assume that $h(x,z) \le k$ for all x and $z \le k$. Write $h(x,z)$ as $h_x(z)$, then $f(x)$ is just the composition

$$f(x) = h_{x-1} \circ h_{x-2} \circ \ldots \circ h_0(g) \tag{9.43}$$

Since $f(x) \le k$ for all x, we are interested in the values of $h_x(z)$ for $z \in \{0,\ldots,k\}$. Thus we will implicitly assume that the domain and range of h_x are $\{0,\ldots,k\}$, i.e., h_x is a function

$$h_x : \{0,\ldots,k\} \to \{0,\ldots,k\}$$

It is often easier to handle h_x when it is a $(k+1)$-*permutations*, i.e., h_x is surjective.

Notation *For a fixed $k \in \mathbb{N}$, we say that $h(x,\cdot)$ is a $(k+1)$-*permutation *(or just* permutation*) if*

$$h_x : \{0,\ldots,k\} \to \{0,\ldots,k\}$$

is onto for all x. Let ${}^{(k+1)}(k+1)$ denote the set of all functions $\{0,\ldots,k\} \to \{0,\ldots,k\}$, and $S_k \subseteq {}^k k$ the set of all k-permutations. Also denote by $Rng_k(h_x)$ the "range" of h_x in ${}^{(k+1)}(k+1)$:

$$Rng_k(h_x) = \{h(x,0),\ldots,h(x,k)\}$$

If h is a k-permutation, and f is obtained from g and h using number recursion, then we also say that f is obtained by number recursion from k-permutation.

Intuitively, we want to show that it suffices to apply the k-bounded number recursion to functions g, h where h is a $(k+1)$-permutation (Theorem 9.87 below). We need to establish a few technical results. First, Lemma 9.85 below states that any application of the k-bounded number recursion where h_x are not $(k+1)$-permutation, for all x, can be simulated by an application of the $(k-1)$-bounded number recursion.

Lemma 9.85. *Let be $h(x,z)$ be a function such that $h_x \notin S_{k+1}$, for all x. Suppose that $k \ge 1$ and that f is obtained from g and h by k-bounded number recursion. Then there are functions*

- *g', h' in the $\mathbf{FAC}^0$ closure of $\{g,h\}$, and*
- *f' obtained from g' and h' by $(k-1)$-bounded number recursion,*

such that f is in the $\mathbf{FAC}^0$ closure of f'.

Proof. The intuition is that since $h_x \notin S_{k+1}$ (for all x), we can compose each h_x with suitable functions to obtain a function $h'_x : \{0, \dots, k-1\} \to \{0, \dots, k-1\}$. We formalize this argument as follows. Let $\ell(x)$ be the following function (so that $\ell(x+1)$ be the minimal number which is not in the range of h_x, for $x \geq 0$):

$$\ell(0) = y \leftrightarrow (g = 0 \wedge y = 1) \vee (0 < g \wedge y = 0)$$
$$\ell(x+1) = y \leftrightarrow \forall z \leq k\, y \neq h(x,z) \wedge \forall v < y \exists z \leq k\, v = h(x,z)$$

Now $Rng_k(h_x) \subseteq \{0, \dots, k\} \setminus \{\ell(x)\}$. We define the functions $r(x,z)$ and $r^{-1}(x,z)$ so that $r_x(z) = r(x,z)$ is a bijection between $\{0, \dots, k\} \setminus \{\ell(x)\}$ and $\{0, \dots, k-1\}$, and $r_x^{-1}(z) = r^{-1}(x,z)$ is its inverse:

$$r(x,z) = \begin{cases} z & \text{if } z < \ell(x) \\ z-1 & \text{if } \ell(x) < z \leq k \\ k & \text{otherwise} \end{cases} \qquad r^{-1}(x,z) = \begin{cases} z & \text{if } z < \ell(x) \\ z+1 & \text{if } \ell(x) \leq z < k \\ \ell(x) & \text{if } k \leq z \end{cases}$$

