Witnessing Theorems in Bounded Arithmetic and Applications

Jean-José Razafindrakoto

Submitted to Swansea University in fulfilment of the requirements for the Degree of Doctor of Philosophy

Swansea University
Prifysgol Abertawe

Department of Computer Science
Swansea University
2016
Declaration

This work has not been previously accepted in substance for any degree and is not being concurrently submitted in candidature for any degree.

Signed ............................................................ (candidate)

Date ............................................................

Statement 1

This thesis is the result of my own investigations, except where otherwise stated. Other sources are acknowledged by footnotes giving explicit references. A bibliography is appended.

Signed ............................................................ (candidate)

Date ............................................................

Statement 2

I hereby give my consent for my thesis, if accepted, to be available for photocopying and for inter-library loan, and for the title and summary to be made available to outside organisations.

Signed ............................................................ (candidate)

Date ............................................................
A total search problem is a binary relation $R(x,y)$ such that $\forall x \exists y R(x,y)$ holds. The search task is as follows: given an $x$, find a solution $y$ such that $R(x,y)$ holds. The most studied class of total search problems is the class of total functions in NP (TFNP), which consists of those total search problems $R(x,y)$ that are polynomial-time computable and honest (that is to say, the length of $y$ is bounded by a polynomial in the length of $x$). The class TFNP contains a host of problems of practical importance, such as the traveling salesman problem or the problem of finding a pure Nash equilibrium in a congestion game.

This thesis is concerned with the study of total search problems, whose totality can be expressed by a formula in $\forall \Sigma^b_1$ and provable in some Bounded Arithmetic theory corresponding to some complexity class between $\text{AC}^0(m)$ and PSPACE. These total search problems are called provably total search problems.

Bounded Arithmetic is closely related to Complexity Theory. From the point of view of Bounded Arithmetic, the set of all true formulae in $\forall \Sigma^b_1$ defines TFNP. The Bounded Arithmetic theories that are of interest to us in this thesis capture the complexity classes from $\text{AC}^0(m)$ to PSPACE and separating them may shed light on the separation of the corresponding complexity classes. It is conjectured that these theories in question are distinct and that they can be separated by formulae in $\forall \Sigma^b_1$. For this reason, it is important to study their provably total search problems and characterize them in terms of subclasses of TFNP. The witnessing theorem method is available to study such sentences: show that if a formula $\forall x \exists y R(x,y)$ in $\forall \Sigma^b_1$ is provable in a theory, then finding a solution to $R(x,y)$ is reducible to finding a solution to some TFNP problem.

In this thesis, we provide characterizations of the provably total search problems of bounded arithmetic theories corresponding to complexity classes from $\text{AC}^0(m)$ to PSPACE in terms of subclasses of TFNP. One direction of our characterizations is obtained via the witnessing theorem method, where we require that the reduction is provable in some weak base theory. For the theories corresponding to complexity classes below PH, our characterizations are in terms of total search problems whose solutions can be verified by an $\text{AC}^0$ procedure. Furthermore, for these theories, we show that the reduction is always provable in the theory $\forall^0$ corresponding to $\text{AC}^0$. However, for the theory $U^1_2$ corresponding to PSPACE, our characterizations are in terms of subclasses of TFNP, where the reduction is provable in the theory $S^1_2$ corresponding
to $P$. This last result begs the question whether the class of provably total search problems of $U_2^1$ (and theories corresponding to complexity classes beyond $\text{PSPACE}$) can be characterized in terms of a subclass of $\text{TFNP}$ whose solutions can be verified by a procedure from some complexity class below $P$, or even $\text{AC}^0$. Additionally, it leaves us with the question whether the reduction can be proven in a theory that is weaker than $S_2^1$, such as $V^0$. 
Acknowledgements

I am forever grateful to my adviser Arnold Beckmann for his endless support over the years, which was vital in making this thesis a reality. Arnold sparked my interest in mathematical logic, bounded arithmetic and proof complexity. He is not only an amazing researcher, but also an inspiring teacher and a role model.

I would like to thank the members of the Computer Science Department at Swansea University for their kindness and support – in particular, Ulrich Berger who is truly a good person. Furthermore, I would like to thank the exam team – Olaf Beyersdorff and Faron Moller – for the time and effort they invested into the examination of this thesis.

I thank God for all He has done for me. He made a wonderful mother and He gave that dear mother to me. And I wish to thank my mother Clemence Rakotonirina who has always been there for me, through the good and the bad times. She is my hero and this thesis is dedicated to her.
# Table of Contents

1 Introduction........................................... 1
   1.1 Aims of the Thesis................................ 5
   1.2 Overview of the Results of the Thesis........... 7
   1.3 Organization of the Thesis......................... 10

2 Preliminaries........................................... 11
   2.1 Two-sorted Logic and Complexity Classes......... 11
   2.2 The Theory $V^0$ for $AC^0$....................... 18
   2.3 Theories for Complexity Classes Below Polynomial-time...... 25
   2.4 Theories for the Levels of PH and Search Problems........ 26

3 Provably Total Search Problems below Polynomial-Time.. 31
   3.1 The Theory $VC$ and $KPTC$....................... 32

4 Provably Total Search Problems for Polynomial-time..... 39
   4.1 Inflationary Polynomial Local Search and Iteration Problems.................. 40
   4.2 Constructions of Inflationary Iteration Problems.................. 47
   4.3 Inflationary Polynomial Local Search and $V^1$.................. 51

5 Generalized Polynomial Local Search and Improved Witnessing Theorem..... 59
   5.1 Generalized Polynomial Local Search and Iteration Problems.................. 60
   5.2 Generalized Iteration Problems and $TV^w$.................. 62
   5.3 Some More Constructions of Generalized Iteration Problems.................. 64
   5.4 Proof of the New-style Witnessing Theorem for $TV^k$.................. 68
   5.5 Discussion on Skolemized Iteration Problems.................. 72

6 Extended Linear Local Improvement Principles and $U^1_2$........ 77
   6.1 Preliminaries........................................ 78
   6.2 Witnessing Theorem for $WT$........................ 84
   6.3 Improved New-style Witnessing Theorem for $U^1_2$............ 87
6.4 Extended Linear Local Improvement Principles ........................................ 98

7 Conclusion .......................................................................................... 109
CHAPTER 1

Introduction

Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Aims of the Thesis</td>
<td>5</td>
</tr>
<tr>
<td>1.2</td>
<td>Overview of the Results of the Thesis</td>
<td>7</td>
</tr>
<tr>
<td>1.3</td>
<td>Organization of the Thesis</td>
<td>10</td>
</tr>
</tbody>
</table>

Robinson arithmetic (Q) is a set of axioms that provides recursive definitions for addition (+) and multiplication (·), and some basic properties of ≤. By augmenting Q with the axiom scheme of induction IND(a, ϕ), which is:

\[ \neg \phi(0) \lor (\exists x < a)[\phi(x) \land \neg \phi(x + 1)] \lor \phi(a), \]

where ϕ is an arbitrary formula, we obtain Peano arithmetic (PA). This thesis is concerned with the study of Bounded Arithmetic (BA), a collective noun for subtheories of Peano arithmetic, which gives a proof-based approach to Complexity Theory: BA connects the study of feasible computability\(^1\) with questions about provability and axiomatizability.

Historically, the study of BA was initiated by Parikh [Par71], when he introduced the first-order theory IΔ₀, which is PA but with induction IND(a, ϕ) restricted to those formulae ϕ in which only bounded quantifiers are occurring. Later, Buss [Bus86] introduced the much-studied hierarchy

\[ S^n_1 \subseteq T^n_1 \subseteq S^n_2 \subseteq \ldots \subseteq S^n_i \subseteq T^n_i \subseteq S^{i+1}_2 \subseteq \ldots \]

of first-order theories corresponding to the levels of the Polynomial Hierarchy (PH) and called the union of these theories S₂, that is to say,

\[ S_2 = \bigcup_{i=1}^{\infty} S^i_2 = \bigcup_{i=2}^{\infty} T^i_2. \]

\(^1\)Feasible computability is the study of functions that can be computed by today’s computers (or tomorrow’s).
1. Introduction

Along with $S_2$, Buss also introduced the second-order theories $U_1^2$ and $V_2^2$, corresponding to PSPACE and EXPTIME\(^2\) respectively [Bus86]. A host of theories corresponding to complexity classes below $P$ have also been developed by other researchers. However, the early standard theories for these small complexity classes below $P$ suffer from the fact that their vocabularies include integer multiplication as a primitive function, which is known to not be in $AC^0$, for example. Because of that, it is awkward to turn them into theories for these small complexity classes. As a result, a new generation of second-order BA theories, $V_0 \subset V_1 \subset V_{TC} \subset V_{NC} \subset V_{L} \subset V_{NL} \subset V_{NC} \subset V_{P} \subset V_{1} \subset V_{2} \subset \ldots$, capturing complexity classes from $AC^0$ to $PH$, were introduced [CK03, CK04, Coo04, CM05, NC06, Ngu08, CN10] (motivated by an earlier work of Zambella [Zam96]) and reported by Cook and Nguyen in [CN10]. This second-order “viewpoint” has the advantages of greatly reducing the number of axioms required to define the theories and also simplifying the bootstrapping of the theories. Via some notion of isomorphism [Tak93] between first- and second-order theories, the theory $V_i$ (respectively, $TV_i$) can be viewed as the second-order version of Buss’s first-order theory $S_i^2$ (respectively, $T_i^2$), for $i \geq 1$.

For decades, researchers have been attempting to separate complexity classes, including those within the following sequence:

\[
AC^0 \subset AC^0(m) \subset TC^0 \subset NC^1 \subset L \subset NL \subset NC \subset P \subset NP \subset PH \subset PSPACE
\]  

(1.1)

To this end, we know, for example, that $NL \not\subset PSPACE$ from standard results in Complexity Theory. Also, Razborov and Smolensky [BS90] showed that, for any $m \geq 2$ and any prime $p$,

\[
AC^0(m) \not\subset AC^0(p)
\]

unless $m$ is a power of $p$. However, for any other two complexity classes in (1.1), excluding $AC^0$, their situation has not been resolved yet. For instance, it is possible that $PH = P$, but it is consistent with our present knowledge that $AC^0(6) = PH$. This is one reason for studying BA theories corresponding to these classes in (1.1) with the hope that a proof of the separation of the theories may shed light on the separation of the corresponding complexity classes.

Another reason for studying BA is due to its close relationship to Propositional Proof Complexity, an area of research that studies the lengths of proofs in proof systems for propositional tautologies and that is fundamentally connected to major questions in Complexity Theory. Historically, Cook [Coo75] was the first to show a connection between BA and Propositional Proof Complexity via the notion of propositional translation. More precisely, Cook showed how theorems of a theory corresponding to $P$ can be translated into families of tautologies that have polynomial-size proofs in a well studied propositional proof system called extended

---

\(^2\)EXPTIME is the class of relations which are exponential-time computable.

\(^3\)Bootstrapping is the process of establishing basic facts, such as commutativity and associativity of addition, transitivity of $\leq$, totality of subtraction, etc.

\(^4\)Cook and Reckhow [CR79] showed that the existence of a propositional proof system in which every tautology has a “short” proof is equivalent to the $NP$ vs $coNP$ question.
Finally, of central importance to the study of BA is Buss’s hierarchy $S_2$, because of its tight relationship to PH. Intuitively, the theory $T_2^i$ is obtained by modifying $I\Sigma_0$ so that induction $\text{IND}(a, \varphi)$ is over a formula $\varphi$ in the class $\Sigma^b_2$, which represents the relations in the level $\Sigma^b_i$ of PH; whereas the theory $S_2^i$ is similar to $T_2^i$ but with $\text{IND}(|a|, \varphi)$ instead of $\text{IND}(a, \varphi)$, where $|a|$ denotes the length of the binary representation of $a$. In fact, the relationship between $S_2$ and PH goes further than that. Already in [Bus86], Buss showed that the $\Sigma^b_i$-definable functions in $S_2^i$ are precisely those functions in $\text{FP}^{\Sigma^b_i}$. Later, finer information regarding the relationship between $S_2$ and PH was proven by Krajíček, Pudlak and Takeuti [KPT91]. Namely, they showed that if $T_2^i = S_2^{i+1}$, for any $i \geq 1$, then PH collapses. Subsequently, Buss [Bus95], and, independently, Zambella [Zam96], strengthened the results of [KPT91] by proving that if $T_2^i = S_2^{i+1}$, then $T_2^i = S_2$ and $T_2^i$ proves that PH collapses. As a result, $S_2$ proves that PH collapses if and only if $S_2 = S_2^i$. Since $S_2$ is finitely axiomatizable, the question of whether $S_2$ collapses or not is equivalent to the question about the finite axiomatizability of $S_2^i$.

Whether or not $S_2$ is finitely axiomatizable is still an open problem. Despite a great amount of research towards a resolution of this problem, the only thing we know is that $S_2^{i+1}$ is $\forall\Sigma^b_{i+1}$-conservative\(^5\) over $T_2^i$ [Bus90], but it is not $\forall\Sigma^b_{i+2}$-conservative, unless $\Sigma^b_{i+2} = \Pi^b_{i+2}$ [KPT91]. Furthermore, $T_2^i$ is not $\forall\Sigma^b_{i+1}$-conservative over $S_2^i$, unless some complexity classes $P^{\Sigma^b_i}[O(\log n)]$ and $\Delta^b_{i+1}$ coincide [Kra93]. It is worth pointing out here that it is unlikely that, for all $i \geq 1$, $T_2^i$ is $\forall\Sigma^b_i$-conservative over $S_2^i$, since that would imply that the $\forall\Sigma^b_i$-theorems\(^6\) of $T_2^i$ are the same as the $\forall\Sigma^b_i$-theorems of $S_2^i$, which would have profound consequences for Complexity Theory: many important problems such as finding a pure Nash equilibrium in a congestion game would have polynomial-time solutions.

The question of whether $S_2$ is finitely axiomatizable or not and questions about the relationship between $S_2^i$ and $T_2^i$ are more accessible in the relativized setting, when the theories are extended with an uninterpreted predicate symbol $\alpha$ – from a computer science viewpoint, $\alpha$ represents an oracle. In this relativized setting, Krajíček, Pudlak and Takeuti [KPT91] showed that the $T_2^i(\alpha)$ is strictly contained within $S_2^{i+1}(\alpha)$. Subsequently, Krajíček showed that the inclusions between $S_2^i(\alpha)$ and $T_2^i(\alpha)$ are also proper [Kra93]. Now, in terms of conservativity results, the relationship between $S_2^{i+1}(\alpha)$ and $T_2^i(\alpha)$ is well understood: $S_2^{i+1}(\alpha)$ is $\forall\Sigma^b_{i+1}(\alpha)$-conservative over $T_2^i(\alpha)$, but not with respect to larger classes of formulæ. However, the relationship between $S_2^i(\alpha)$ and $T_2^i(\alpha)$ is less clear (at least for $i \geq 3$, since $S_2^i(\alpha)$ is

---

\(^5\)Let $\mathcal{R}_1$ and $\mathcal{R}_2$ be two theories such that $\mathcal{R}_1 \subseteq \mathcal{R}_2$. Then $\mathcal{R}_2$ is $\Phi$-conservative over a theory $\mathcal{R}_1$ if they have the same $\Phi$-theorems.

\(^6\)A $\Phi$-theorem (or $\Phi$-consequence) $\varphi$ of a theory $\mathcal{T}$ is a theorem of $\mathcal{T}$ such that $\varphi \in \Phi$. 
∀Σ^b_1(α)-separated\(^7\) from \(T^1_2(α)\) [Pud92, Kra92, BK94]\(^8\) and \(S^2_2(α)\) is also ∀Σ^b_1(α)-separated from \(T^2_2(α)\) [CK98, CK99]. But, in general, the best known result is that \(T^1_2(α)\) is not ∀Σ^b_1(α)-conservative over \(S^2_2(α)\) [BK94].

**Conjecture 1.1** \(T^1_2(α)\) is not ∀Σ^b_1(α)-conservative over \(S^2_2(α)\).

A promising approach towards solving the above conjecture is to characterize the ∀Σ^b_1-theorems of \(T^2_2\) in terms of a class of total search problems within TFNP [MP91]. By a characterization of the ∀Σ^b_1-theorems of \(T^2_2\), we mean a class \(\mathcal{S}\), which is many-one complete for the ∀Σ^b_1-theorems of \(T^2_2\).

A total search problem is a relation \(R(x,y)\) such that \((\forall x)(\exists y)R(x,y)\) is true. The search task is: given an \(x\), find a solution \(y\) such that \(R(x,y)\) holds. One of the most important classes of search problems is TFNP, consisting of total search problems \(R(x,y)\) which are polynomial-time computable and such that if \(R(x,y)\) holds, then the length of the binary representation of \(y\) is bounded by a polynomial in the length of the binary representation of \(x\).

The class TFNP has been extensively studied from the point of view of Complexity Theory and contains a host of important problems. For instance, problems in \(\text{NP} \cap \text{coNP}\) and the Polynomial Local Search (PLS) problems of Johnson, Papadimitriou and Yannakakis [JPY88] are all examples of TFNP problems. Additionally, it contains numerous combinatorially defined problems [PY88, Pap90, Pap94]. From the point of view of BA, the set of all true ∀Σ^b_1-formulae defines TFNP. Thus, an important step towards a resolution of Conjecture 1.1 is to characterize the ∀Σ^b_1-theorems of \(T^2_2\) in terms of a subclass of TFNP. This activity is not only a good way to generate new and interesting subclasses of TFNP, but it also provides tools for showing separations of the relativized versions of the search problem classes.

Towards a characterization of the ∀Σ^b_1-theorems of \(T^2_2\), for all \(i \geq 1\), Buss and Krajíček [BK94] showed that the set of all ∀Σ^b_1-theorems of \(T^1_2\) is characterized by PLS. Then Krajíček, Skelley and Thapen [ST11] characterized the ∀Σ^b_1-theorems of \(T^2_2\) in terms of some kind of a generalized PLS that they called colored PLS. Additionally, they gave characterizations of the ∀Σ^b_1-theorems of \(T^3_2\) in terms of a reflection principle and some kind of a recursion called verifiable recursion. Subsequently, a characterization of the ∀Σ^b_1-theorems of \(T^3_2\), for all \(i \geq 1\), based on \(\ell\)-turn games was given by Skelley and Thapen [ST11]. Independently, Beckmann and Buss [BB09] introduced a new class of search problems by using a relativized notion of PLS and called this class \(\Pi^p_2\)-PLS with \(\Pi^b_2\)-goal. They then argued that the ∀Σ^b_1-theorems of \(T^2_2\) are characterized by \(\Pi^p_2\)-PLS with \(\Pi^b_{g-1}\)-goal, for \(1 \leq g \leq i+1\). Other characterizations of the ∀Σ^b_1-theorems of the whole bounded arithmetic hierarchy appeared in [Pud06, Pud08, KNT11, Tha11, PT12].

The characterizations mentioned above concern the ∀Σ^b_1-theorems of the theories in \(S_2\). However, Kołodziejczyk et al. [KNT11] extended this kind of characterization to the second-order theories \(U^1_2\) and \(V^1_2\), corresponding to \(\text{PSPACE}\) and \(\text{EXPTIME}\) respectively. More precisely, they showed that the ∀Σ^b_1-theorems of \(V^1_2\) are characterized by the local improvement principle LI. Furthermore, they showed that the linear local improvement principle \(\text{LII}_{\log}\) characterizes the ∀Σ^b_1-theorems of \(U^1_2\). Subsequently, Beckmann and Buss [BB14] improved on the

---

\(^7\)We say that two theories are Φ-separated if they do not have the same Φ-theorems.

\(^8\)Buss also showed that \(S^3_2(α)\) and \(T^1_2(α)\) is ∀Σ^b_1(α)-separated in an unpublished work.
results of [KNT11] by showing that $U^1_2$ already proves the linear local improvement principle $LL1$ (therefore showing that $LL1$ is equivalent to $LL1_{log}$) and that the local improvement principle $LL1_{log}$ is equivalent to $LI$. Very recently, other characterizations of the $\forall \Sigma^1_i$-theorems of $U^1_2$ and $V^1_2$ have appeared in [BB, Kra15].

Earlier, we mentioned that a new generation of second-order BA theories

$$V^0 \subseteq V^0(m) \subseteq VTC^0 \subseteq VNC^1 \subseteq VL \subseteq VNL \subseteq VNC \subseteq VP \subseteq V^1 \subseteq TV^1 \subseteq V^2 \subseteq \ldots$$

corresponding to complexity classes from $AC^0$ to PH were introduced in [CK03, CK04, Coo04, CM05, NC06, Ngu08, CN10], motivated by earlier work of Zambella [Zam96]. In particular, we mentioned that via some notion of isomorphism between first- and second-order theories, the theory $V^i$ (respectively, $TV^i$) can be viewed as the second-order version of Buss’s first-order theory $S^i_2$ (respectively, $T^i_2$), for $i \geq 1$. Therefore, all the results we stated above regarding $S^i_2$ and $T^i_2$ also apply to $V^i$ and $TV^i$. Finally, we would like to point out that in this second-order setting, the class $\Sigma^i_B$ of Buss would correspond to the class $\Sigma^i_{TV}$, for $i \geq 1$, where the superscript “$B$” means that we are in the second-order setting.

Let us now focus on the theories within the sequence

$$V^0 \subseteq V^0(m) \subseteq VTC^0 \subseteq VNC^1 \subseteq VL \subseteq VNL \subseteq VNC \subseteq VP,$$

where the theory $V^0$ and $V^0(m)$ corresponds to $AC^0$ and $AC^0(m)$ respectively, and the theory $VC$ corresponds to the complexity class $C$, for $C = TC^0, NC^1, \ldots, P$. Intuitively, the theory $VC$ is axiomatized by the axioms of $V^0$ and a single axiom $Ax_C$, which asserts the existence of a solution for a complete problem in $C$, where $Ax_C$ is given by a $\Sigma^1_B$-formula. Because $Ax_C$ is given by a $\Sigma^1_B$-formula, these theories are conjectured to be $\forall \Sigma^B_i$-separated. Despite that, not much work has been done towards characterizing the $\forall \Sigma^B_i$-theorems of these theories, apart from the fact that their provably total functions are precisely those in $FC$. Instead, most of the recent work all tends towards separating them with respect to formulae of lower complexity, to the best of our knowledge; namely, of $\Sigma^0_B$-complexity, where $\Sigma^0_B$ is a class of formulæ representing relations in $AC^0$. This avenue of research, which is to $\Sigma^0_B$-separate those theories, is supported by the fact that the $\Sigma^0_B$-theorems of these theories translate into families of tautologies with polynomial-size proofs in the corresponding propositional proof systems. As an example, let us consider the pigeonhole principle $PHP^n$, which states that there is no one-to-one map from $\{0, 1, \ldots, n\}$ to $\{0, 1, \ldots, n-1\}$. Ajtai [Ajt94] showed that the tautologies $PHP^n$ do not have polynomial-size proofs in constant-depth Frege, a propositional proof system corresponding to $V^0$. It follows that $V^0$ does not prove the $\Sigma^0_B$-formula $PHP(x,X)$, associated with $PHP^n$. However, it is known that $VTC^0$ proves $PHP(x,X)$. Thus, $V^0 \subseteq VTC^0$. But it is still unknown whether $PHP(x,X)$ can be proven in $V^0(2)$ or not.

### 1.1 Aims of the Thesis

One of the main objectives of this thesis is to provide new characterizations of the $\forall \Sigma^1_B$-theorems of the second-order theories

$$V^0(m) \subseteq VTC^0 \subseteq VNC^1 \subseteq VL \subseteq VNL \subseteq VNC \subseteq VP \subseteq V^1$$


in terms of suitable classes of total search problems within TFNP. Furthermore, since none of the previously known characterizations of the $\forall \Sigma^b_1$-theorems of $T^i_2$ and $U^i_2$, which we gave an account of earlier, led to separation results, it also makes sense to look for other characterizations for these theories. Therefore, another objective of this thesis is to provide such characterizations for the theories $TV^i$ and $U^i_2$. Additionally, we aim to obtain improved new-style witnessing theorems for the theories from $\forall^0(m)$ up to $U^i_2$.

A popular method for studying the theorems of $BA$ theories, of the form $(\forall x)(\exists y)R(x,y)$, is the witnessing theorem method. The basic idea behind the witnessing theorem method is as follows: the existence of a proof of $(\forall x)(\exists y)R(x,y)$ in a theory $\mathcal{T}$ implies the existence of a function $f$ such that

$$(\forall x)R(x,f(x))$$

holds (as a quick remark, condition (1.2) can be rewritten as follows:

$$(\forall x)(G_f(x,y) \supset R(x,y)),$$

where $G_f$ is the graph of $f$). Depending on the complexity of $f$ and what kind of formula $R$ is, we can deduce some conclusion. For the sake of illustration, let $\Phi$ be a class of formulae of the form $\forall x \exists y R(x,y)$ such that every formula in $\Phi$ is provable in $\mathcal{T}$. Suppose that for every formula $\forall x \exists y R(x,y)$ in $\Phi$, there is a function $f$ in some function class $FC$ such that (1.2) holds. Then we can deduce that the $\Phi$-theorems of $\mathcal{T}$ are many-one reducible to $FC$.

There are different types of witnessing theorems in the literature, but we are only going to describe three examples here, which are directly relevant to this thesis: the Herbrand theorem, the Buss-style witnessing theorem and the new-style witnessing theorem.

The Herbrand theorem is a witnessing theorem that only applies to universal theories. It states that if a universal theory $\mathcal{T}$ proves $(\forall x)(\exists y)R(x,y)$, where $R$ is a quantifier-free formula, then there are terms $t_1(x), t_2(x), \ldots, t_k(x)$ in the vocabulary of $\mathcal{T}$ such that

$$\mathcal{T} \vdash \phi(x,t_1(x)) \lor \phi(x,t_2(x)) \lor \ldots \lor \phi(x,t_k(x)).$$

For our purpose, the set of all terms in the vocabulary of $\mathcal{T}$ correspond exactly to some function class.

The Buss-style witnessing theorem is a refined version of the basic idea of a witnessing theorem described above. Basically, it states that if $R \in \Phi$ and $\mathcal{T}$ proves $(\forall x)(\exists y)R(x,y)$, then there exists a function $f$ and a formula $G_f \in \Phi$ such that the following three conditions hold:

(a) $G_f(x,y) \leftrightarrow y = f(x),$

(b) $\mathcal{T} \vdash (\forall x)(\exists y)G_f(x,y),$

(c) $\mathcal{T} \vdash (\forall x)(\forall y)(G_f(x,y) \supset R(x,y)).$

In case $y = f(x)$ is a total search problem, then condition (b) becomes

(b') $\mathcal{T} \vdash (\forall x)(\exists y)G_f(x,y).$

---

9The term “witnessing theorem” was originally coined by Buss [Bus86].
Some recent witnessing theorems have followed an improved paradigm in the sense that condition (c) is now replaced with
\[(c') \; B \vdash (\forall x)(\forall y)(G_f(x,y) \supset R(x,y)),\]
where \(B\) is a theory that is weaker than \(T\) – thus, the correctness of the witnessing function \(f\) is now proved in \(B\) rather than \(T\). These witnessing theorems, with improved paradigms, are called new-style witnessing theorems and were used implicitly in Skelley and Thapen [ST11], and more explicitly in Beckmann and Buss [BB09, BB10], Kołodziejczyk et al.[KNT11] and Thapen [Tha11]. An earlier result by Ferreira [Fer95] can be viewed as a precursor to new-style witnessing theorems. In this thesis, we say that we have an improved new-style witnessing theorem for \(T\) over the base theory \(B\), if \(B\) is weaker than the best known result in the literature.

The study of new-style witnessing theorems for BA theories is important. Unlike the other witnessing theorems described previously, new-style witnessing theorems allow us to ask the following question:

What is the weakest base theory \(B\) where the correctness of the witnessing function is provable?

In this sense, the study of new-style witnessing theorems for BA theories can be seen as a contribution towards Bounded Reverse Mathematics, a program developed by Cook and Nguyen [CN10] and whose goal is to find the weakest theory capable of proving a given theorem that is of interest in Computer Science. From the point of view of Complexity Theory, the idea is to find the smallest complexity class such that the theorem can be proved using concepts in that class.

### 1.2 Overview of the Results of the Thesis

In this thesis, we define a new class of total search problems that we call \(\forall \exists AC^0\), which consists of TFNP search problems \(R(x,y)\) such that \(R(x,y)\) is \(AC^0\)-computable. We believe that \(\forall \exists AC^0\) is the appropriate notion for studying the \(\forall \Sigma^B_1\)-theorems of the theories

\[V^0(m) \subseteq VTC^0 \subseteq \ldots \subseteq VP \subseteq V^1 \subseteq TV^1 \subseteq V^2 \subseteq \ldots \subseteq V^i \subseteq TV^i \subseteq \ldots\]

corresponding to complexity classes below \(PH\). This is because, in this thesis, our characterizations of the \(\forall \Sigma^B_1\)-theorems of these theories are in terms of subclasses of \(\forall \exists AC^0\).

For each theory \(VC\) in

\[V^0(m) \subseteq VTC^0 \subseteq VNC^1 \subseteq VL \subseteq VNL \subseteq VNC \subseteq VP,\]

Nguyen and Cook [NC06] showed that the provably total functions in \(VC\) are precisely those functions in \(FC\). In order to prove this correspondence between \(VC\) and \(FC\), their approach was to construct a universal conservative extension \(\overline{VC}\) of \(VC\), where the terms of \(\overline{VC}\) represent precisely those functions in \(FC\). Via the Herbrand theorem, they then obtained their desired
1. Introduction

correspondence. Thus, the base theory over which the correctness of their witnessing function is provable is VC.

Our results with respect to the theory VC are as follows. In Chapter 3, we define a new class of \( \forall \exists AC^0 \) search problems that we call KPTC. Basically, it is a class of total search problems motivated by the KPT witnessing theorem [KPT91], where finding a solution to an instance \( x \) of a problem \( Q \) in KPTC is carried out cooperatively between a student S and a teacher T: the student provides a potential solution that T either accepts or rejects, and in the case that T rejects, then T must come up with a counterexample that S can then use in order to compute the next candidate solution. Then we show that the class KPTC is \( AC^0 \)-many-one complete for the \( \forall \Sigma^b_1 \)-theorems of VC, where the reduction is provable in \( V^0 \).

Our results concerning VC above extend and improve the results of [NC06, CN10] in the following way. First, our characterization of the \( \forall \Sigma^b_1 \)-theorems of VC is in terms of a class of \( \forall \exists AC^0 \) search problems, rather than a function class. Second, our characterization is obtained via a new-style witnessing theorem – without using Herbrand’s theorem. As a result, we have an improved new-style witnessing theorem for VC.

Our next set of results concern the study of the \( \forall \Sigma^b_1 \)-theorems of the theory \( V^1 \), whose provably total functions are precisely those functions in FP [Bus86].

Johnson, Papadimitriou and Yannakis [JPY88] defined the class PLS, which models the difficulty of finding a local optimal solution to an optimization problem. Informally, in order to find a local optimal solution, a PLS algorithm starts from an initial solution \( i \), which is computed by a polynomial-time function, and then iterates a polynomial-time neighborhood function \( N \) on \( i \) until a local optimal solution is found. Because the search space over which a PLS algorithm operates is exponential in size, in the worst case scenario, it takes exponential-time in order to find a local optimal solution. Now, Cook and Nguyen [CN10] showed that the class PLS remains the same even if the neighborhood function \( N \) and the candidate solution \( i \) were to be computed by \( AC^0 \)-functions.

In Section 4.1, we define a new class of \( \forall \exists AC^0 \) search problems that we call Inflationary Polynomial Local Search (IPLS), which is a derivative of PLS in the sense that we now require the neighborhood function \( N \) to be “inflationary”, in addition to being an \( AC^0 \)-function, whereas we still allow the candidate solution \( i \) to be computed by an \( AC^0 \)-function – with the neighborhood function being inflationary, a local optimal solution is now guaranteed to be found in polynomial-time, despite the search space still being exponential in size. Also, in Section 4.1, we show that IPLS has a complete problem which we call Inflationary Iteration (IITER). Then in Section 4.3, we show that IITER is \( AC^0 \)-many-one complete for the \( \forall \Sigma^b_1 \)-theorems of \( V^1 \), where the reduction is provable in \( V^0 \). Thus, obtaining an improved new-style witnessing theorem for \( V^1 \).

Our results improve and extend Buss’s results [Bus86] for \( V^1 \) in the following way. First, Buss’s witnessing theorem for \( V^1 \) (with respect to FP) is a Buss-style witnessing theorem, whereas ours is a new-style witnessing theorem, where the reduction is provable in \( V^0 \). Second, our characterization is in terms of a class of \( \forall \exists AC^0 \) search problems, instead of a function class.

---

10 A function class can be viewed as a class of total search problems with a single solution.
We would also like to mention that Thapen [Tha11] has a new-style witnessing theorem for $V^1$. However, his new-style witnessing theorem is obtained via a suitable encoding of the computation of a polynomial-time Turing machine. Furthermore, his reduction is provable over Cook’s theory $PV$ [Coo75], a theory that is strictly stronger than $V^0$, but seemingly weaker than $V^1$.

Beckmann and Buss [BB09] introduced the class $\Pi_k^p$-PLS with $\Pi_k^p$-goals, which is a generalization of PLS: the complexity of checking whether a point in the search space is a candidate solution or not is not polynomial-time anymore; instead, it comes from a much higher level $\Pi_k^p$ of PH. Furthermore, it comes with a stopping condition, which is of $\Pi_k^p$-complexity. Then they showed that these generalized PLS problems can be formalized in $V^1$ and that, for $0 \leq g \leq k$, the resulting formalizable $\Pi_k^g$-PLS problems with $\Pi_k^g$-goals are $P$-many-one complete for the $\forall \Sigma_{k+1}^g$-theorems of $TV^{k+1}$, where the reduction is provable in $V^1$. Additionally, they argued that the definitions of $\Pi_k^g$-PLS problems with $\Pi_k^g$-goals can be Skolemized with polynomial-time functions and that the resulting Skolemizable $\Pi_k^g$-PLS problems with $\Pi_k^g$-goals can be used to obtain a stronger characterization of the $\forall \Sigma_{g+1}^B$-consequences of $TV^{k+1}$, where the reduction is provable in $V^1$. Finally, they introduced a $\forall \Sigma_1^B(\alpha)$-principle that they conjectured to separate $TV^k(\alpha)$ and $TV^{k+1}(\alpha)$.