Let

$$h'_x(z) = r_{x+1} \circ h_x \circ r_x^{-1}(z)$$

Then it is easy to see that $h'_x \in {}^k k$, for all x. Let $f'(x)$ be defined as

$$f'(0) = r(0,g)$$
$$f'(x+1) = h'(x, f'(x))$$

Then $f'(x) = r(x, f(x))$ for all $x \geq 0$. Hence $f(x) = r^{-1}(x, f'(x))$. □

The next lemma says that if h_0 is not a $(k+1)$-permutation, then k-bounded number recursion using h can be simulated by $(k-1)$-bounded number recursion and number recursion using $(k+1)$-permutation:

Lemma 9.86. *Let $k \geq 1$ and $h(x,z)$ be a function such that $h_0 \notin S_{k+1}$. Suppose that f is obtained from g and h by k-bounded number recursion. Then f can also be obtained from g and h by taking* $\mathbf{FAC}^0$ *closure, $(k-1)$-bounded number recursion and number recursion using $(k+1)$-permutations.*

Proof Sketch. Since h_0 is not a $(k+1)$-permutation, for each $x \geq 0$ the range of h_x that is needed to compute f can be regarded as a proper subset of $\{0, \dots, k\}$. The issue is to (uniformly) identify this subset, then we can use Lemma 9.85 above. Indeed, we will identify a number $\ell(x) \leq k$ which can be removed from the codomain of h_x without changing the value of f.

Let $m(x)$ be the function

$$m(x) = \min\{u \leq x : h_u \notin S_{k+1}\}$$

Notice that $0 \leq m(x) \leq x$ for $x \geq 0$. To define $\ell(x)$, first consider the case where $m(x) = x$. Then we can simply take

$$\ell(x) = \min\{y \leq k : \neg\exists z \leq k\; h_x(z) = y\}$$

For the case where $m(x) < x$, then h_u are all $(k+1)$-permutations, for $m(x) < u \le x$. Thus the value of $\ell(x)$ can be obtained by number recursion using these $(k+1)$-permutations. Formally, define the $(k+1)$-permutation

$$h'(x,u,z) = \begin{cases} h(u,z) & \text{if } m(x) < u \le x \\ z & \text{other wise} \end{cases}$$

Let $\ell'(x,u)$ be obtained by number recursion using $(k+1)$-permutation:

$$\ell'(x,0) = \min\{y \le k : \neg\exists z \le k\ h_{m(x)}(z) = y\}$$
$$\ell'(x,u+1) = h'(x,u,\ell'(x,u))$$

Now $\ell(x) = \ell'(x,x)$ can be safely removed from the codomain of h_x without changing f. In other words, define

$$h''(x,z) = \begin{cases} h(x,z) & \text{if } h(x,z) \ne \ell'(x,x) \\ \min\{y \le k : y \ne \ell'(x,x)\} & \text{otherwise} \end{cases}$$

Then $h''_x \notin S_{k+1}$ for all x, and f is obtained from g and h'' by number recursion. The conclusion follows from Lemma 9.85 above. □

Now we state the main theorem of this subsection:

Theorem 9.87. *Suppose that $k \in \mathbb{N}$ and f is obtained from g and h using k-bounded number recursion. Then f can be obtained from g and h by taking* $\mathbf{FAC}^0$ *closure and number recursion using $(k+1)$-permutations.*

Proof Sketch. We prove by induction on k. The base case $(k=0)$ is trivially true. For the induction step, suppose that h is not a $(k+1)$-permutation. The idea is to identify the first point m where h is not a permutation. Then h_x is a $(k+1)$-permutation for $x \le m$, and for $x > m$ we can use Lemma 9.86 above and the induction hypothesis. Formally, let

$$m(x) = \begin{cases} \min\{u \le x : h_u \notin S_{k+1}\} & \text{if } \exists u \le x\ h_u \notin S_{k+1} \\ x+1 & \text{otherwise} \end{cases}$$

Let $h'(x,u,z)$ be the $(k+1)$-permutation:

$$h'(x,u,z) = \begin{cases} h(u,z) & \text{if } u < m(x) \\ z & \text{otherwise} \end{cases}$$

Let $f'(x,u)$ be defined as

$$f'(x,0) = g$$
$$f'(x,u+1) = h'(x,u,f'(x,u))$$

Then $f(u) = f'(x,u)$ for $u \le m(x)$. In particular, $f(x) = f'(x,x)$ if $m(x) = x+1$.

For the case $m(x) \le x$, define

$$h''(x,u,z) = h(m(x)+u,z), \qquad \text{for } u \ge 0$$

Then $h''_{x,0} \notin S_{k+1}$. Let $f''(x,u)$ be defined as

$$f''(x,0) = f'(x,m(x))$$
$$f''(x,u+1) = h''(x,u,f''(x,u))$$

Then $f(u) = f''(x,u+m(x))$ for $u \ge 0$. In conclusion, we have

$$f(x) = \begin{cases} f'(x,x) & \text{if } m(x) = x+1 \\ f''(x,x-m(x)) & \text{otherwise} \end{cases}$$

Now since $h''_{x,0} \notin S_{k+1}$, by Lemma 9.86 and then the induction hypothesis, f'' can be defined from f' and h'' by taking $\mathbf{FAC}^0$ closure and number recursion using $(k+1)$-permutation. $\square$

9.7.4 $\mathbf{FAC}^0(2)$, $\mathbf{FAC}^0(6)$ and $\mathbf{FNC}^1$

This subsection is devoted to the proof of the following theorem.

Theorem 9.88. **a)** $\mathbf{FAC}^0(2)$ *is precisely the class of functions obtained from* $\mathbf{FAC}^0$ *by finitely many applications of composition, string comprehension, and 1-bounded number recursion.*

b) $\mathbf{FAC}^0(6)$ *can be obtained from* $\mathbf{FAC}^0$ *by finitely many applications of composition, string comprehension, and 2-bounded number recursion.*

c) $\mathbf{FAC}^0(6)$ *is closed under 3-bounded number recursion.*

d) $\mathbf{FNC}^1$ *is closed under k-bounded number recursion, for any $k \in \mathbb{N}$.*

e) $\mathbf{FNC}^1$ *can be obtained from* $\mathbf{FAC}^0$ *by finitely many applications of composition, string comprehension, and 4-bounded number recursion.*

Proof. **a)** First we show that the functions in $\mathbf{FAC}^0(2)$ can be obtained from $\mathbf{FAC}^0$ by finitely many applications of composition, string comprehension, and 1-bounded number recursion. By Definition 9.38 (for $m = 2$) and Theorem 9.6, it suffices to show that mod_2 can be obtained from $\mathbf{FAC}^0$ functions by composition, string comprehension and 1-bounded number recursion. In fact,

$$mod_2(0,X) = 0$$
$$mod_2(y+1,X) = (X(y) + mod_2(y,X)) \mod 2$$

For the other direction, we prove:

Claim If g and h are $\mathbf{\Sigma}^1_1$ definable in $\mathbf{V}^0(2)$, and f is obtained from g and h using 1-bounded number recursion, then f is also $\mathbf{\Sigma}^1_1$ definable in $\mathbf{V}^0(2)$.

To prove the claim, we will give a high level argument showing that if g and h are $\mathbf{FAC}^0(2)$ functions, then so is f. It is straightforward to formalize this argument in $\mathbf{V}^0(2)$, and hence the claim follows. Suppose that $f(x) \leq 1$ for all x, and

$$f(0) = g$$
$$f(x+1) = h(x, f(x))$$

We can assume w.l.o.g. that $h(x, y) \leq 1$, for all x and $y \leq 1$. We abuse the notation a little by letting $h(-1, 0) = h(-1, 1) = g$. Then it is easy to check that $f(x) = 0$ iff there is z, $-1 \leq z < x$ such that $h(z, 0) = h(z, 1)$ and for all u, $z < u < x$, $h(u, 0) \neq h(u, 1)$, and either

- $h(z, 0) = h(z, 1) = 0$ and the number of u, $z < u < x$, such that $h(u, 0) \neq 0$ is even, or
- $h(z, 0) = h(z, 1) = 1$, and the number of u, $z < u < x$, such that $h(u, 0) \neq 0$ is odd.

b) As for the first direction of **a**, mod_2 can be obtained from $\mathbf{FAC}^0$ functions by 1-bounded number recursion. Similarly, mod_3 can be obtained from $\mathbf{FAC}^0$ function by 2-bounded number recursion. Using mod_2 and mod_3 it is easy to obtain mod_6. Thus by Definition 9.38 (for $m = 6$) and Theorem 9.6, every $\mathbf{FAC}^0(6)$ function can be obtained from $\mathbf{FAC}^0$ functions by finitely many applications of composition, string comprehension, and 2-bounded number recursion.

c) First we prove an easier result that $\mathbf{FAC}^0(6)$ is closed under 2-bounded number recursion. By Theorem 9.87, it suffices to show that $\mathbf{FAC}^0(6)$ is closed under number recursion using 3-permutations. In other words:

Claim Suppose that $g()$ and $h(x, z)$ are in $\mathbf{FAC}^0(6)$, $g() \leq 2$ and $h_x \in S_3$ for all x. Let f be obtained from g and h using number recursion. Then f is also in $\mathbf{FAC}^0(6)$.