Unfortunately, Beckmann and Buss’s proof of the aforementioned results contain an error. In order to correct their error, in Section 5.1, we redefine $\Pi_k^g$-PLS with $\Pi_k^g$-goals so that the set of candidate solutions are now given by a $\Sigma_{k+1}^g$-predicate, instead of a $\Pi_k^g$-predicate, and we call the resulting class $\Sigma_{k+1}^g$-PLS with $\Pi_k^g$-goals. Furthermore, we show that these generalized PLS problems have complete problems that we call $\Sigma_{k+1}^g$-ITER problems with $\Pi_k^g$-goals. In Section 5.2, we argue that the definitions of these $\Sigma_{k+1}^g$-ITER problems with $\Pi_k^g$-goals are formalizable in the theory $V^0$. In Section 5.4, we show that these formalizable $\Sigma_{k+1}^g$-ITER problems with $\Pi_k^g$-goals are $AC^0$-many-one complete for the $\forall \Sigma_{k+1}^g$-theorems of $TV^{k+1}$, where the reduction is provable in $V^0$. Finally, in Section 5.5, we prove that these $\Sigma_{k+1}^g$-ITER problems with $\Pi_k^g$-goals can be Skolemized with $AC^0$-functions and that the resulting Skolemized problems can be used to obtain a stronger characterization of the $\forall \Sigma_{g+1}^B$-consequences of $TV^{k+1}$, where the reduction is provable in $V^0$. Overall, our results are improvements over Beckmann and Buss’s results (mentioned in the previous paragraph) in several ways.

Kołodziejczyk et al. [KNT11] introduced a class of TFNP search problems called the linear local improvement principle, which is about labelings of a directed acyclic graph $G$ on vertices $\{0, 1, \ldots, a - 1\}$ and whose directed edges are $(x - 1, x)$, for $0 < x \leq a - 1$. In Section 6.4, we extend the definition of the linear local improvement principle by adding more edges to the underlying graph $G$ in the following way: every vertex $x$ in $G$ may now have up to polynomially-many edges $(y, x)$ coming into it, where $y < x$, in addition to $(x - 1, x)$. Then we call the resulting principle extended linear local improvement principle. The extended linear local improvement principle comes in different types: namely, ELLI, ELLI_{log^d} and ELLI_{k}, which differ from each other by the number of rounds allowed in order to improve on the labels. For
1. Introduction

ELLI, the number of rounds of improvements is bounded by $a$, whereas for ELLI$_{\log d}$ and ELLI$_k$, the number of rounds is by bounded by $\log^d a$ and $k$ respectively, where $k$ is a constant. For all $d \geq 1$, we show that ELLI$_{\log d}$ is P-many-one complete for the $\forall \Sigma^b_1$-theorems of $U^1_2$, where the reduction is provable in $S^1_2$. Furthermore, we show that, for all $k \geq m$, where $m$ is some constant which depends on the constant number of queries asked by some polynomial-time Turing machine to some oracle, ELLI$_{\log d}$ and ELLI$_k$ are equivalent and that this equivalence is provable in $S^1_2$. By the results of Kołodziejczyk et al. [KNT11] (who characterized the $\forall \Sigma^b_1$-theorems of $U^1_2$ in terms of the linear local improvement principle LLI$_{\log}$) and Beckmann and Buss [BB14] (who showed that the linear local improvement principle LLI is provable in $U^1_2$), it follows that LLI, LLI$_{\log}$, ELLI$_{k \geq m}$ and ELLI$_{\log \geq 1}$ are all equivalent over $S^1_2$. In Section 6.3, we formulate a new-style witnessing theorem for $U^1_2$ over a base theory WT, which is weaker than $S^1_2$; thus, improving the result of Beckmann and Buss [BB14], who showed it over $S^1_2$.

A third-order theory extending Cook and Nguyen’s second-order viewpoint has been developed in [Ske04, Ske07]. Nevertheless, we have decided to work with $U^1_2$ in this thesis. There are two reasons for this choice. First, the language of $U^1_2$ allows for easier presentation of our results. Second, it would have made sense to consider the third-order viewpoint if we had managed to obtain new-style witnessing theorems for $U^1_2$ over $V^0$. But since we never get down to $V^0$, Buss’s second-order viewpoint (that is to say, $U^1_2$) is again beneficial for us: Buss’s setting allowed us to draw a conclusion on our results regarding new-style witnessing theorems for $U^1_2$, which we do not know if it could have been inferred using the third-order setting. More specifically, Buss’s setting allowed us to conclude that the theory $\Sigma^0_b$-LIND is too weak to use as a base theory for a new-style witnessing theorem for $U^1_2$ that uses standard PSPACE Turing machines.

1.3 Organization of the Thesis

In Chapter 2, we provide some definitions and results from the literature that are needed for this thesis. Chapter 3 discusses our results regarding the class KPTC and the provably total search problems below polynomial-time. In Chapter 4, we introduce the classes IPLS and IITER and demonstrate that they characterize the provably total search problems in $V^1$. Chapter 5 is concerned with our characterizations of the provably total search problems in the theories $TV^i$. Finally, in Chapter 6, we formulate our improved new-style witnessing theorem for $U^1_2$ and introduces the extended linear local improvement principle. Furthermore, we also show how the extended linear local improvement principle is used to obtain characterizations of the provably total search problems in $U^1_2$. 
The prerequisites for reading this thesis are background in first-order logic and an understanding of basic proof theory of first-order logic. Also, an understanding of standard complexity classes ranging from $AC^0$ to PSPACE is required. Nevertheless, we will remind the reader of the definitions of the complexity classes needed in this thesis without going into the details. A good introduction to first-order logic can be found in [End01]. Buss [Bus98] is the best source for the proof theory of first-order logic. Finally, Vollmer [Vol99] and Arora and Barak [AB09] cover all the standard complexity classes of interest in this thesis.

In this chapter, we survey various definitions and results from the literature that are relevant to our work in this thesis. Our exposition of the materials that we present in this chapter follows [CN10].

### 2.1 Two-sorted Logic and Complexity Classes

#### Two-sorted Logic.

Two-sorted (or second-order) logic is an extension of one-sorted (or first-order) logic. Terms and formulae in two-sorted logic are built from the following symbols: an infinite set of variables $x, y, z, \ldots$ (respectively, $X, Y, Z, \ldots$) of the first sort (respectively, second sort) called *number variables* (respectively, *string variables* (or *set variables*)), which are intended to range over $\mathbb{N}$ (respectively, the finite subsets of $\mathbb{N}$); the propositional connectives $\neg, \lor, \land$ and logical constants $\bot$ (for false) and $\top$ (for true); the existential quantifier $\exists$ and universal quantifier $\forall$; parentheses $( $ and $)$; and finally, function and relation (or predicate)
2. Preliminaries

Function and relation symbols can take arguments of both sorts and there are two kinds of function symbols: number function symbols, which range over objects of the first sort, and string function symbols, which range over objects of the second sort.

Let \( n, m \in \mathbb{N} \). Then an \((n,m)\)-ary number function symbol \( f \) is a function symbol that takes \( n \) arguments of the first sort and \( m \) arguments of the second sort. Similarly, one can define the notion of an \((n,m)\)-ary string function symbol and \((n,m)\)-ary relation symbol. A \((0,0)\)-ary function symbol is called a constant symbol.

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 relation symbols.

The vocabulary \( L_2^A \) of two-sorted BA is:

\[
\{0, 1, +, \cdot, |X|, =_1, =_2, \leq, \in\}. \tag{2.1}
\]

Here \( 0, 1, +, \cdot, =_1, \leq \) are the usual symbols of arithmetic on \( \mathbb{N} \); the intended meaning of the function \( |X| \) is 1 plus the largest element in \( X \), or 0 if \( X \) is the empty set; the binary relation \( \in \) takes as arguments a number and a set and is intended to denote set membership. Finally, the binary relation \( =_2 \) is intended to be the equality relation on sets. For notational convenience, we write \( = \) for both \( =_1 \) and \( =_2 \). It will be clear from the context which is intended. Additionally, we use \( X(t) \) as a shorthand for \( t \in X \).

In what follows, when we write \( L \supseteq L_2^A \), we mean that \( L \) is a vocabulary that extends \( L_2^A \); that is to say, \( L \) may contain predicate or function symbols that are not in \( L_2^A \).

**Definition 2.1** Let \( L \supseteq L_2^A \). Then:

1. All number variables are \( L \)-number terms.
2. All string variables are \( L \)-string terms.
3. If \( f \) is an \((n,m)\)-ary number function symbol in \( L \) and \( t_1, \ldots, t_n \) are \( L \)-number terms and \( T_1, \ldots, T_m \) are \( L \)-string terms, then \( f(t_1, \ldots, t_n, T_1, \ldots, T_m) \) is an \( L \)-number term.
4. If \( F \) is an \((n,m)\)-ary string function symbol in \( L \) and \( t_1, \ldots, t_n \) are \( L \)-number terms and \( T_1, \ldots, T_m \) are \( L \)-string terms, then \( F(t_1, \ldots, t_n, T_1, \ldots, T_m) \) is an \( L \)-string term.

In general, the set of all \( L \)-number terms and \( L \)-string terms are called \( L \)-terms.

**Notation 2.2** We usually denote number terms by \( r, s, t, \ldots \) and string terms by \( T_1, \ldots, T_n \).

In two-sorted BA, the most useful form of quantifiers are bounded quantifiers. There are two kinds of bounded quantifiers: bounded number quantifiers and bounded string quantifiers. **Bounded number quantifiers** are quantifiers of the form \((\forall x \leq t)\) and \((\exists x \leq t)\), where \( t \) is any...
term not involving \( x \), and \textit{bounded string quantifiers} are quantifiers of the form \((\forall X \leq t)\) and \((\exists X \leq t)\), where \( t \) is any term not involving \( X \). A formula that only contains bounded quantifiers is called a \textit{bounded formula}. Additionally, a formula that contains no quantifier is called an \textit{open formula} (or a \textit{quantifier-free formula}) and we denote by \( \text{OPEN}(\mathcal{L}) \), the class of open formulae over the vocabulary \( \mathcal{L} \). Finally, a variable that is not within the scope of a quantifier is called a \textit{free variable}.

\textbf{Notation 2.3} We use \( \varphi \supset \psi \) as a shorthand for \( \neg \varphi \lor \psi \) and write \( \varphi \leftrightarrow \psi \) for
\[(\varphi \supset \psi) \land (\psi \supset \varphi).\]
Furthermore, we write \((\exists x \leq t)\varphi(x)\) for \((\exists x)(x \leq t \land \varphi(x))\) and \((\forall x \leq t)\varphi(x)\) for
\[(\forall x)(x \leq t \supset \varphi(x)).\]
For bounded string quantifiers, we write \((\exists X \leq t)\varphi(X)\) for \(\exists X(|X| \leq t \land \varphi(X))\) and \((\forall X \leq t)\varphi(X)\) for \(\forall X(|X| \leq t \supset \varphi(X))\). Additionally, we write \((\exists \bar{x} \leq \bar{t})\varphi\) for \((\exists x_1 \leq t_1)\ldots(\exists x_k \leq t_k)\varphi\), for some constant \( k \), where no \( x_i \) occurs in any \( t_j \), and, similarly for \((\forall \bar{x} \leq \bar{t})\), \((\exists \bar{x} \leq \bar{t})\) and \((\forall \bar{x} \leq \bar{t})\).

Before we can define the class \( \Sigma^B_0 \) of bounded formulae, we first need to define what are the atomic formulae over \( \mathcal{L}^2 \).

\textbf{Definition 2.4} Let \( \mathcal{L} \supseteq \mathcal{L}^2_\Lambda \). Then \( \mathcal{L} \)-atomic formulae are those formulae of the forms
\[t_1 = t_2, T_1 = T_2, t_1 \leq t_2, T(t), R(t_1, \ldots, t_n), T_1, \ldots, T_m),\]
where \( t_1, t_2, \ldots, t_n, T_1, \ldots, T_m, T \) are \( \mathcal{L} \)-terms and \( R \) is an \((n, m)\)-ary relation symbol in \( \mathcal{L} \setminus \mathcal{L}^2_\Lambda \).

Let us now define the class \( \Sigma^B_0 \) of bounded formulae, where only number quantifiers are allowed (but string variables may occur freely):

\textbf{Definition 2.5} Let \( \mathcal{L} \supseteq \mathcal{L}^2_\Lambda \). Then:

- All \( \mathcal{L} \)-atomic formulae are in \( \Sigma^B_0(\mathcal{L}) = \Pi^B_0(\mathcal{L}) \).
- If \( \varphi, \psi \in \Sigma^B_0(\mathcal{L}) \), then so are \( \varphi \land \psi \), \( \varphi \lor \psi \) and \( \neg \psi \).
- If \( \varphi(\bar{x}) \in \Sigma^B_0(\mathcal{L}) \), then so are \( (\exists \bar{x} \leq \bar{t})\varphi(\bar{x}) \) and \( (\forall \bar{x} \leq \bar{t})\varphi(\bar{x}) \).

When \( \mathcal{L} = \mathcal{L}^2_\Lambda \), then we drop mention of \( \mathcal{L} \).

We next define the classes \( \Sigma^B_i \) and \( \Pi^B_i \) of bounded formulae, for \( i \geq 1 \):

\textbf{Definition 2.6} Let \( \mathcal{L} \supseteq \mathcal{L}^2_\Lambda \). Then, for \( i \geq 1 \), \( \Sigma^B_i(\mathcal{L}) \) is the smallest set satisfying the following conditions:

1. \( \Sigma^B_0(\mathcal{L}) = \Pi^B_0(\mathcal{L}) \subseteq \Sigma^B_1(\mathcal{L}) \).
2. If \( \varphi \in \Sigma^B_i(\mathcal{L}) \), then \( (\exists X \leq t)\varphi \in \Sigma^B_i(\mathcal{L}) \) and \( (\forall X \leq t)\varphi \in \Pi^B_{i+1}(\mathcal{L}) \).

The class \( \Pi^B_i(\mathcal{L}) \) is defined dually to \( \Sigma^B_i(\mathcal{L}) \). Finally, the class \( \Sigma^B_1(\mathcal{L}) \) consists of those formulae of the form \( \exists \bar{X} \varphi \), where \( \bar{X} \) is a vector of zero or more string variables and \( \varphi \in \Sigma^B_0(\mathcal{L}) \).

When \( \mathcal{L} = \mathcal{L}^2_\Lambda \), then we drop mention of \( \mathcal{L} \).
Two-sorted sequent calculus LK. In two-sorted sequent calculus, there are two kinds of variables: free variables and bound variables. Free variables can either be number variables \(a, b, c, \ldots\) or string variables \(A, B, C, \ldots, \alpha, \beta, \gamma, \ldots\). Similarly, bound variables can either be number variables \(x, y, z, \ldots\) or string variables \(X, Y, Z, \ldots\).

A sequent \(S\) has the form

\[
\varphi_1, \ldots, \varphi_k \rightarrow \psi_1, \ldots, \psi_l
\]

(2.2)

where \(\varphi_1, \ldots, \varphi_k, \psi_1, \ldots, \psi_l\) are sequences of two-sorted formulae such that each \(\varphi_i\) and \(\psi_j\) have no “free” occurrences of bound variables and no “bound” occurrences of free variables; the symbol \(\rightarrow\) is new and is usually called a sequent arrow (not to be confused with the implication symbol \(\supset\)). The sequence of formulae on the left-hand-side of \(\rightarrow\) is called the antecedent of \(S\) and the one on the right-hand-side of \(\rightarrow\) is called the succedent of \(S\). The intuitive meaning of \(S\) is that the conjunction of the \(\varphi_i\)'s implies the disjunction of the \(\psi_j\)'s. Thus, a sequent \(S\) is equivalent in meaning to the formula

\[
\bigwedge_{i=1}^k \varphi_i \supset \bigvee_{j=1}^l \psi_j.
\]

(2.3)

**Notation 2.7** We write \(S(\vec{a}, \vec{\alpha})\) to denote the sequent \(S\) with free variables \(\vec{a}, \vec{\alpha}\).