Proof of the Claim. Write $f(x)$ as in (9.43) (page 259):

$$f(x) = h_{x-1} \circ h_{x-2} \circ \ldots \circ h_0(g)$$

Let A_3 be the normal subgroup of S_3 which consists of the even permutations, and e be the identity of S_3. Then $S_3 = \langle (0\ 1) \rangle A_3$, i.e., every element in S_3 is the product $\gamma \circ \sigma$ where $\gamma \in \{e, (0\ 1)\}$ and $\sigma \in A_3$. Thus we have

$$h_u = \gamma_u \circ \sigma_u$$

where $\gamma_u \in \{e, (0\ 1)\}$ and $\sigma_u \in A_3$, for all u. For $u < x$ let

$$\delta_u = \gamma_{x-1} \circ \ldots \circ \gamma_u$$

Notice that $\gamma_u = (0\ 1)^{\epsilon_u}$ where $\epsilon_u \in \{0, 1\}$ for all u. Hence δ_u can be computed in $\mathbf{FAC}^0(2)$ by computing $(\epsilon_{x-1} + \ldots + \epsilon_u) \mod 2$.

Now it is easy to see that

$$f(x) = (\prod_{u=x-1}^{u=0} (\delta_u \circ \sigma_u \circ \delta_u^{-1})) \circ \delta_0(g)$$

We compute $\eta_u = \delta_u \circ \sigma_u \circ \delta_u^{-1}$ simultaneously for all $u < x$. Here $\eta_u \in A_3$, and therefore is of the form

$$(0\ 1\ 2)^{\epsilon'_u}$$

where $\epsilon'_u \in \{0, 1, 2\}$, for all $u < x$. As a result, $\prod_{u=x-1}^{u=0} \eta_u$ can be computed in $\mathbf{FAC}^0(3)$ by computing $(\epsilon'_{x-1} + \ldots + \epsilon'_u) \mod 3$. This shows that $f(x)$ can be computed in $\mathbf{FAC}^0(6)$. □

Now we prove **c**. Again by Theorem 9.87, it suffices to show that $\mathbf{FAC}^0(6)$ is closed under number recursion using 4-permutations. So let $g()$ and $h(x, z)$ be in $\mathbf{FAC}^0(6)$, $g() \leq 3$ and $h_x \in S_4$ for all x. Let f be obtained from g and h using number recursion. We claim that f is also in $\mathbf{FAC}^0(6)$.

This claim is proved in two steps, each uses the same idea as the previous claim. First, as before we reduce the computation of $f(x)$ to the problem of computing the product $\prod_{u=x-1}^{u=0} \eta_u$, where $\eta_u \in A_4$ for all u. (A_4 is the normal subgroup of S_4 which consists of all the even permutations.) Next, note that $A_4 = \langle(0\ 1\ 2)\rangle V$, where $V \lhd A_4$ is the Klein group,

$$V = \langle(0\ 1)(2\ 3), (0\ 2)(1\ 3)\rangle$$

Using this fact we reduce the above problem to the problem of

- computing $\gamma_{x-1} \circ \ldots \circ \gamma_u$ for $u < x$, where $\gamma_v \in \langle(0\ 1\ 2)\rangle$ for all $v < x$, and
- computing $\rho_{x-1} \circ \ldots \circ \rho_0$, where $\rho_u \in V$ for all $u < x$.