**Definition 2.8** Let \(\Phi\) be a set of formulae. Then an LK-\(\Phi\) proof \(\pi\) of a sequent \(S\) is a finite rooted tree whose nodes are sequents and whose root is \(S\). Leaves (or initial sequents) in \(\pi\) are logical axioms, which can be of the form

\[
\varphi \rightarrow \varphi, \quad \perp \rightarrow, \quad \rightarrow \top,
\]

where \(\varphi\) is atomic, or a non-logical axiom of the form

\[
\rightarrow \varphi,
\]

where \(\varphi \in \Phi\), or an instance of one of the following equality axioms (in what follows, let \(\Lambda\) stand for \(t_1 = u_1, \ldots, t_n = u_n, T_1 = U_1, \ldots, T_m = U_m\)):

- **E1** \(\rightarrow t = t;\)
- **E2** \(\rightarrow T = T;\)
- **E3** \(t = u \rightarrow u = t;\)
- **E4** \(T = U \rightarrow U = T;\)
- **E5** \(t = u, u = v \rightarrow t = v;\)
- **E6** \(T = U, U = V \rightarrow T = V;\)
- **E7** \(\Lambda \rightarrow f(t_1, \ldots, t_n, T_1, \ldots, T_m) = f(u_1, \ldots, u_n, U_1, \ldots, U_m);\)
- **E8** \(\Lambda \rightarrow F(t_1, \ldots, t_n, T_1, \ldots, T_m) = F(u_1, \ldots, u_n, U_1, \ldots, U_m);\)
A non-leaf sequent in $\pi$ follows from the sequent(s) immediately above it by one of the following rules of inference:

The weak structural rules:

- weakening-left: $\frac{\Gamma \rightarrow \Delta}{\phi, \Gamma \rightarrow \Delta}$ and weakening-right: $\frac{\Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, \phi}$

- exchange-left: $\frac{\Gamma_1, \phi, \psi, \Gamma_2 \rightarrow \Delta}{\Gamma_1, \psi, \phi, \Gamma_2 \rightarrow \Delta}$ and exchange-right: $\frac{\Gamma \rightarrow \Delta_1, \phi, \psi, \Delta_2}{\Gamma \rightarrow \Delta, \psi, \phi, \Delta_2}$

- contraction-left: $\frac{\phi, \phi, \Gamma \rightarrow \Delta}{\phi, \Gamma \rightarrow \Delta}$ and contraction-right: $\frac{\Gamma \rightarrow \Delta, \phi, \phi}{\Gamma \rightarrow \Delta, \phi}$

The propositional rules:

- $\neg$-left: $\frac{\Gamma \rightarrow \Delta, \phi}{\neg \phi, \Gamma \rightarrow \Delta}$ and $\neg$-right: $\frac{\phi, \Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, \neg \phi}$

- $\land$-left: $\frac{\phi, \psi, \Gamma \rightarrow \Delta}{\phi \land \psi, \Gamma \rightarrow \Delta}$ and $\land$-right: $\frac{\Gamma \rightarrow \Delta, \phi \land \psi}{\Gamma \rightarrow \Delta, \phi}$

- $\lor$-left: $\frac{\phi, \Gamma \rightarrow \Delta}{\phi \lor \psi, \Gamma \rightarrow \Delta}$ and $\lor$-right: $\frac{\Gamma \rightarrow \Delta, \phi \lor \psi}{\Gamma \rightarrow \Delta, \phi}$

The cut rule:

\[
\text{Cut: } \frac{\Gamma \rightarrow \Delta, \phi \phi, \Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta}
\]

The formula $\phi$ is called the cut formula. The structural rules consists of the weak structural rules and the cut rule.

The bounded number quantifier rules:

- $\forall$-left: $\frac{\phi(t), \Gamma \rightarrow \Delta}{t \leq s, (\forall x \leq s) \phi(x), \Gamma \rightarrow \Delta}$ and $\forall$-right: $\frac{b \leq s, \phi(b), \Gamma \rightarrow \Delta}{\Gamma \rightarrow \Delta, (\forall x \leq s) \phi(x)}$

- $\exists$-left: $\frac{b \leq s, \phi(b), \Gamma \rightarrow \Delta}{t \leq s, \phi(t), \Gamma \rightarrow \Delta}$ and $\exists$-right: $\frac{\Gamma \rightarrow \Delta, \phi(t)}{(\exists x \leq s) \phi(x), \Gamma \rightarrow \Delta}$
For the bounded number quantifier rules, the free variable \( b \) is an *eigenvariable* and must not occur in the conclusion of \( \forall \)-right and \( \exists \)-left.

The string quantifier rules:

\[
\begin{align*}
\forall \text{-left: } & \quad \frac{\phi(T), \Gamma \rightarrow \Delta}{\forall X \phi(X), \Gamma \rightarrow \Delta} \quad \text{and} \quad \frac{\Gamma \rightarrow \Delta, \forall X \phi(X)}{\forall X \phi(X), \Gamma \rightarrow \Delta} \\
\exists \text{-left: } & \quad \frac{\phi(\beta), \Gamma \rightarrow \Delta}{\exists X \phi(X), \Gamma \rightarrow \Delta} \quad \text{and} \quad \frac{\Gamma \rightarrow \Delta, \exists X \phi(X)}{\exists X \phi(X), \Gamma \rightarrow \Delta}
\end{align*}
\]

The free variable \( \beta \) is an *eigenvariable* and must not occur in the conclusion of the string quantifier rules \( \forall \)-right and \( \exists \)-left.

Two-sorted complexity classes and representation theorems. Elements of a two-sorted complexity class are relations \( R(\vec{x}, \vec{X}) \) that are taking arguments of both sorts: number arguments \( \vec{x} \) are presented to the accepting machine (Turing machines or circuits) in unary, whereas string arguments \( \vec{X} \) (finite subsets of \( \mathbb{N} \)) are presented in binary to the accepting machine. Here, the main inputs are the string arguments, whereas the number arguments play an auxiliary role and can be used as indices to the string arguments; thus, their values are comparable in size to the length of the string arguments.

**Definition 2.9** The binary representation \( w(S) \) of a finite subset \( S \) of \( \mathbb{N} \) is defined as follows:

\[
w(S) = \begin{cases} \\
\varepsilon & \text{if } S = \emptyset \\
\chi(n) \cdots \chi(1) \chi(0) & \text{otherwise,} 
\end{cases}
\]

where \( \varepsilon \) represents the empty string, \( \chi \) is the characteristic function of \( S \) and \( n \) is the largest element in \( S \). For example, let \( S = \{0, 3, 5\} \). Then

\[
w(S) = 101001.
\]

Furthermore, we define \( (w(S))_2 \) as follows:

\[
(w(S))_2 = \begin{cases} \\
0 & \text{if } S = \emptyset \\
\chi(0) \cdot 2^0 + \chi(1) \cdot 2^1 + \ldots + \chi(n) \cdot 2^n & \text{otherwise,}
\end{cases}
\]

With these conventions in mind, we now define two-sorted uniform circuit complexity classes. The uniformity we use here is FO (First-Order) uniformity [MBIS90, Imm99].

**Definition 2.10** For \( k \geq 0 \), \( AC^k \) (respectively, \( NC^k \)) is the class of relations accepted by uniform families of unbounded (respectively, bounded) fan-in Boolean circuits of size polynomial in \( n \), where \( n \) is the number of inputs, and of depth \( O((\log n)^k) \), and let

\[
NC = \bigcup_{k \geq 0} NC^k.
\]
The class $\text{TC}^0$ (respectively, $\text{AC}^0(m)$) consists of those relations that are accepted by uniform families of polynomial-size and constant-depth unbounded fan-in Boolean circuits with majority gates (respectively, modulo $m$ gates). Here, a majority gate outputs 1 if, and only if, at least half of its inputs are one; and a modulo $m$ gate outputs one if, and only if, the number of one inputs is 1 modulo $m$.

**Definition 2.11** Let $\mathcal{L}$ be a vocabulary that extends $\mathcal{L}_2^A$ and let $\Phi$ be a class of formulae over $\mathcal{L}$. Then a $(k,l)$-ary relation $R$ is represented by a formula $\phi(x_1, \ldots, x_k, X_1, \ldots, X_l)$ in $\Phi$ if for all $n_1, \ldots, n_k \in \mathbb{N}$ and all finite subsets $S_1, \ldots, S_l$ of $\mathbb{N}$, $R(n_1, \ldots, n_k, S_1, \ldots, S_l)$ holds if, and only if, $\phi(n_1, \ldots, n_k, S_1, \ldots, S_l)$ is true the standard model, where all predicate and function symbols of $\mathcal{L}$ have their intended meaning.

For the circuit class $\text{AC}^0$, there is this following correspondence between $\text{AC}^0$ and $\Sigma^B_0$:

**Theorem 2.12** ($\Sigma^B_0$ Representation Theorem [Zam96]) A relation $R(\vec{x}, \vec{X})$ is in $\text{AC}^0$ if, and only if, it is represented by a $\Sigma^B_0$-formula.

We denote by $L$ (respectively, $NL$) the class of relations accepted by a Turing machine (respectively, non-deterministic Turing machine) using a logarithmic amount of memory space.

We next define the Polynomial Hierarchy (PH) and its levels. For the purpose of the definition, let $P$ denote the class of relations accepted by Turing machines in polynomial-time. Furthermore, let $NP$ denote the class of relations accepted by a non-deterministic Turing machines in polynomial-time and let $coNP$ denote the complement of $NP$. Then $\Sigma_0^P = \Pi_0^P = P$ and, for $i \geq 0$,

\[ \Sigma_{i+1}^P = \text{NP}^{\Sigma_i^P}, \]

\[ \Pi_{i+1}^P = \text{coNP}^{\Pi_i^P}, \]

where $\text{NP}^{\Sigma_i^P}$ is the class of relations accepted by a non-deterministic Turing machine with access to an oracle in $\Sigma_i^P$ in polynomial-time, and let

\[ \text{PH} = \bigcup_{i \geq 0} \Sigma_i^P. \]

**Theorem 2.13** ($\Sigma^B_i$ Representation Theorem [Zam96]) For $i \geq 1$, a relation $R(\vec{x}, \vec{X})$ is in $\Sigma_i^P$ if, and only if, is represented by a $\Sigma^B_i$-formula.

We note that Zambella [Zam96] states Theorems 2.12 and 2.13. However, Theorem 2.13 goes back to [Wra78, Fag73, Sto76].

**Two-sorted function classes.** Every two-sorted complexity class $C$ has a corresponding two-sorted function class $FC$. In this paragraph, we define what it means for a two-sorted function to be in $FC$. Here, a two-sorted function is either a number function $f(\vec{x}, \vec{X})$, whose codomain is $\mathbb{N}$, or a string function $F(\vec{x}, \vec{X})$, whose codomain is the set of all finite subsets of $\mathbb{N}$. We are particularly interested in functions for which the length of the output is bounded by a polynomial in the length of the input.
2. Preliminaries

**Notation 2.14** Let $T = T_1, \ldots, T_n$ be a sequence of string terms. Then $|T|$ is a shorthand for $|T_1|, \ldots, |T_n|$.

**Definition 2.15** A number (respectively, string) function $f$ (respectively, $F$) is $p$-bounded if $f(x, X) \leq p(x, |X|)$ (respectively, $|F(x, X)| \leq p(x, |X|)$), for some polynomial $p$.

**Definition 2.16** The graph of a number function $f(x, X)$ (respectively, string function $F(x, X)$) is the relation $G_f(y, x, X)$ (respectively, $G_F(y, x, X)$) such that $G_f(y, x, X)$ holds (respectively, $G_F(y, x, X)$ holds) if, and only if, $y = f(x, X)$ (respectively, $Y = F(x, X)$). The bit-graph of a string function $F(x, X)$ is the relation $B_F(i, x, X)$ such that $B_F(i, x, X)$ holds if, and only if, $F(x, X)(i)$ holds.

For a number function, its graph is used when defining $FC$; whereas for a string function, the right notion to use is its bit-graph. The reason why bit-graphs are used for string functions, instead of graphs, is because there are string functions that are not known to be polynomial-time computable, but their graphs are. For instance, the string function $F(X)$ which computes the prime factorization of $X$ is not known to be polynomial-time computable, but its graph is a polynomial-time relation.

**Definition 2.17** Suppose that $C$ is a two-sorted complexity class. Then a number function (respectively, string) is said to be in $FC$ if it is $p$-bounded and its graph (respectively, bit-graph) is in $C$.

Finally, we denote by $FP^\Sigma^p$ the class of functions that can be computed by a polynomial-time Turing machine with access to an oracle in $\Sigma^p$.

### 2.2 The Theory $V^0$ for $AC^0$

The theory $V^0$. In this paragraph, we define the theory $V^0$. But before we do so, we first need to define certain notions.

The universal closure of a formula $\phi$, denoted $\forall \phi$, is the formula obtained by adding a universal quantifier for every free variable in $\phi$. For a set $\Phi$ of formulae, $\forall \Phi$ denotes the class of all formulae $\forall \phi$, where $\phi \in \Phi$.

For a set $\Phi$ of formulae and a formula $\phi$, we write $\Phi \models \phi$ to mean that $\phi$ is a logical consequence of $\Phi$. Furthermore, when we say that a sequent

$$\phi_1, \ldots, \phi_k \to \psi_1, \ldots, \psi_l$$

is a logical consequence of $\Phi$, we mean

$$\Phi \models \bigwedge_{i=1}^k \phi_i \supset \bigvee_{j=1}^l \psi_j.$$
A *theory* over a vocabulary $\mathcal{L}$ is a set $\mathcal{T}$ of $\mathcal{L}$-formulae that is closed under logical consequence and universal closure. Often we specify $\mathcal{T}$ by a set $\Gamma$ of *axioms* for $\mathcal{T}$, where $\Gamma$ is a set of $\mathcal{L}$-formulae, and in that case,

$$\mathcal{T} = \{ \phi : \phi \text{ is an } \mathcal{L} \text{-formula and } \forall \Gamma \models \phi \}.$$ 

All two-sorted theories that we consider in this thesis extend the base theory $\mathcal{V}^0$ for $\mathcal{AC}^0$, which has as vocabulary $\mathcal{L}_A^2$ and is axiomatized by $2\text{BASIC}$, which is:

A1. $x + 1 \neq 0,$
A2. $x + 1 = y + 1 \supset x = y,$
A3. $x + 0 = x,$
A4. $x + (y + 1) = (x + y) + 1,$
A5. $x \cdot 0 = 0,$
A6. $x \cdot (y + 1) = (x \cdot y) + x,$
A7. $(x \leq y \land y \leq x) \supset x = y,$
A8. $x \leq x + y,$
A9. $0 \leq x,$
A10. $x \leq y \lor y \leq x,$
A11. $x \leq y \leftrightarrow x < y + 1,$
A12. $x \neq 0 \supset (\exists y \leq x)(y + 1 = x)$

L1. $X(y) \supset y < |X|,$
L2. $y + 1 = |X| \supset X(y),$
SE. $(|X| = |Y| \land (\forall i < |X|)(X(i) \leftrightarrow Y(i))) \supset X = Y,$

and the *comprehension axiom scheme* for $\Sigma^B_0$, denoted $\Sigma_0^B$-COMP, which is

$$\exists X \leq y)(\forall z < y)(X(z) \leftrightarrow \phi(z)),$$

where $\phi(z)$ is a $\Sigma^B_0$-formula and $X$ does not occur free in $\phi(z)$.

In general, for a class $\Phi$ of formulae, the comprehension axiom scheme for $\Phi$ is denoted $\Phi$-COMP and has the form of (2.6), but with $\phi \in \Phi$. 

2. Preliminaries

Some basic properties of $V^0$. Even though $V^0$ does not have the axiom of induction explicitly stated in its set of axioms, it is nevertheless able to prove the following schemes, for every formula $\phi \in \Sigma^B_0$, by using $\Sigma^B_0$-COMP and the function $|X|$, which outputs the largest element of $X$:

- The number induction scheme for $\Sigma^B_0$-formulae, denoted $\Sigma^B_0$-IND, which is
  \[
  [\phi(0) \land (\forall x)(\phi(x) \lor \phi(x+1))] \supset (\forall x)\phi(x).
  \] (2.7)

- The number minimization scheme for $\Sigma^B_0$-formulae, denoted $\Sigma^B_0$-MIN, which is
  \[
  \phi(y) \supset (\exists x \leq y)(\phi(x) \land \neg(\exists z < x)\phi(z)).
  \] (2.8)

- The number maximization scheme for $\Sigma^B_0$-formulae, denoted $\Sigma^B_0$-MAX, which is
  \[
  \phi(0) \supset (\exists x \leq y)(\phi(x) \land \neg(\exists z \leq y)(x < z \land \phi(z))).
  \] (2.9)

Similarly to $\Phi$-COMP, one can also talk about the number induction scheme, number minimization scheme and number maximization scheme for $\Phi$, denoted $\Phi$-IND, $\Phi$-MIN and $\Phi$-MAX, respectively, where $\phi$ is now required to be in $\Phi$.

Definability. In this paragraph, we define different notions of definability, one of which is then used to make the correspondence between $V^0$ and FAC$^0$, and later for theories corresponding to other complexity classes.

Notation 2.18 In what follows, we write $(\exists! y)\phi(y)$ for
\[
(\exists y)\phi(y) \land [(\forall y_1)(\forall y_2)[\phi(y_1) \land \phi(y_2) \supset y_1 = y_2]].
\]
The notation $(\exists! Y)\phi(Y)$ is defined similarly to $(\exists! y)\phi(y)$.

Definition 2.19 Let $\mathcal{L} \supseteq \mathcal{L}_A^2$ and $\mathcal{T}$ be a theory over $\mathcal{L}$. Furthermore, let $\Phi$ be a set of formulae over $\mathcal{L}$. Then we say that a string function $F$ is $\Phi$-definable in $\mathcal{T}$ if there is a formula $G_F \in \Phi$ such that
\[
G_F(\vec{x}, \vec{X}, Y) \leftrightarrow Y = F(\vec{x}, \vec{X})
\] (2.10) and
\[
\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists Y G_F(\vec{x}, \vec{X}, Y).
\] (2.11)

In the case of a number function $f$, we say that $f$ is $\Phi$-definable in $\mathcal{T}$ if there is a formula $G_f \in \Phi$ such that
\[
G_f(y, \vec{x}, \vec{X}) \leftrightarrow y = f(\vec{x}, \vec{X})
\] (2.12) and
\[
\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists y G_f(y, \vec{x}, \vec{X}).
\] (2.13)

We say that (2.10) and (2.12) are defining axioms for $F$ and $f$, respectively. Finally, we say that a function (number or string) is provably total in $\mathcal{T}$ if it is $\Sigma^B_1$-definable in $\mathcal{T}$. 