The first product can be computed in $\mathbf{FAC}^0(3)$ by letting $\gamma_v = (0\ 1\ 2)^{\epsilon_v}$ (where $\epsilon_v \in \{0, 1, 2\}$) and computing $(\epsilon_{x-1} + \ldots + \epsilon_u) \mod 3$. The second product can be computed in $\mathbf{FAC}^0(2)$ since V is Abelian and its members have order ≤ 2.

d) Note that for a constant $k \in \mathbb{N}$, any function $h \in {}^{(k+1)}(k+1)$ can be encoded using at most $(k+1)^2$ bits, e.g., $p_{x,y}$ is true iff $h(x) = y$, for $x, y \leq k$. Hence the composition $h_1 \circ h_2$ can be done uniformly by a circuits of constant depth and constant fanin (the constants depend on k).

Now suppose that $g \leq k$ and $h_x \in {}^{(k+1)}(k+1)$ are $\mathbf{NC}^1$ functions, and that f is obtained from g and h using number recursion. We need to show that f is also in $\mathbf{FNC}^1$. Recall that to compute $f(x)$ (for $x \geq 1$) we need to compute the composition

$$h_{x-1} \circ \ldots \circ h_0(g)$$

We implement the divide-and-conquer technique: Using the *MFVP* axiom, it is easy to $\mathbf{\Sigma}_1^1$-define in $\mathbf{VNC}^1(h)$ the function $r(x, z)$ so that (write $r_x(z)$ for

$r(x, z)$:

$$\begin{aligned} r_{x+u} &= h_u && \text{for } 0 \le u < x \\ r_u &= r_{2u} \circ r_{2u+1} && \text{for } 0 < u < x \end{aligned}$$

Then $f(x) = r_1(g)$. □

9.8 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 [**?**].

An analogue of the $\mathbf{\Sigma}_0^B$ Representation Theorem 4.17 for $\mathbf{TC}^0$ using the *threshold quantifier* in two-sorted logic can be found in [**?**]. It is shown that $\mathbf{TC}^0$ is exactly the class of relations represented by $\mathbf{\Sigma}_0^{B,Th}$ formulas, where $\mathbf{\Sigma}_0^{B,Th}$ is the class of formulas built in the same way as $\mathbf{\Sigma}_0^B$, except now we allow threshold quantifiers in addition to bounded number quantifiers.

The proof of $\mathbf{VTC}^0 \subseteq \mathbf{VNC}^1$ (Theorem 9.52) formalizes the arguments from [**?**]. Another arguments that can be formalized is to use the so-called ambiguous arithmetic notation. See for example [**?**] (CHECK: Section 5.4 or 5.5).

The theory $\mathbf{V}^1$-**KROM** is introduced in [**?**] and [**?**], and is shown to characterize **NL**. The proof of Theorem 9.68 is a modification of the proofs from [**?**] and [**?**] which use this result to show that $\mathbf{V}^0 \subseteq \mathbf{V}^1$-**KROM** (Lemma 9.67).

The Path problem is also known as the *Reachability*, or *GAP* problem.

Lind's original characterization of **FL** [**?**] is stated for string functions. Our number recursion in Subsection 9.7.1 corresponds to his *log bounded recursion on notation*.

The results in Subsection 9.7.3 are essentially from [**?**]. Parts **a** and **b** of Theorem 9.88 are two-sorted statement of the results from in [**?**]. Part **e** of that theorem is proved using an idea from [**?**].

DISCUSS THEORIES FOR NL AND L FROM CLOTE AND TAKEUTI

Chapter 10

Proof Systems for Small Theories

Appendix A

Computation Models

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 $\lceil \log_2 n \rceil$.

A.1 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 $\mathsf{\Gamma}$. 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 A.1. *For a natural number $k \geq 1$, a k–tape DTM M is specified by a tuple $\langle \mathsf{Q}, \mathsf{\Sigma}, \mathsf{\Gamma}, \sigma \rangle$ 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) $\mathsf{\Sigma}$ is the finite, non-empty set of input symbols.
3) $\mathsf{\Gamma}$ is the finite set of working symbols, $\mathsf{\Sigma} \subset \mathsf{\Gamma}$. It contains a special symbol $\not b$ (read "blank"), and $\not b \in \mathsf{\Gamma} \setminus \mathsf{\Sigma}$.

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 $s_1, \dots, s_k$, and $\sigma(q, \vec{s}) = (q', s'_1, h_1, \dots, s'_k, h_k)$, then q' is the new state, $\vec{s'}$ are the symbols printed, and for $1 \le i \le k$, 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.

Convention For a multi-tape Turing machine, we require that the input tape head is read–only. Also, unless specified otherwise, Turing machines are multi-tape.

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.

It is straightforward to extend the above definition to define Turing machine that accepts a relation or computes a function of some fixed arity. For example, to compute a partial function, if the function is defined on the input, the machine will halt in q_{accept} with the function value on the tape. Here it might be practical to introduce a distinct symbol # as a separator for the input arguments.

A *configuration* of M is a tuple $\langle q, u_1, v_1, \dots, u_k, v_k \rangle \in \mathsf{Q} \times (\Gamma^* \times \Gamma^*)^k$. The intuition is that q is the current state of the control, $u_i v_i$ is the non-blank (but possibly empty) content of the tape i, whose head is reading the first 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.

Formally we require that for each i, u_i does not start with the blank symbol $\not{b}$, and v_i does not end with $\not{b}$. This is to make sure that the content of the tape and the tape head position are uniquely represented by the pair $\langle u_i, v_i \rangle$. Although we allow trailing $\not{b}$ in u_i or leading $\not{b}$ in v_i, in practice, for each "meaningful" Turing machine the length of such blank segment are bounded by some constant.

The *computation* of M on an input x is the (possibly infinite) sequence of configurations of M, starting with the *initial configuration* $\langle \mathsf{q}_{initial}, \epsilon, x, \epsilon, \epsilon, \dots, \epsilon, \epsilon \rangle$, 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.

A.1.1 L, P, PSPACE and EXP

If 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 $x \in L(\mathsf{M})$, $time_{\mathsf{M}}(x) \le f(|x|)$, 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 A.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 A.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 A.4 (P and EXP).

$$\mathbf{P} = \bigcup_{k \ge 1} \mathbf{DTime}(n^k) \qquad \mathbf{EXP} = \bigcup_{k \ge 1} \mathbf{DTime}(2^{n^k})$$

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* 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 $x \in L(\mathsf{M})$, $space_{\mathsf{M}}(x) \le f(|x|)$. In this case we also say that M *accepts the language $L(\mathsf{M})$ in space $f(n)$*.

Definition A.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 A.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 polynomial space is defined as follows.

Definition A.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 A.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 A.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

$$\mathbf{L} \subseteq \mathbf{P} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}.$$

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 3 inclusions as shown above 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* $f(n)$. The space bounds that we are interested in are all constructible.

Theorem A.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

$$\mathbf{DSpace}(g) \setminus \mathbf{DSpace}(f) \neq \emptyset$$

A.2 Nondeterministic Turing Machines

Definition A.11. *A k–tape nondeterministic Turing machine (NTM) is specified by a tuple $\langle \mathsf{Q}, \Sigma, \Gamma, \sigma \rangle$ as in Definition A.1, with the modification that*

$$\sigma : ((\mathsf{Q} \setminus \{\mathsf{q}_{accept}, \mathsf{q}_{reject}\}) \times \Gamma^k) \to \mathcal{P}(\mathsf{Q} \times (\Gamma \times \{\mathsf{L}, \mathsf{R}\})^k)$$

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 such an accepting computation of length (i.e., the length of the sequence of configurations) $\leq f(|x|)$, and M accepts *x in space $f(n)$* if there is an accepting computation such that the number of squares visited at least once by the tape head of M 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 A.12 (NTime and NSpace). *For a function $f(n)$, define*

$$\begin{aligned} \mathbf{NTime}(f) &= \{L : \textit{there is a NTM accepting } L \textit{ in time } f(n)\} \\ \mathbf{NSpace}(f) &= \{L : \textit{there is a NTM accepting } L \textit{ in space } f(n)\} \end{aligned}$$

The Speed-up Theorem (A.3) and Tape Compression Theorem (A.6) continue to hold for NTMs.

Definition A.13 (NP and NL).

$$\mathbf{NP} = \bigcup_{k \geq 1} \mathbf{NTime}(n^k), \qquad \mathbf{NL} = \mathbf{NSpace}(\log n)$$

It is straightforward that

$$\mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}$$

However, resolving whether these are proper containments is a major open problem in Computer Science.