20
2.2. The Theory $V^0$ for $AC^0$

**Notation 2.20** We write $\mathcal{T}(F)$ for the theory obtained by adding $F$ in $L$ and (2.10) in $\mathcal{T}$. Similarly, for $\mathcal{T}(f)$.

For the theory $V^0$, the appropriate notion of definability is $\Sigma^b_1$-definability. This is highlighted by the following theorem, whose proof can be found in [CN10]:

**Theorem 2.21** ([CN10]) A function is in $FAC^0$ if, and only if, it is provably total in $V^0$.

Another notion of definability is bit-definability for string functions:

**Definition 2.22** Let $L \supseteq L^2_A$ and let $\Phi$ be a set of $L$-formulae. Then we say that a string function symbol $F \notin L$ is $\Phi$-bit definable from $L$ if there is a formula $\varphi(i, \vec{x}, \vec{X}) \in \Phi$ and an $L^2_A$-term $t(\vec{x}, \vec{X})$ such that

$$F(\vec{x}, \vec{X})(i) \iff (i < t(\vec{x}, \vec{X}) \land \varphi(i, \vec{x}, \vec{X})).$$

(2.14)

We say that (2.14) is a bit defining axiom (or bit definition) for $F$. Again, if $L = L^2_A$, then we drop mention of $L$.

For string functions in $FAC^0$, there is this additional theorem, whose proof can be found in [CN10]:

**Theorem 2.23** ([CN10]) A string function is $\Sigma^b_0$-bit definable if, and only if, it is in $FAC^0$.

Finally, there is the following “semantic” notion of definability, which is useful for defining the notion of $AC^0$-reduction later.

**Definition 2.24** Let $L$ be a collection of functions and relations. Then we say that a string function (respectively, a number function) is $\Sigma^b_0$-definable from $L$ if it is p-bounded and its bit-graph (respectively, its graph) is represented by a $\Sigma^b_0(L)$-formula.

**Parikh’s Theorem.** In the previous paragraph, we explained what it means for a function (number and string) to be definable in a theory. In this paragraph, we state Parikh’s theorem, which says that a function whose graph is represented by a bounded formula and definable in a BA theory $\mathcal{T}$ is majorized by a term in the vocabulary of $\mathcal{T}$.

**Definition 2.25** Let $\mathcal{T}$ be a theory over a vocabulary $L \supseteq L^2_A$. Then we say that a number function $f$ is p-bounded in $\mathcal{T}$ if there is an $L^2_A$-term $t(\vec{x}, \vec{X})$ such that

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} [f(\vec{x}, \vec{X}) \leq t(\vec{x}, \vec{X})].$$

(2.15)

In the case of a string function, we replace (2.15) by

$$\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} [F(\vec{x}, \vec{X}) \leq t(\vec{x}, \vec{X})].$$

Now, we say that $\mathcal{T}$ is a p-bounded theory if $V^0 \subseteq \mathcal{T}$, each function in $L$ is p-bounded in $\mathcal{T}$ and $\mathcal{T}$ is axiomatized by a set of bounded formulae.

Let us now state Parikh’s theorem:
2. Preliminaries

Theorem 2.26 (Parikh’s Theorem [Par71]) Let \( \mathcal{T} \) be a p-bounded theory and \( \varphi(\bar{x}, \bar{X}, Y) \) be a bounded formula with all free variables displayed and such that
\[
\mathcal{T} \vdash \forall \bar{x} \forall \bar{X} \exists Y \varphi(\bar{x}, \bar{X}, Y).
\]
Then there is an \( \mathcal{L}_A^2 \)-term \( t = t(\bar{x}, \bar{X}) \) such that
\[
\mathcal{T} \vdash \forall \bar{x} \forall \bar{X} (\exists Y \leq t) \varphi(\bar{x}, \bar{X}, Y).
\]

Universal conservative extension \( \mathcal{V}^0 \) of \( \mathcal{V}_0 \). A universal formula is a formula in prenex form where all quantifiers are universal. A universal theory is a theory that is axiomatized by a set of universal formulae.

With the notion of a universal theory in mind, we can now describe the purpose of this paragraph. Basically, we want to define a universal theory called \( \mathcal{V}^0 \) that satisfies certain properties, among which is that the terms in its language represent precisely the functions in \( \text{FAC}^0 \).

Intuitively, the theory \( \mathcal{V}^0 \) is constructed by augmenting \( \mathcal{V}_0 \) with number functions (with universal defining axioms) and string functions (with universal bit defining axioms) that are provably total in \( \mathcal{V}_0 \). Then, using these new functions, the theory \( \mathcal{V}^0 \) proves the axioms of \( \mathcal{V}_0 \) with existential quantifiers: the \( \Sigma^B_0 \)-COMP axioms, \( \text{A12} \) and \( \text{SE} \).

First, we need to replace \( \text{A12} \) with the following pair of universal axioms for the predecessor function \( pd \):

\( \text{A12}' \) \( \text{pd}(0) = 0. \)

\( \text{A12}'' \) \( x \neq 0 \Rightarrow pd(x) + 1 = x. \)

Then we replace \( \text{SE} \) by introducing a new function \( f_{\text{SE}} \), whose universal defining axiom \( \text{SE}’ \) says that \( f_{\text{SE}} \) is the least number \( y < |X| \), which distinguishes \( X \) and \( Y \) if such \( y \) exists, and \( |X| \) otherwise.

We next define the vocabulary of \( \mathcal{V}^0 \):

Definition 2.27 \( \mathcal{L}_{\mathcal{V}^0} \) is the smallest set satisfying the following conditions:

1. \( \mathcal{L}_{\mathcal{V}^0} \supseteq \mathcal{L}_A^2 \cup \{pd, f_{\text{SE}}\}. \)

2. For every open formula \( \varphi(y, \bar{x}, \bar{X}) \) over \( \mathcal{L}_{\mathcal{V}^0} \) and \( \mathcal{L}_A^2 \)-term \( t = t(\bar{x}, \bar{X}) \), there is a number function \( f_{\varphi, t}(\bar{x}, \bar{X}) \) in \( \mathcal{L}_{\mathcal{V}^0} \), whose universal defining axiom is (omitting the parameters \( \bar{x}, \bar{X} \) in \( t \) and \( \varphi \) and writing \( f \) for \( f_{\varphi, t} \))
\[
f \leq t \land (f < t \lor \varphi(f)) \land (y < f \lor \neg \varphi(y)), \tag{2.16}
\]

and there is a string function \( F_{\varphi, t}(\bar{x}, \bar{X}) \) in \( \mathcal{L}_{\mathcal{V}^0} \), whose universal bit defining axiom is
\[
F_{\varphi, t}(\bar{x}, \bar{X})(y) \leftrightarrow (y < t \land \varphi(y, \bar{x}, \bar{X})). \tag{2.17}
\]
Remark 2.28 All functions in $\mathcal{L}_{\mathcal{V}^0}$ are p-bounded.

Definition 2.29 The theory $\mathcal{V}^0$ is over $\mathcal{L}_{\mathcal{V}^0}$ and has the following set of axioms: $\text{A1-A11, A12', A12'', L1 (see 2BASIC), L2 (see 2BASIC), SE'}$ and (2.16), for every function $f_{\varphi,t}$ in $\mathcal{L}_{\mathcal{V}^0}$, and (2.17), for every function $F_{\varphi,t}$ in $\mathcal{L}_{\mathcal{V}^0}$.

We now state certain facts about the theory $\mathcal{V}^0$.

Theorem 2.30 For every $\Sigma^B_0(\mathcal{L}_{\mathcal{V}^0})$-formula $\varphi$, there is an open $\mathcal{L}_{\mathcal{V}^0}$-formula $\varphi'$ such that $\mathcal{V}^0$ proves $\varphi \leftrightarrow \varphi'$.

Theorem 2.31 The terms in $\mathcal{L}_{\mathcal{V}^0}$ represent precisely the functions in $\text{FAC}^0$.

The intuitive reason behind Theorem 2.31 is as follows: Somehow, each number function in $\mathcal{L}_{\mathcal{V}^0}$ has a $\Sigma^B_0$-defining axiom, whereas each string function in $\mathcal{L}_{\mathcal{V}^0}$ has a $\Sigma^B_0$-bit defining axiom. Therefore, by the $\Sigma^B_0$ representation theorem (Theorem 2.12), the fact that number functions in $\mathcal{L}_{\mathcal{V}^0}$ are p-bounded (Remark 2.28) and the definition of $\text{FC}$ (Definition 2.17), it follows that every function in $\mathcal{L}_{\mathcal{V}^0}$ is in $\text{FAC}^0$. The other direction, that every function $\text{FAC}^0$ is in $\mathcal{L}_{\mathcal{V}^0}$ follows from the definition of $\text{FC}$, the $\Sigma^B_0$ representation theorem and Theorem 2.30.

In addition to all these, the theory $\mathcal{V}^0$ also enjoys some comprehension, number induction and number minimization as stated by the following theorem:

Theorem 2.32 The theory $\mathcal{V}^0$ proves $\Sigma^B_0(\mathcal{L}_{\mathcal{V}^0})$-COMP, $\Sigma^B_0(\mathcal{L}_{\mathcal{V}^0})$-IND and $\Sigma^B_0(\mathcal{L}_{\mathcal{V}^0})$-MIN.

Definition 2.33 Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two theories over $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively. Suppose that $\mathcal{T}_1 \subseteq \mathcal{T}_2$ and $\mathcal{L}_1 \subseteq \mathcal{L}_2$. Then we say that $\mathcal{T}_2$ is a conservative extension of $\mathcal{T}_1$ if for every formula $\varphi$ over $\mathcal{L}_1$, if $\mathcal{T}_2$ proves $\varphi$, then $\mathcal{T}_1$ proves $\varphi$.

Theorem 2.34 $\mathcal{V}^0$ is a universal conservative extension of $\mathcal{V}^0$.

Theorem 2.35 (Herbrand Theorem) Let $\mathcal{T}$ be a universal theory and $\varphi(x, \bar{X}, Y)$ be a quantifier-free formula with all free variables displayed such that

$$\mathcal{T} \vdash \forall \bar{x} \forall \bar{X} \exists Y \varphi(\bar{x}, \bar{X}, Y).$$

Then there are terms $T_1(\bar{x}, \bar{X}), \ldots, T_k(\bar{x}, \bar{X})$ in the vocabulary of $\mathcal{T}$ such that

$$\mathcal{T} \vdash \forall \bar{x} \forall \bar{X} [\varphi(\bar{x}, \bar{X}, T_1(\bar{x}, \bar{X})) \lor \ldots \lor \varphi(\bar{x}, \bar{X}, T_k(\bar{x}, \bar{X}))].$$

Finally, we point out that the theory $\mathcal{V}^0$ and the Herbrand theorem can somehow be used in order to obtain a proof of the “if direction” of Theorem 2.21; that is to say, the witnessing theorem for $\mathcal{V}^0$.

Further properties of $\mathcal{V}^0$. An interesting property about the theory $\mathcal{V}^0$ is that if we extend $\mathcal{V}^0$ with a collection of $\text{AC}^0$-functions, then the resulting theory not only remains conservative...
Finally, we define $\langle \rangle$ where $t$

Lemma 2.38 $\Sigma^0_0$-COMPL, where $\mathcal{L}$ is the resulting vocabulary. The following theorem formalizes that:

**Theorem 2.36** Let $\mathcal{T}$ be a p-bounded theory extending $\mathcal{V}$, $\mathcal{L}$. Assume that the vocabulary $\mathcal{L}$ of $\mathcal{T}$ has the same predicate symbols as $\mathcal{L}_A$. Suppose that for every number function $f \in \mathcal{L}$, $\mathcal{T}$ proves a $\Sigma^0_0$-defining axiom for $f$. Furthermore, for every string function $F \in \mathcal{L}$, $\mathcal{T}$ proves a $\Sigma^0_0$-bit defining axiom for $F$. Then for every $i \geq 0$ and every $\Sigma^0_0(\mathcal{L})$-formula $\phi'$, there is a $\Sigma^0_0$-formula $\phi$ such that $\mathcal{T} \vdash \phi' \leftrightarrow \phi$.

Let us next define some useful $\mathcal{AC}^0$-functions that will allow us to pair objects and code a sequence of numbers and strings.

The pairing function that we consider here is the function $\langle x, y \rangle$, which is defined to be twice the Cantor’s pairing function, that is to say,

$$\langle x, y \rangle = (x + 1)(x + y + 1) + 2y.$$  (2.18)

The function $\langle x, y \rangle$ can be recursively extended in order to pair more than two numbers as follows:

$$\langle x_1, \ldots, x_{k+1} \rangle = \langle \langle x_1, \ldots, x_k \rangle, x_{k+1} \rangle.$$  (2.19)

As with numbers, we can also pair strings. Thus, we define the pairing function $\langle X, Y \rangle$ on strings as follows:

$$\langle X, Y \rangle(i) \leftrightarrow (\exists j \leq i)(i = \langle 0, j \rangle \land X(j)) \lor (i = \langle 1, j \rangle \land Y(j)).$$  (2.20)

Also, we can pair more than two strings:

$$\langle X_1, \ldots, X_{i+1} \rangle = \langle \langle X_1, \ldots, X_k \rangle, X_{k+1} \rangle.$$  (2.21)

In order to pair objects of different sorts, we first need to define the $\mathcal{AC}^0$-function $\mathcal{POW}^2(x)$:

$$\mathcal{POW}^2(x)(i) \leftrightarrow i = x.$$  (2.22)

Finally, we define $\langle x_1, \ldots, x_k, X_1, \ldots, X_l \rangle$ as follows:

$$\langle x_1, \ldots, x_k, X_1, \ldots, X_l \rangle = \langle \mathcal{POW}^2(x_1), \ldots, \mathcal{POW}^2(x_k), X_1, \ldots, X_l \rangle.$$  (2.23)

**Notation 2.37** We write $\mathcal{Z}(x_1, \ldots, x_k, X_1, \ldots, X_l)$ for $\mathcal{Z}(\langle x_1, \ldots, x_k, X_1, \ldots, X_l \rangle)$.

Another very useful $\mathcal{AC}^0$-function is the $\mathcal{Row}(x, Z)$ function (also denoted $Z[i]$), which computes the $x$-th row of the binary array $Z$:

$$\mathcal{Row}(x, Z)(i) \leftrightarrow [(i < |Z|) \land Z(x, i)].$$  (2.24)

**Lemma 2.38** $\mathcal{V}^0(\mathcal{Row})$ proves

$$\forall X_1 \forall X_2 \ldots \forall X_k \exists t(X_1 = Z[1] \land \ldots \land X_k = Z[k]),$$  (2.25)

where $t = \langle k, |X_1| + |X_2| + \ldots + |X_k| \rangle$.  

24
We next define a function which codes a sequence \( y_0, y_1, \ldots \) of numbers into a string \( Z \). The idea is that \( y_i \) is the smallest element in \( Z^i \), or \( |Z| \) if \( Z^i \) is empty:

\[
y = \text{seq}(x, Z) \leftrightarrow [y < |Z| \land Z(x, y) \land (\forall z < y)(\neg Z(x, z)) \lor (\forall z < |Z|)(\neg Z(x, z) \land y = |Z|)]. \tag{2.26}
\]

The function \( \text{seq}(x, Z) \) is usually denoted by \( (Z)^x \).

## 2.3 Theories for Complexity Classes Below Polynomial-time

In this section, we define the theories \( \text{VC} \) for small complexity classes below polynomial-time.

Before we define the theory \( \text{VC} \), we first need to define the notion of \( \text{AC}^0 \)-closure of a function. This is because, for every two-sorted complexity class \( \mathcal{C} \) considered here, we fix a function \( F \in \text{FC} \) so that \( \mathcal{C} \) is the \( \text{AC}^0 \)-closure of \( F \). Then we use \( F \) in order to define \( \text{VC} \).

Let us first start with the notion of \( \text{AC}^0 \)-reduction. Intuitively, a function \( G \) is \( \text{AC}^0 \)-reducible to a collection \( \mathcal{L} \) of functions if \( G \) is computed by a uniform family of circuits of polynomial-size and constant-depth and that have unbounded fan-in gates computing functions in \( \mathcal{L} \), in addition to Boolean gates (see [MBIS90]).

**Definition 2.39** Let \( \mathcal{L} \) be a collection of functions. Then a string function \( G \) (respectively, a number function \( g \)) is said to be \( \text{AC}^0 \)-reducible to \( \mathcal{L} \) if there is a sequence of functions \( G_1, \ldots, G_n \) (where \( n \geq 0 \)) such that

\[
G_i \text{ is } \Sigma^B_0 \text{-definable from } \mathcal{L} \cup \{G_1, \ldots, G_{i-1}\}, \text{ for } i = 1, \ldots, n, \tag{2.27}
\]

and \( G \) (respectively, \( g \)) is \( \Sigma^B_0 \)-definable from \( \mathcal{L} \cup \{G_1, \ldots, G_n\} \).

A relation \( R \) is said to be \( \text{AC}^0 \)-reducible to \( \mathcal{L} \) if there is a sequence \( G_1, \ldots, G_n \) satisfying (2.27) and \( R \) is represented by a \( \Sigma^B_0(\mathcal{L} \cup \{G_1, \ldots, G_n\}) \)-formula.

Now that we have defined the notion of \( \text{AC}^0 \)-reducibility, we can define the notion of \( \text{AC}^0 \)- and \( \text{FAC}^0 \)-closure of a function.