For a class **C** of languages, we define *co*-**C** to be the class of the complements of the languages in **C**. It is also easy to see that

$$\mathbf{P} \subseteq co\text{-}\mathbf{NP} \subseteq \mathbf{PSPACE}$$

But the questions $\mathbf{P} \overset{?}{=} co\text{-}\mathbf{NP}$, $\mathbf{NP} \overset{?}{=} co\text{-}\mathbf{NP}$ and $co\text{-}\mathbf{NP} \overset{?}{=} \mathbf{PSPACE}$ are open.

For **NL** vs. *co*-**NL** we have an affirmative answer, due to Immerman and Szelepcsényi:

Theorem A.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 A.15 (Savitch's Theorem). *For any space constructible function $f(n) \geq \log n$,*

$$\mathbf{NSpace}(f) \subseteq \mathbf{DSpace}(f^2)$$

Note that it also follows that $\mathbf{NL} \subsetneq \mathbf{PSPACE}$.

A.3 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 enter 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.

The running time of M on an input x is defined as before. Note that the time it takes to write down the queries 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 **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 A.16 (PH). $\mathbf{\Delta}_0^p = \mathbf{\Sigma}_0^p = \mathbf{\Pi}_0^p = \mathbf{P}$. *For* $i \geq 0$,

$$\mathbf{\Sigma}_{i+1}^p = \mathbf{NP}(\mathbf{\Sigma}_i^p), \qquad \mathbf{\Pi}_{i+1}^p = co\text{-}\mathbf{\Sigma}_{i+1}^p, \qquad \mathbf{\Delta}_{i+1}^p = \mathbf{P}(\mathbf{\Sigma}_i^p)$$

And

$$\mathbf{PH} = \bigcup_{i \geq 0} \mathbf{\Sigma}_i^p$$

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 A.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 **C** of languages, let **NLinTime**(**C**) be the class of languages accepted by a linear time Oracle TM with oracle from **C**. Then the Linear Time Hierarchy is defined as follows.

Definition A.18 (LTH).

$$\mathbf{\Sigma}_0^{lin} = \mathbf{LinTime}, \qquad \mathbf{\Sigma}_{i+1}^{lin} = \mathbf{NLinTime}(\mathbf{\Sigma}_i^{lin}) \textit{ for } i \geq 0, \qquad \mathbf{LTH} = \bigcup_{i \geq 0} \mathbf{\Sigma}_i^{lin}$$

Both **PH** and **LTH** can be alternatively defined using the notion of *alternating Turing machines*, which we will define in the next Section.

A.4 Alternating Turing Machines

An *alternating Turing machine* (ATM) M is defined as in Definition A.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.

Now a computation of M on x is an *accepting computation* if it consists only of accepting configurations of M. We say that an ATM M accepts x iff it contains at least one accepting computation. Note 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.

If an ATM M accepts x then the running time $time_{\mathsf{M}}(x)$ of M on x is the length of the shortest accepting computation of M on x, otherwise $time_{\mathsf{M}}(x) = \infty$.

A.4.1 $\mathbf{NC}^1$ and $\mathbf{AC}^0$

Definition A.19 ($\mathbf{AC}^0$). *DEFINE* $\mathbf{AC}^0$ *USING CIRCUIT CLASS, CHANGING THIS DEFINITION MAY CHANGE DISCUSSION IN THE PROOF OF* $\mathbf{\Sigma}_0^B$ *REPRESENTATION THEOREM.*

Theorem A.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

A.5 Implementation of Multiplication

Suppose that we are to multiply two numbers x, y whose binary representations are $x_{n-1}\ldots x_0$, $y_{n-1}\ldots y_0$. The "school algorithm" is to write down a matrix which has n rows of the form $0\ldots 0x_{i,n-1}\ldots x_{i,0}0\ldots 0$ ($n-i$ 0's in the front, i 0's follow $x_{i,0}$), for $i = 0,\ldots,n-1$, where $x_{i,j} = 0$ if $y_i = 0$, and $x_{i,j} = x_j$ otherwise. The next step is to add each column of this matrix, starting from the right, remembering the carries.

This method seems not applicable if we are using alternating Turing machines with constant alternations, linear time, or when we are working within "low" complexity classes, such as $\mathbf{TC}^0$. Since the ability to carry out multiplication using such limited resources is crucial in different discussions, we will present a method which is applicable in these situations. The idea is to avoid adding columns of the matrix one by one. As can be seen, it suffices to compute the sum of n numbers, each has a binary representation of length n. We will address this issue first.