**Definition 2.40** Let \( \mathcal{L} \) be a class of functions. Then the \( \text{AC}^0 \)-closure of \( \mathcal{L} \) is the class of relations that are \( \text{AC}^0 \)-reducible to \( \mathcal{L} \), whereas the \( \text{FAC}^0 \)-closure of \( \mathcal{L} \) is the class of functions that are \( \text{AC}^0 \)-reducible to \( \mathcal{L} \).

For all the complexity classes \( \mathcal{C} \subseteq \text{P} \) of interest here, fix a function \( F \in \text{FC} \) so that \( \mathcal{C} \) is the \( \text{AC}^0 \)-closure of \( F \) (we keep \( F \) fixed throughout the rest of this section) – hence, \( \text{FC} \) (see Definition 2.17) is the \( \text{FAC}^0 \)-closure of \( F \) – and so that there is a \( \Sigma^B_0 \)-formula \( \delta_F(X,Y) \) and some \( \mathcal{L}_\delta^2 \)-term \( t(X) \) such that

\[
Y = F(X) \leftrightarrow |Y| \leq t(X) \land \delta_F(X,Y). \tag{2.28}
\]

Additionally, assume that \( \forall^0 \) proves the uniqueness of the value of \( F \), that is to say, \( \forall^0 \) proves

\[
\forall X \forall Y \forall Z (|Y| \leq t(X) \land \delta_F(X,Y) \land |Z| \leq t(X) \land \delta_F(X,Z) \lor Y = Z).
\]
2. Preliminaries

We are almost ready to define the theory VC. But first, we need to explain the notion of an aggregate function of a function. Basically, the aggregate function $G^*(b,Z_1,\ldots,Z_k,X_1,\ldots,X_l)$ of a function $G(x_1,\ldots,x_k,X_1,\ldots,X_l)$ gathers the values of $G$ for a polynomially long sequence of arguments. Thus, $G^*$ is defined so that for all $i < b$,

$$G^*(b,Z_1,X_1,\ldots,X_l) = G((Z_1)^{(i)},\ldots,(Z_k)^{(i)},X_1^{(i)},\ldots,X_l^{(i)}).$$

(2.29)

Finally, we come to the definition of the theory VC:

**Definition 2.41** Let $G_F(b,X,Y)$ be a $\Sigma^0_1$ formula that states that $Y$ is the value for the aggregate function $F^*(b,X)$ of $F(X)$. Then the theory VC is axiomatized by the axioms of $\Sigma^0_1$ and the $\Sigma^0_1$-statement $(\exists Y \leq (b,t))G_F(b,X,Y)$, where $t = t(X)$ is from (2.28).

The following theorem states the correspondence between VC and FC:

**Theorem 2.42** A function is in FC if and only if it is provably total in VC.

A proof of the above theorem can be found in [NC06] or [CN10, Chapter IX]. The idea is to construct a universal conservative extension $\overline{VC}$ (in the style of $\overline{V^0}$) of VC, where the terms in the vocabulary of $\overline{VC}$ represent precisely those functions in FC. Then, via the Herbrand theorem (see Theorem 2.35), one can obtain Theorem 2.42.

2.4 Theories for the Levels of $PH$ and Search Problems

**The theories $\Sigma^0_1$ and $TV^1$.** In this paragraph, we define the theories $\Sigma^0_1$ and $TV^1$, which are the two-sorted versions of Buss’s $S^1_2$ and $T^1_2$. In order to do so, we need to give a definition of the constant $\emptyset$ (which is intended to be the empty set), the binary successor function $S(X)$ and the addition function $X + Y$ on strings. The constant $\emptyset$ is defined as follows:

$$\emptyset(z) \leftrightarrow z < 0.$$ 

As for the binary successor function $S(X)$, it has the following bit defining axiom:

$$S(X)(i) \leftrightarrow (i \leq |X| \land ((X(i) \land (\exists j<i)\neg X(j))) \lor (\neg X(i) \land (\forall j<i)X(j))).$$

(2.30)

Finally, the $\Sigma^0_1$-function $X + Y$ is the addition on strings; that is to say, it is binary addition, when viewing $X$ and $Y$ as binary strings. Therefore, $S(X)$ is basically $X + 1$, where $1$ represents the binary representation of the set $\{0\}$ (see the paragraph on two-sorted complexity class in Section 2.1 for further information on how finite subsets of $\mathbb{N}$ can be viewed as binary strings and vice-versa).

Now, for $i \geq 0$, the theory $\Sigma^0_i$ is axiomatized by the axioms of $\Sigma^0_0$ and the $\Sigma^0_i$-COMP axioms. The theory $TV^i$ is the same as $\Sigma^0_i$, except that instead of the $\Sigma^0_i$-COMP axioms, it is defined in terms of the string induction axiom scheme for $\Sigma^0_i$, denoted $\Sigma^0_i$-SIND, which is

$$[\varphi(\emptyset) \land \forall X(\varphi(X) \supset \varphi(S(X)))] \supset \varphi(Y),$$

26
2.4. Theories for the Levels of \( \text{PH} \) and Search Problems

where \( \varphi \in \Sigma^B_i \) and may have free variables other than \( X \). Then the theory \( V^\infty \) is defined to be

\[
V^\infty = \bigcup_{1 \leq i} V^i. \tag{2.31}
\]

The theory \( TV^i \) proves the following string maximization axiom scheme for \( \Sigma^B_i \), denoted by \( \Sigma^B_i \text{-SMAX} \), which is

\[
\varphi(\emptyset) \supset (\exists X \leq Y)[\varphi(X) \land - (\exists Z \leq Y)(X < Z \land \varphi(Z))], \tag{2.32}
\]

where \( \varphi \in \Sigma^B_i \) and \( X \leq Y \) is the ordering relation on strings with the following defining axiom:

\[
X \leq Y \leftrightarrow X = Y
\]

\[
\lor (|X| \leq |Y| \land (\exists z \leq |Y|)(Y(z) \land -X(z) \land (\forall u \leq |Y|)(z < u \supset (X(u) \supset Y(u))))). \tag{2.33}
\]

We write \( X < Y \) for \( X \leq Y \land - (X = Y) \).

Buss [Bus86] showed the following correspondence between \( V^i \) and \( \text{FP}^{\Sigma^B_{i-1}} \), for \( i \geq 1 \):

**Theorem 2.43** ([Bus86]) For \( i \geq 1 \), \( \text{FP}^{\Sigma^B_{i-1}} \) is precisely the class of functions \( \Sigma^B_i \)-definable in \( V^i \).

In fact, the relationship between \( \text{PH} \) and \( V^\infty \) goes beyond Theorem 2.43, as evidenced by the following theorem:

**Theorem 2.44** ([KPT91, Bus95, Zam96]) For \( i \geq 0 \), if \( TV^i = V^{i+1} \), then \( TV^i = V^\infty \) and \( \Sigma^B_{i+2} = \Pi^B_{i+2} = \text{PH} \) and \( \text{TV}' \) proves \( \Sigma^B_{i+3} = \Pi^B_{i+3} = \text{PH} \).

Theorem 2.44 is an application of the following theorem, which is usually called the KPT witnessing theorem:

**Theorem 2.45** (KPT Witnessing Theorem [KPT91]) Let \( \mathcal{T} \) be a universal theory with vocabulary \( \mathcal{L} \) and let \( \theta(X, Y, Z) \) be a quantifier-free formula over \( \mathcal{L} \) such that \( \mathcal{T} \) proves

\[
\forall X \exists Y \forall Z \theta(X, Y, Z).
\]

Then there exists a finite sequence \( T_1, \ldots, T_k \) of \( \mathcal{L} \)-terms such that \( \mathcal{T} \) proves

\[
\theta(X, T_1(X), Z_1) \lor \theta(X, T_2(X, Z_1), Z_2) \lor \ldots \lor \theta(X, T_k(X, Z_1, \ldots, Z_{k-1}), Z_k),
\]

where the notation \( T_i(X, Z_1, \ldots, Z_{i-1}) \) means that only the displayed variables may occur in \( T_i \).

Finally, for \( i \geq 0 \), we have the following inclusions, originally proven by Buss [Bus86]:

\[
V^i \subseteq TV^i \subseteq V^{i+1}.
\]

As a quick remark, for \( i = 0 \), the theory \( V^0 \) is a strict subset of \( TV^0 \). Also, because of the above inclusions, all \( \Sigma^B_i \)-definable functions in \( V^i \) are also \( \Sigma^B_i \)-definable in \( TV^i \). Hence, functions in \( \text{FP} \) are \( \Sigma^B_1 \)-definable in \( TV^1 \). But unlike \( V^1 \), there is no known characterization of the provably total functions in \( TV^1 \). However, there is a nice characterization of the all of \( \forall \Sigma^B_1 \)-theorems of \( TV^1 \) in terms of search problems.
2. Preliminaries

Search problem. We define a search problem to be a relation \( R(\vec{x}, \vec{X}, Y) \). The search task associated with \( R \) is the following: given an instance \( \vec{x}, \vec{X} \) of \( R \), find a solution \( Y \) such that \( R(\vec{x}, \vec{X}, Y) \) holds, if such \( Y \) exists, and reply “no” otherwise. We also call \( R \) the graph of the search problem.

Notation 2.46 Let \( R(\vec{x}, \vec{X}, Y) \) be a search problem. Then we write \( R(\vec{x}, \vec{X}) \) to denote the search problem \( R \) on instance \( \vec{x}, \vec{X} \). We also say that \( Y \) is a solution to \( R(\vec{x}, \vec{X}) \) to mean that \( R(\vec{x}, \vec{X}, Y) \) holds.

A search problem \( R \) is a total search problem, if, for every instance of \( R \), a solution is guaranteed to exist. The most important class of total search problems is TFNP. Here, a total search problem \( R \) is said to be in TFNP if \( R \) is a polynomial-time relation and polynomially-balanced (that is to say, if \( R(\vec{x}, \vec{X}, Y) \), then \( |Y| \) is bounded by some polynomial \( p(\vec{x}, |\vec{X}|) \)).

In this thesis, we are especially interested in a subclass of TFNP that we call \( \forall\exists\text{AC}^0 \), where we require that \( R \) be given by an \( \text{AC}^0 \) relation instead of a polynomial-time relation. An example of a highly-studied subclass of \( \forall\exists\text{AC}^0 \) is Polynomial Local Search (PLS), which is defined as follows:

Definition 2.47 A PLS problem \( Q(\vec{x}, \vec{X}, Y) \) is specified by the following:

1. An \( \text{AC}^0 \)-relation \( F_Q(\vec{x}, \vec{X}, Y) \) and an \( \mathcal{L}_A^2 \)-term \( t(\vec{x}, \vec{X}) \) such that the following conditions hold:

\[
F_Q(\vec{x}, \vec{X}, \emptyset),
F_Q(\vec{x}, \vec{X}, Z) \supset |Z| \leq t(\vec{x}, \vec{X}).
\]

Here, \( \{Y : F_Q(\vec{x}, \vec{X}, Y)\} \) is the set of candidate solutions for \( Q \) on instance \( \vec{x}, \vec{X} \). Note that \( F_Q \) is a polynomially-balanced relation, since the length \( |Z| \) of every candidate solution \( Z \) is bounded by \( t \).

2. An \( \text{FAC}^0 \)-function \( P_Q(\vec{x}, \vec{X}, Y) \), which computes the profit of \( Y \), and an \( \text{FAC}^0 \)-function \( N_Q(\vec{x}, \vec{X}, Y) \), which computes the neighbor of \( Y \), such that for any \( Y \) satisfying \( F_Q(\vec{x}, \vec{X}, Y) \), the following holds:

\[
[N_Q(\vec{x}, \vec{X}, Y) = Y] \lor [F_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Y)) \land P_Q(\vec{x}, \vec{X}, Y) < P_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Y))].
\]

A solution to \( Q(\vec{x}, \vec{X}) \) is any string \( Y \) such that

\[
F_Q(\vec{x}, \vec{X}, Y) \land N_Q(\vec{x}, \vec{X}, Y) = Y
\]

holds.

We note that our definition of PLS above follows [CN10] and differs from the original definition [JPY88], where \( F_Q, P_Q \) and \( N_Q \) are given by a polynomial-time relation and functions respectively. However, [CN10] showed that the above definition and the original one are equivalent.
2.4. Theories for the Levels of PH and Search Problems

Earlier, we mentioned that there is no known characterization of the provably total functions in TV$^1$. However, there is a nice characterization of the $\forall \Sigma^B_1$-consequences of TV$^1$ in terms of search problems; more precisely, in terms of PLS. In order to make this characterization precise, let us define what it means for a search problem to be definable in a theory and the notion of many-one reduction.

**Definition 2.48** Let $R(\vec{x}, \vec{X}, Y)$ be a total search problem, $\mathcal{T}$ be a theory over a vocabulary $\mathcal{L}$ and $\Phi$ be a set of $\mathcal{L}$-formulae. Then $R$ is said to be $\Phi$-definable in $\mathcal{T}$ if there is a $\Phi$-formula $\varphi_R(\vec{x}, \vec{X}, Y)$ such that the following two conditions hold:

\[
\mathcal{T} \vdash \forall \vec{x} \forall \vec{X} \exists Y \varphi_R(\vec{x}, \vec{X}, Y),
\]  
(2.34)  
\[
\varphi_R(\vec{x}, \vec{X}, Y) \supset R(\vec{x}, \vec{X}, Y)
\]  
(2.35)

The search problem $R$ is said to be provably total in $\mathcal{T}$ if it is $\Sigma^B_1$-definable in $\mathcal{T}$. Furthermore, $R$ is said to be definable in $\mathcal{T}$ if it is $\Psi$-definable in $\mathcal{T}$, for some class $\Psi$ of formulae.

**Definition 2.49** Let $C$ be a two-sorted complexity class and let $R_1$ and $R_2$ be two total search problems. Then $R_1$ is said to be $C$-many-one reducible to $R_2$, denoted $R_1 \leq^C_m R_2$, if there are functions $\vec{f}, \vec{F}, G \in FC$ such that for all $\vec{x}, \vec{X}, Y$, if $R_2(\vec{f}(\vec{x}, \vec{X}), \vec{F}(\vec{x}, \vec{X}), Y)$ holds, then so is $R_1(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}, Y))$.

For two classes $\Gamma$ and $\Delta$ of total search problems, we say that $\Gamma$ is $C$-many-one reducible to $\Delta$, denoted $\Gamma \leq^C_m \Delta$, if for every search problem $R$ in $\Gamma$, there exists a search problem $Q$ in $\Delta$ such that $R \leq^C_m Q$. Furthermore, $\Delta$ is said to be $C$-many-one complete for $\Gamma$ if $\Delta \subseteq \Gamma$ and $\Gamma \leq^C_m \Delta$. Additionally, $\Gamma$ and $\Delta$ are said to be $C$-equivalent if $\Gamma$ and $\Delta$ are $C$-many-one reducible to each other.

Sometimes it is also convenient to talk about a formula being reducible to another formula. To this end, for the remainder of this chapter, we assume that formulae are true in the standard model and that they are always of the form

\[
\forall \vec{x} \forall \vec{X} \exists Y \varphi(\vec{x}, \vec{X}, Y).
\]

Let $\phi \equiv \forall \vec{x} \forall \vec{X} \exists Y \varphi_0(\vec{x}, \vec{X}, Y)$ and $\psi \equiv \forall \vec{x} \forall \vec{X} \exists Y \varphi_0(\vec{x}, \vec{X}, Y)$. Then we say that $\phi$ is $C$-many-one reducible to $\psi$, denoted $\phi \leq^C_m \psi$, if there are functions $\vec{f}, \vec{F}, G \in FC$ such that for all $\vec{x}, \vec{X}, Y$, if $\varphi_0(\vec{f}(\vec{x}, \vec{X}), \vec{F}(\vec{x}, \vec{X}), Y)$ holds, then so is $\varphi_0(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}, Y))$.

For two classes $\Phi$ and $\Psi$ of formulae, we say that $\Phi$ is $C$-many-one reducible to $\Psi$, denoted $\Phi \leq^C_m \Psi$, if for every formula $\phi$ in $\Phi$, there exists a formula $\psi$ in $\Psi$ such that $\phi \leq^C_m \psi$. Additionally, we say that $\Psi$ is $C$-many-one complete for $\Phi$ if $\Psi \subseteq \Phi$ and $\Phi \leq^C_m \Psi$. Finally, $\Phi$ and $\Psi$ are $C$-equivalent if $\Phi$ and $\Psi$ are $C$-many-one reducible to each other.

Buss and Krajíček [BK94] showed that PLS is $\mathcal{P}$-many-one complete for the provably total search problems in TV$^1$, where the reduction is provable over the theory TV$^1$. Then Cook and Nguyen [CN10] strengthened these results by showing that PLS is actually $AC^0$-many-one
2. Preliminaries

complete for the provably total search problems in TV$^1$ and that $V^0$ suffices to prove the reduction; hence obtaining a new-style witnessing theorem for TV$^1$.

In what follows, we say that a function $F(\vec{x}, \vec{X})$ solves a total search problem $Q(\vec{x}, \vec{X}, Y)$ if, and only if, $Q(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}))$ holds, for all $\vec{x}, \vec{X}$.

**Definition 2.50** We define SPC to be the class of total search problems that are solvable by a function in FC.

For example, SPP consists of those total search problems solvable by a function in FP.
In Section 2.3, we gave a formal definition of the theory VC. There, we also saw that the provably total functions in VC are precisely those functions in FC and explained how this correspondence was obtained: via the Herbrand theorem, instead of using a new-style witnessing theorem or a Buss-style witnessing theorem.