A.5.1 Adding n Numbers of Length n

Suppose that we are to compute the sum of n numbers, $z = x^0 + \ldots + x^{n-1}$, where the binary representation of x^i is $x^i = x^i_{n-1} \ldots x^i_0$, for $i = 0, \ldots, n-1$. Imagining that these numbers form the rows of an $n \times n$ matrix. We will divide this matrix into $2m$ blocks, each has ℓ columns, where $\ell = \lceil 2 \log n \rceil$, and $m = \lceil n/(2\ell) \rceil$. Informally, we will add the "odd" and "even" blocks separately, and then add the two sums to get the result.

Formally, for $i < n$ let

$$u^i = x^i_{(2m-1)\ell-1} \ldots x^i_{(2m-2)\ell} \underbrace{0 \ldots 0}_{\ell} \ldots \underbrace{0 \ldots 0}_{\ell} x^i_{\ell-1} \ldots x^i_0$$

$$v^i = x^i_{2m\ell-1} \ldots x^i_{(2m-1)\ell} \underbrace{0 \ldots 0}_{\ell} \ldots x^i_{2\ell-1} \ldots x^i_{\ell} \underbrace{0 \ldots 0}_{\ell}$$

then $x^i = u^i + v^i$.

Let $u = \sum_{i=0}^{n-1} u^i$ (sum of the "even" blocks), and $v = \sum_{i=0}^{n-1} v^i$ (sum of the "odd" blocks), then $z = u + v$. The advantage of getting u, v separately is that in calculating u and v, the sum of a non-zero block do not carry to the next non-zero block. This property allows "fast, parallel" computation of u and v.

A.5.2 Multiplication in LTH

Now we will show that multiplication can be done in **LTH** by showing that the relation $z = x \cdot y$ can be checked in **LTH** (to compute $x \cdot y$ in **LTH**, just guess a value z and then perform the checking). We will unwind the algorithm presented in A.5.1.

Suppose that the length of x and y are n. In this case, we will add n numbers of length $2n$, i.e., our matrix will have rows of the form $x^i_{2n-1} \ldots x^i_0$, for $i = 0, \ldots, n-1$. We will use the block size $\ell = \lceil 2 \log n \rceil$, and there are $2m$ blocks, where $m = \lceil n/\ell \rceil$. The values of ℓ and m can be guessed and checked in time $\Omega(n)$.

Next, guess the values (in binary representation) of u and v. Note that the lengths of u, v are bounded by $2n + \ell$. Then, we have to check that u is the sum of the "even" blocks, and v is the sum of the "odd" blocks. For symmetry, we will just give an **LTH** algorithm to check for the correctness of u.

Let $u = u_{2m\ell-1} \ldots u_0$. We have to check that for each $j < m$,

$$u_{(2j+2)\ell-1} \ldots u_{2j\ell} = \sum_{i=0}^{n-1} x^i_{(2j+1)\ell-1} \ldots x^i_{2j\ell}. \tag{A.1}$$

(I.e., entering universal states to check Equation A.1 for all $j < m$.) Equation A.1 can be checked using the usual addition algorithm, i.e., adding column by column, with the carries from the previous ones. It can be carried out in **LTH** as follows.

Guess ℓ "sums" $s^{\ell-1}, \ldots, s^0$, each of length ℓ (i.e., entering ℓ^2 existential states). Each s^k is intended to be the sum of the column k (i.e., $x^0_{2j\ell+k} +$

$\ldots + x^{n-1}_{2j\ell+k}$) and the appropriate bits of $s^{k-1}, \ldots, s^0$. This can be checked by entering universal states, checking that for each $k \geq 1$:

$$s^k = x^0_{2j\ell+k} + \ldots + x^{n-1}_{2j\ell+k} + s^{k-1}_1 + \ldots + s^0_k. \tag{A.2}$$

($s^0 = 0$.) The RHS of Equation A.2 is represented in unary string of length at most $n + \ell \leq 2n$. Its unary representation can be computed in time linear in n (note that each $x^i_{2j\ell+k}$ depends on y_i and $x_{2j\ell+k}$).