In this chapter, we define a new class of $\forall\exists\mathsf{AC}^0$ search problems that we call KPTC, which has been motivated by the KPT witnessing theorem. Intuitively, KPTC is a class of total search problems, where finding a solution to an instance $X$ of a problem $Q$ in the class is carried out cooperatively between a student and a teacher: the student provides a potential solution and the teacher either accepts or rejects; in case the teacher rejects, then he must come up with a counterexample that the student can then use in order to compute the next candidate solution. We show that KPTC is $\mathsf{AC}^0$-many-one complete for the provably total search problems in VC. Our characterization is obtained via a new-style witnessing theorem, which states that the class of provably total search problems in VC is $\mathsf{AC}^0$-many-one reducible to KPTC, where the reduction is provable in $\mathsf{V}^0$.

Our results extend and improve the results of [NC06, CN10] in the following way. Our characterization of the provably total search problems in VC is in terms of a class of $\forall\exists\mathsf{AC}^0$ search problems, instead of a function class. Furthermore, our reduction is provable in $\mathsf{V}^0$, instead of VC.

This chapter is organized as follows. We start by introducing the class KPTC. Then we argue that KPTC is a class of $\forall\exists\mathsf{AC}^0$ search problems, which is provably total in the theory VC. Subsequently, we show the converse: the new-style witnessing theorem for VC. As a result, the class of provably total search problems in VC is characterized by KPTC, where the reduction is provable in $\mathsf{V}^0$. 
3.1 The Theory VC and KPTC

We start this section by reminding the reader of some relevant definitions. Remember that the aggregate function \( G \) of a function \( G(x_1, \ldots, x_k, X_1, \ldots, X_l) \) gathers the values of \( G \) for a polynomially long sequence of arguments. Thus, \( G \) is defined so that for all \( i < b \),

\[
G^b(b, Z, X)[i] = G((Z_1)[i], \ldots, (Z_k)[i], X_1[i], \ldots, X_l[i]). \tag{3.1}
\]

Next, fix a function \( F(X) \) in \( FC \) so that \( C \) is the \( AC^0 \)-closure of \( F \) (we will keep \( F \) fixed throughout the rest of this chapter). Furthermore, let \( \delta_F(X, Y) \) be a \( \Theta^0 \)-formula and \( t = t(X) \) be some \( \mathcal{L}_A^2 \)-term such that

\[
Y = F(X) \leftrightarrow |Y| \leq t \land \delta_F(X, Y). \tag{3.2}
\]

Additionally, assume that \( V^0 \) proves the uniqueness of the value of \( F \), that is to say,

\[
\forall X \forall Y \forall Z |Y| \leq t(X) \land \delta_F(X, Y) \land |Z| \leq t(X) \land \delta_F(X, Z) \land Y = Z.
\]

Finally, let \( G_F(b, X, Y) \) be a \( \Theta^0 \)-formula that states that \( Y \) is the value for the aggregate function \( F^b(b, X) \) of \( F(X) \). Then the theory \( VC \) (Definition 2.41) is axiomatized by the axioms of \( V^0 \) plus the \( \Theta^1 \)-statement \( (\exists Y \leq (b,t))G_F(b, X, Y) \).

**Notation 3.1** We write \( F \) for \( F^b \).

A theorem that is going to be useful in this chapter is the deduction theorem of first-order logic, which is stated as follows:

**Theorem 3.2** (Deduction Theorem of First-order Logic) Let \( \Gamma \) be a class of formulae, let \( \phi \) be a formula with no free variables and \( \psi \) be a formula such that \( \Gamma \cup \{ \phi \} \vdash \psi \). Then \( \Gamma \vdash \phi \supset \psi \).

The following lemma is an application of the KPT witnessing theorem (Theorem 2.45). It says that if the theory \( VC \) proves \( \forall X \exists Y \phi(X,Y) \), where \( \phi \) is a \( \Theta^0 \)-formula, then, for a given \( X \), we can construct a witness for \( \exists Y \phi(X,Y) \) in a collaborative fashion using the aggregate function \( F \) and some \( AC^0 \)-functions \( F_1(X), \ldots, F_k(X, Z_1, \ldots, Z_{k-1}) \), which are suggested by the KPT witnessing theorem.

**Lemma 3.3** Let \( \phi(X, Y) \) be a \( \Theta^0 \)-formula and \( \theta(X, Y, Z) \) denote

\[
G_F([Y[1], Y[2]], Z) \supset \phi(X, Y[0]).
\]

Suppose that the theory \( VC \) proves \( \forall X \exists Y \phi(X,Y) \). Then there exists some \( AC^0 \)-functions \( F_1(X), \ldots, F_k(X, Z_1, \ldots, Z_{k-1}) \) such that \( V^0 \) proves

\[
\bigvee_{i=1}^{k} \theta(X, F_i(X, Z_1, \ldots, Z_{i-1}), Z_i). \tag{3.3}
\]
Student-teacher game interpretation of Lemma 3.3  Before we dive into the proof of the above lemma, let us describe the student-teacher game interpretation of it. For that, let E be the student and U be the teacher. Then, for a given $X$, we can think of the student-teacher game interpretation of Lemma 3.3 as a game about the formula

$$\exists Y \forall Z \theta(X,Y,Z),$$

(3.4)

between E and U, where E’s role is to find a witness $Y$ to the existential quantifier in (3.4), but has computing power limited to $\text{FAC}^0$, whereas U’s role is to find a counterexample $Z$ to the universal quantifier in (3.4), if it exists. More precisely, the game starts with E producing a potential witness $Y_1 = F_1(X)$, which U either approves or rejects – U approves a potential witness $Y$ if $\forall Z \theta(X,Y,Z)$ is true, otherwise U rejects $Y$ and has to provide a counterexample $Z$ such that $\neg \theta(X,Y,Z)$ is true, that is to say, $Z = F(|Y^{(1)}|,Y^{(2)})$ and $\neg \phi(X,Y^{(0)})$ is true. If U rejects $Y_1$, then U provides E with a counterexample $Z_1$, which E uses in order to compute the next potential witness $Y_2 = F_2(X,Z_1)$. Again, either U approves or rejects $Y_2$. As before, if U rejects $Y_2$, then he has to provide E with a counterexample $Z_2$. This process will continue for at most $k$ steps, after which E finds a witness to the existential quantifier in (3.4). However, note here that E cannot compute $F$, since $F$ is beyond E’s computing power.

Proof of Lemma 3.3. The idea is to bring

$$\text{VC} \vdash \forall X \exists Y \phi(X,Y)$$

into a form so that the KPT witnessing theorem is applicable. Since $\text{VC}^0$ is a universal conservative extension of $\text{VC}^0$ (see Theorem 2.34), we have that

$$\text{VC}^0 \cup \{ \forall b \forall X \exists Y \forall F(b,X,Y) \} \vdash \forall X \exists Y \phi(X,Y).$$

By the deduction theorem of first-order logic (see Theorem 3.2), it follows that $\text{VC}^0$ proves

$$\forall B \forall X \exists Y G_F([B],X,Y) \supset \forall X' \exists Y' \phi(X',Y').$$

(3.5)

The next step is to transform (3.5) into a prenex form, which will result in $\text{VC}^0$ proving

$$\forall X' \exists Y' \exists B \exists X \forall Y [G_F([B],X,Y) \supset \phi(X',Y')].$$

Then, by Lemma 2.38, we can combine $Y', B$ and $X$ into one string $(Y', B, X)$. Therefore, we get that $\text{VC}^0$ proves

$$\forall X' \exists (Y',B,X) \forall Y [G_F([B],X,Y) \supset \phi(X',Y')].$$

Now, Theorem 2.30 states that $G_F$, which is a $\Sigma^0_0(\mathcal{L}^\phi)$-formula, is equivalent to some open $\mathcal{L}^\phi$-formula, provable in $\text{VC}^0$ (similarly, for $\phi$). Thus, without loss of generality, we can assume that $G_F$ and $\phi$ are open $\mathcal{L}^\phi$-formulae. Finally, by applying the KPT witnessing theorem and Theorem 2.30, we obtain some $\text{AC}^0$-functions $F_1(X), \ldots, F_k(X,Z_1, \ldots, Z_{k-1})$ such that $\text{VC}^0$ proves (3.3).
In the student-teacher game interpretation of Lemma 3.3, the student is always guaranteed to find a value \( Y \) such that
\[
\forall Z \theta(X, Y, Z)
\]
holds after, at most, \( k \) steps. However, if \( \varphi(X, Y) \) and \( F_1(X), \ldots, F_k(X, Z_1, \ldots, F_k) \) were to be picked arbitrarily, then there is no guarantee that the student would still win, that is to say that he would find a value \( Y \) that satisfies \( \forall Z \theta(X, Y, Z) \). This is because, for an arbitrary \( X \), it is not always the case that there is a \( Y \) such that \( \varphi(X, Y) \) is true. Also, even if \( \forall X \exists Y \varphi(X, Y) \) happened to be true, nothing tells us that
\[
\forall Z \theta(X, F_j(X, Z_1, \ldots, Z_{j-1}), Z)
\]
will hold, for some \( F_j \) in \( F_1, \ldots, F_k \). The class \( \text{KPTC} \) will be defined with the student-teacher game interpretation of Lemma 3.3 in mind, but where \( \varphi \) and \( F_1, \ldots, F_k \) are given arbitrarily. Therefore, some care needs to be taken when defining \( \text{KPTC} \) in order to ensure its totality. More precisely, if in case there is no \( F_j \) in \( F_1, \ldots, F_k \) such that
\[
\forall Z \theta(X, F_j(X, Z_1, \ldots, Z_{j-1}), Z)
\]
holds, then we will just force part of the formula that defines the graph of a \( \text{KPTC} \) search problem to be trivially true.

**Notation 3.4** Let \( i \leq j \). Then we write \( W^{[i, \ldots, j]} \) for \( W^{[i]}, \ldots, W^{[j]} \).

**Definition 3.5** A \( \text{KPTC} \) search problem \( Q(X, W) \) is specified by a \( k \in \mathbb{N} \), a \( \Sigma^B_0 \)-formula \( \varphi(X, Y) \) and \( \text{AC}^0 \)-functions \( F_1(X), \ldots, F_k(X, Z_1, \ldots, Z_{k-1}) \). A string \( W \) is a solution to an instance \( X \) of \( Q \) if, and only if, the following two conditions are satisfied (suppressing the parameter \( X \)):

1. For all \( i \) from 1 to \( k \),
\[
G_F(|F_i(W^{[1, \ldots, j-1]}{]_[1]}|, F_i(W^{[1, \ldots, j-1]}{]_[2]}, W^{[i]}).
\] (3.6)

2. There exists an \( i \) between 1 and \( k \) such that the following holds:
\[
|W^{[0]} = F_i(W^{[1, \ldots, j-1]}{]_[0]}| \land [i < k \ni \varphi(F_i(W^{[1, \ldots, j-1]}{]_[0]})) \land \bigwedge_{j<i} -\varphi(F_j(W^{[1, \ldots, j-1]}{]_[0]}))].
\] (3.7)

We will call \( \varphi \) and \( F_1, \ldots, F_k \) the components of \( Q \).

We will explain (3.6) and (3.7) here. The formula in (3.6) says that \( W^{[i]} \) stores the counterexample

\[
F(|F_i(W^{[1, \ldots, j-1]}{]_[1]}|, F_i(W^{[1, \ldots, j-1]}{]_[2]}).
\]
given by the teacher to the student. In fact, note that, even if \( \varphi(F_i(W^{[1, \ldots, j-1]}{]_[0]})) \) is true, then \( W^{[i]} \) always stores

\[
F(|F_i(W^{[1, \ldots, j-1]}{]_[1]}|, F_i(W^{[1, \ldots, j-1]}{]_[2]}).
\]
Next, the formula in (3.7) guarantees the totality of $Q$. If there is no $F_j$ in $F_1, \ldots, F_k$ such that \( \varphi(F_j(W^{[1,\ldots,j-1]}[0])) \) is true, then the above formula trivially holds by taking $i = k$ and $W^{[0]}$ to be equal to 
\[
F_i(W^{[1,\ldots,j-1]}[0]),
\]
and in case there is an $F_i$ in $F_1, \ldots, F_k$ such that
\[
\varphi(F_i(W^{[1,\ldots,j-1]}[0])
\]
is true, then the formula in (3.7) tells us that $i$ is the least value in $\{1, \ldots, k\}$ such that
\[
\varphi(F_i(W^{[1,\ldots,j-1]}[0])
\]
is true.

Finally, note that $Q(X, W)$, in Definition 3.5 is an $AC^0$-relation, which follows from the $\Sigma^B_0$ representation theorem (see Theorem 2.12). Also, it is worth pointing out that there is an implicit $\mathcal{L}_2$-term $t(|X|)$ that bounds the length of a solution $W$ to a KPTC search problem $Q(X)$. This is because the functions used in order to compute $W^{[i]}$ are within FC – and FC-functions are $p$-bounded (see Definition 2.17).

The following lemma shows that KPTC is provably total in VC:

**Lemma 3.6** Let $Q$ be a KPTC search problem. Then VC proves that $Q$ is total.

**Proof.** We reason in the theory VC. Let $X$ be arbitrary. We will show that there exists a $W$ such that $Q(X, W)$ holds. Let $\varphi(X, Y)$ and $F_1(X), \ldots, F_k(X, Z_1, \ldots, Z_{k-1})$ be the components of $Q$. By the definition of VC, there is a sequence $W_1, \ldots, W_k$ of strings such that
\[
\bigwedge_{i=1}^{k} \varphi(G_X([F_i(X, W_1, \ldots, W_{i-1})[1]], F_i(X, W_1, \ldots, W_{i-1})[2], W_i).
\]
Next, we want to find an $i$ such that $1 \leq i \leq k$ and the following holds:
\[
[i \leq k \supset \varphi(x, F_i(X, W_1, \ldots, W_{i-1})[0])] \land \bigwedge_{j<i} \neg \varphi(x, F_j(X, W_1, \ldots, W_{j-1})[0])
\]
The idea is to start checking if $\varphi(x, F_i(X)^0)$ holds. If it does, then set $i = 1$; otherwise, check if $\varphi(x, F_j(X, W_1)^0)$ holds. This process carries on until either $\varphi(x, F_i(X, W_1, \ldots, W_{i-1})[0])$ holds, for some $i_1 < k$, or $\neg \varphi(x, F_j(X, W_1, \ldots, W_{j-1})[0])$ holds, for every $j$ from 1 to $k-1$. If the former case holds, then set $i = i_1$; otherwise, set $i = k$. Now, define $W_0$ to be equal to $F_i(X, W_1, \ldots, W_{i-1})[0]$ and $W = (W_0, W_1, \ldots, W_k)$. It is easy to verify that $Q(X, W)$ is true. \qed

The next theorem is the new-style witnessing theorem for VC and is a converse of Lemma 3.6. It says that the set of all provably total search problems in VC is $AC^0$-many-one reducible to KPTC. Furthermore, this reduction is provable in $\mathsf{V}^0$. 

35
Theorem 3.7 (New-style Witnessing Theorem for VC) Let $\varphi(X,Y)$ be a $\Sigma^B_1$-formula such that VC proves

$$\forall X \exists Y \varphi(X,Y).$$

Then there is a KPTC search problem $Q$ and an $AC^0$-function $H$ such that $\overline{\nabla^0}$ proves

$$Q(X,W) \supset \varphi(X,H(X,W)).$$

In order to prove the new-style witnessing theorem for VC, we first prove its simple form, that is to say, the case when $\varphi$ is a $\Sigma^B_0$-formula. The proof of Theorem 3.7 will then follow.

Lemma 3.8 Let $\varphi(X,Y)$ be a $\Sigma^B_0$-formula such that VC proves

$$\forall X \exists Y \varphi(X,Y).$$

Then there is a KPTC search problem $Q$ such that $\overline{\nabla^0}$ proves

$$Q(X,W) \supset \varphi(X,W^{[0]}).$$

(3.8)

Proof. By Lemma 3.3, let $F_1(X), \ldots, F_k(X,Z_1,\ldots,Z_k)$ be some $AC^0$-functions such that $\overline{\nabla^0}$ proves

$$\forall X \forall Z_1 \ldots \forall Z_k \left( \bigvee_{i=1}^k \theta(X,F_i(X,Z_1,\ldots,Z_{i-1}),Z_i) \right),$$

(3.9)

where $\theta(X,Y,Z)$ is the formula

$$G_F([Y^{[1]}],Y^{[2]},Z) \supset \varphi(X,Y^{[0]}).$$

Now, we can use $\varphi$ and $F_1,\ldots,F_k$ to define a KPTC search problem $Q$. Arguing in $\overline{\nabla^0}$, we want to show (3.8). So, suppose that $Q(X,W)$ holds. Therefore, we have that (3.7) is true, for some $i$ between 1 and $k$. There are now two cases to consider: $i < k$ and $i = k$. If $i < k$, then $\varphi(X,W^{[0]})$ follows directly. So, assume that $i = k$. In this case, we have that

$$\bigwedge_{j<k} -\varphi(X,F_j(X,W^{[1]\ldots\cdot j^{[1]}})^{[0]})$$

(3.10)

holds. Because $Q(X,W)$ holds by assumption, it follows that (3.6) is true, for all $i$ from 1 to $k$. Since $\overline{\nabla^0}$ proves (3.9), we get that $\varphi(X,F_{i_0}(X,W^{[1\ldots\cdot i_0^{[1]}]})^{[0]})$ holds, for some $i_0$ between 1 and $k$. By (3.10), $i_0 = k$. Since we are in the case that (3.7) is true for $i = k$ (hence, $W^{[0]} = F_k(X,W^{[1\ldots\cdot k^{[1]}]})^{[0]}$), we conclude that $\varphi(X,W^{[0]})$ holds. This finishes the proof.

Let us now sketch the proof of Theorem 3.7:

Proof Sketch of Theorem 3.7. The proof of this theorem is very similar to that of Theorem V.5.1 [CN10, page 119]. For the sake of simplicity, assume that $\varphi(X,Y)$ is of the form

$$\exists Z \psi(X,Y,Z),$$
3.1. The Theory VC and KPTC

where $\psi$ is a $\Sigma^B_0$-formula. Hence, VC proves

$$\forall X \exists Y \exists Z \psi(X, Y, Z). \quad (3.11)$$

Now, the idea is to use the Row function in order to encode the strings $Y, Z$ into a single string $W$ (see Lemma 2.38). Then, applying Lemma 3.8, we obtain a KPTC search problem $Q$ whose solution $W$ on an instance $X$ can be turned into a witness for $\exists Y \varphi(X, Y)$, by using an $AC^0$-function that is based on the function $Row$. Furthermore, this reduction from $\varphi$ to $Q$ is provable in $V^0$.

Finally, combining Lemma 3.6, Theorem 3.7 and the fact that $V^0$ is a universal conservative extension of $V^0$, we obtain the following theorem:

**Theorem 3.9** KPTC is $AC^0$-many-one complete for the provably total search problems in VC. Furthermore, the reduction is provable in the theory $V^0$. 


In the previous chapter, we defined a new class of $\forall\exists AC^0$ search problems that we called KPTC, which is based on the KPT witnessing theorem. We showed that KPTC is $AC^0$-many-one complete for the provably total search problems in $VC$, where the reduction is provable over the base theory $V_0$.

One of the aims of this chapter is to characterize the provably total search problems in $V_1$ in terms of a subclass of $\forall\exists AC^0$. Another aim of this chapter is to provide an improved new-style witnessing theorem for $V_1$. Towards these goals, we introduce two classes of $\forall\exists AC^0$ search problems that are considered here for the first time: Inflationary Polynomial Local Search (IPLS) and Inflationary Iteration (IIITER). Intuitively, the class IPLS is PLS, but with some restriction on the definition of the neighborhood function. The class IIITER is based on the iteration principle [CK98], which can be viewed as the problem of finding vertices with a certain property in an exponentially large directed acyclic graph. Then we show that IIITER is $AC^0$-many-one complete for IPLS. Additionally, we show that IIITER is $AC^0$-many-one complete for SPP (cf. Definition 2.50), the subclass of TFNP solvable in polynomial-time. Our second set of results relates IPLS to the set of provably total search problems in $V_1$. More precisely, we show that IPLS is $AC^0$-many-one complete for the set of provably total search problems in $V_1$, where the reduction is provable over the base theory $V_0$.

Historically, Buss [Bus86] is the first to study the set of provably total search problems in $V_1$, but in the context of one-sorted bounded arithmetic. He showed that the provably total functions in $V_1$ are precisely those functions in FP. Our results in this chapter extend and improve Buss’s results in the following way. First, our characterization of the provably total
search problems in $V^1$ is in terms of a class of $\forall \exists AC^0$ search problems instead of a function class. Second, our reduction is provable in $V^0$, which is strictly weaker than $V^1$.

This chapter is organized as follows. Section 4.1 is mainly about our first set of results. We start that section by defining IPLS and showing that IPLS is a $\forall \exists AC^0$ search problem. Then we introduce the class IITER and argue that it is $AC^0$-many-one complete for IPLS and SPP. Because IITER is $AC^0$-many-one complete for IPLS, our focus will be on IITER in order to prove our second set of results. Section 4.2 serves as a preparation for the proof of the new-style witnessing theorem for $V^1$, which is then formally stated and proven in Section 4.3. In Section 4.3, we also show the converse of the new-style witnessing theorem for $V^1$, which says that IITER is provably total in $V^1$.

We would like to state that the results of this chapter grew out of the discussions we had with Naohi Eguchi on his attempt to capture the complexity class $P$ by building a two-sorted theory using an axiom of inductive definitions [Egu13].

### 4.1 Inflationary Polynomial Local Search and Iteration Problems

In Section 2.1, we explained how finite subsets of $\mathbb{N}$ can be viewed as finite binary strings and how finite binary strings with no leading zeros can be viewed as finite subsets of $\mathbb{N}$. In particular, we described a method of representing a finite subset of $\mathbb{N}$ as a binary string. For the purpose of this section, let us recall how this method works. So, let $S$ be a finite subset of $\mathbb{N}$ and let $\varepsilon$ represent the empty string. Furthermore, let $\chi$ be the characteristic function of $S$. Then the binary representation $w(S)$ of $S$ is defined as follows:

$$w(S) = \begin{cases} 
\varepsilon & \text{if } S = \emptyset \\
\chi(n)\ldots\chi(1)\chi(0) & \text{otherwise},
\end{cases}$$

where $n$ is the largest element in $S$. For example, let $S = \{0, 3, 5\}$. Then

$$w(S) = 101001.$$ 

Therefore, with this method of representing a finite subset of $\mathbb{N}$ as a string in mind, we can view a string function $F(\bar{x}, \bar{X})$ as taking finite binary strings as values, since $F$’s codomain is over the finite subsets of $\mathbb{N}$.

Now that we have reminded the reader of how finite subsets of $\mathbb{N}$ can be viewed as finite binary strings with no leading zeros and vice versa, let us get back to our main concerns in this chapter. The first notion that we define in this chapter is the notion of an “inflationary” string function:

**Definition 4.1** A string function $F(\bar{x}, \bar{X}, Z)$ is said to be inflationary if, and only if, for all $\bar{x}, \bar{X}, Z$, we have that $Z \subseteq F(\bar{x}, \bar{X}, Z)$.

It is worth pointing out that the concept of inflationary function is not new. Already it appeared in the context of Order Theory; more precisely, the Bourbaki-Witt theorem [Bou50]:
4.1. Inflationary Polynomial Local Search and Iteration Problems

a basic fixed-point theorem for partially ordered sets. The Bourbaki-Witt theorem has various important applications. An interesting application of it lies in the area of Descriptive Complexity, where it is shown that the logic obtained by closing first-order logic under the formation of inflationary fixed points has expressive power equivalent, on finite ordered structures, to the complexity class $P$. We will not go into further details here – the interested reader is referred to [Imm99] for more information.

Now that we have defined what it means for a string function to be inflationary, our next task is to use it to define a new class of total search problems based on the complexity class $\text{PLS}$. But first, let us give a brief, intuitive description of $\text{PLS}$. A formal definition can be found in Definition 2.47. A local search algorithm is a method for finding a local maximum of a function such that every point $s$ in the domain of the function is associated with a finite set of points, called the neighbors of $s$, whose members satisfy certain criteria. In order to find a local maximum of a function, a local search algorithm starts at a point in the domain of the function. From there, the algorithm proceeds by finding a neighbor of the point that increases the value of the function. This process is repeated until no such neighbor exists, at which time, the algorithm has identified a local maximum of the function. $\text{PLS}$ formalizes the difficulty of finding such a local maximum of a function as a search problem, when the function is in $\text{FAC}^0$ and a suitable neighboring point can be found in $\text{FAC}^0$. Additionally, if the process of finding a suitable neighboring point can be computed by an inflationary $\text{FAC}^0$-function, then we obtain the class $\text{IPLS}$:

**Definition 4.2** An $\text{IPLS}$ problem $Q(\vec{x}, \vec{X}, Y)$ is specified by the following:

1. An $\text{AC}^0$-relation $F_Q(\vec{x}, \vec{X}, Y)$ and an $L_2^A$-term $t(\vec{x}, \vec{X})$ such that the following conditions hold:

$$
F_Q(\vec{x}, \vec{X}, \emptyset),
F_Q(\vec{x}, \vec{X}, Z) \supset |Z| \leq t(\vec{x}, \vec{X}).
$$

Here, $\{Y : F_Q(\vec{x}, \vec{X}, Y)\}$ is the set of candidate solutions for $Q$ on instance $\vec{x}, \vec{X}$. Note that $F_Q$ is a polynomially-balanced relation, since the length $|Z|$ of every candidate solution $Z$ is bounded by $t$.

2. An $\text{FAC}^0$-function $P_Q(\vec{x}, \vec{X}, Y)$, which computes the profit of $Y$, and an inflationary $\text{FAC}^0$-function $N_Q(\vec{x}, \vec{X}, Y)$, which computes the neighbor of $Y$, such that for any $Y$ that satisfies $F_Q(\vec{x}, \vec{X}, Y)$, the following holds:

$$
[N_Q(\vec{x}, \vec{X}, Y) = Y] \lor [F_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Y)) \land P_Q(\vec{x}, \vec{X}, Y) < P_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Y))].
$$

A solution to an instance $\vec{x}, \vec{X}$ of $Q$ is any string $Y$ such that

$$
F_Q(\vec{x}, \vec{X}, Y) \land N_Q(\vec{x}, \vec{X}, Y) = Y
$$

holds. We will usually refer to $F_Q, P_Q, N_Q$ and $t$ as the components of $Q$. 

41
The following observation states that IPLS is a total search problem. Moreover, it says that checking if a string is a solution to an instance of an IPLS problem is an AC$^0$-property:

**Observation 4.3** Every IPLS problem is a $\forall\exists$AC$^0$ search problem.

**Proof.** Let $Q$ be an IPLS problem. By the definition of a $\forall\exists$AC$^0$ search problem, we need to show that $Q$ is an AC$^0$-relation that is polynomially-balanced and $Q$ is a total search problem.

Let $F_Q$ and $N_Q$ be as in Definition 4.2. Then it is easy to see that $Q$ is an AC$^0$-relation, since $F_Q$ is an AC$^0$-relation and $N_Q$ is an FAC$^0$-function. Also, $Q$ is polynomially-balanced, since $F_Q$ is a polynomially-balanced relation.

Also, it is straightforward to see that $Q$ is a total search problem. For that, assume that $Q$ is not total, for the sake of contradiction. Hence, let $\bar{x},\bar{X}$ be an instance of $Q$ with no solution. First, remember that $\emptyset$ is always a candidate solution and that the set of candidate solutions is bounded. Since the neighbor of a candidate solution is always a candidate solution and the neighborhood function $N_Q$ is inflationary and $Q$ has no solution on $\bar{x},\bar{X}$, this means that repeated application of the neighborhood function on $\emptyset$ will result in an infinite sequence $\emptyset,N_Q(\bar{x},\bar{X},\emptyset),N_Q^2(\bar{x},\bar{X},\emptyset),\ldots$ of candidate solutions such that

$$\emptyset \subset N_Q(\bar{x},\bar{X},\emptyset) \subset N_Q^2(\bar{x},\bar{X},\emptyset) \subset \ldots.$$  

Thus, we have obtained a contradiction to the fact that the set of candidate solutions is bounded.

We will next introduce the class IITER, which is based on the iteration principle [CK98]. Intuitively, the iteration principle states that every directed acyclic graph $\mathcal{G} = (V,E)$ with at least one edge and whose vertices are ordered so that $(U,V) \in E$ implies $U < V$ has a sink. In this intuitive description of the iteration principle, the edges in $\mathcal{G}$ are polynomial-time computable. That is to say, there is an FP-function $F$ such that $(U,V) \in E$ if, and only if, $F(U) = V$. Therefore, if $\mathcal{G}$ were to be a graph of exponential-size, it would take exponentially many steps, in the worst case scenario, to get from a source to a sink using $F$. However, if $F$ were to be inflationary, in addition to being an FP-function, then in order to get to a sink from a source using $F$, it would take, at most, polynomially many steps. This is one motivation for defining the class IITER, which is formally defined as follows, but with an inflationary FAC$^0$-function $F$ instead:

**Definition 4.4** An IITER $Q_F(\bar{x},\bar{X},Y)$ is specified by an inflationary FAC$^0$-function $F(\bar{x},\bar{X},Y)$ and an $\mathcal{L}^2_F$-term $t(\bar{x},\bar{X})$. A solution to an instance $\bar{x},\bar{X}$ of $Q_F$ is a string $Y$ satisfying the formula $\psi_F(\bar{x},\bar{X},Y)$, which is (omitting the parameters $\bar{x},\bar{X}$) given as follows:

$$[Y = \emptyset \land F(Y) = Y] \lor [||Y|| \leq t \land Y < F(Y) \land [t < |F(Y)| \lor F(F(Y)) = F(Y)]]. \quad (4.1)$$

We will usually refer to $F$ and $t$ as the components of $Q_F$. Sometimes, we will make mention of $F$ as the transition function of $Q_F$ and $t$ as the bounding term for $Q_F$. We say that a string $Y$ is a candidate solution to $Q_F$ on instance $\bar{x},\bar{X}$ if $Y$ satisfies the following condition (omitting the parameters $\bar{x},\bar{X}$):

$$|Y| \leq t \land (Y = \emptyset \lor Y < F(Y)). \quad (4.2)$$
As a side remark, the definition of the iteration principle in [CK98] is slightly different from ours. In [CK98], the subformula $F(F(Y)) = F(Y)$ of (4.1) is replaced by $F(F(Y)) \leq F(Y)$. In our case, $F(F(Y)) = F(Y)$ suffices, since we require $F$ to be inflationary.

It has been shown that the iteration principle is $\text{AC}^0$-many-one complete for PLS [Mor01, CN10]. In what will follow, we show that introducing the notion of inflationary function into the definition of PLS, and the iteration principle, does not change anything: $\text{IITER}$ is $\text{AC}^0$-many-one complete for $\text{IPLS}$.

**Lemma 4.5** Every $\text{IITER}$ problem is an $\text{IPLS}$ problem.

*Proof Sketch.* The proof of this lemma is exactly the same as the one for Lemma VIII.5.7 [CN10]. As a result, we will only give a sketched proof here. To start, let $Q_F$ be an $\text{IITER}$ problem and $F,t$ be its components. The goal is to define an $\text{IPLS}$ problem $Q$ such that $Q(\vec{x},\vec{X},Y)$ holds if, and only if, $Q_F(\vec{x},\vec{X},Y)$ holds. First note that the largest $Y$ such that

$$Y \leq t(\vec{x},\vec{X}) \land [Y = \emptyset \lor Y < F(\vec{x},\vec{X},Y)]$$

is always a solution to $Q_F(\vec{x},\vec{X})$. Therefore, we could take as a set of candidate solutions for $Q(\vec{x},\vec{X},Y)$, those $Y$ satisfying (4.3). This implies that the bounding term for the set of candidate solutions for $Q(\vec{x},\vec{X})$ is $t(\vec{x},\vec{X})$. Second, note that the process of finding a solution for an $\text{IITER}$ problem is very much the same as the one for $\text{IPLS}$: In the case of the $\text{IITER}$ problem, one could start from a string $Y$ satisfying (4.3) and iterate $F$ on $Y$ until a solution is found; whereas for an $\text{IPLS}$ problem, one would start from a candidate solution $Y$ and iterate the neighborhood function on $Y$ until a solution is found. Therefore, since $F$ is an inflationary $\text{FAC}^0$-function, the neighborhood function $N_Q$ for $Q$ can easily be defined from $F$. Finally, the profit function $P_Q$ is defined so that $P_Q(\vec{x},\vec{X},Y) = Y$. □

**Lemma 4.6** Every $\text{IPLS}$ problem is $\text{AC}^0$-many-one reducible to an $\text{IITER}$ problem.

*Proof.* Let $Q$ be an $\text{IPLS}$ problem. We want to construct an $\text{IITER}$ problem $Q_F$ and an $\text{FAC}^0$-function $G$ such that for all $\vec{x},\vec{X},Y$, if $Y$ is a solution to $Q_F(\vec{x},\vec{X})$, then $G(\vec{x},\vec{X},Y)$ is a solution to $Q(\vec{x},\vec{X})$.

Let $F_Q,N_Q,P_Q$ and $t$ be the components of $Q$, as in Definition 4.2. We define the function $F$ as follows (omitting the parameters $\vec{x},\vec{X}$):

$$F(Y) = \begin{cases} N_Q(Y) & \text{if } F_Q(Y) \land N_Q(Y) \neq Y \\ Y & \text{otherwise,} \end{cases}$$

(4.4)

Now, let $F$ and $t$ be the components of $Q_F$, as in Definition 4.4, and (omitting the parameters $\vec{x},\vec{X}$) define

$$G(Y) = \begin{cases} N_Q(Y) & \text{if } |Y| \leq t \land Y < F(Y) \land F(F(Y)) = F(Y) \\ Y & \text{otherwise,} \end{cases}$$

Let us now show that if $Y$ is a solution to $Q_F(\vec{x},\vec{X})$, then $G(\vec{x},\vec{X},Y)$ is a solution to $Q(\vec{x},\vec{X})$. So suppose that $Y$ is a solution to $Q_F(\vec{x},\vec{X})$. Hence (omitting $\vec{x},\vec{X}$),

$$[Y = \emptyset \land F(Y) = Y] \lor [|Y| \leq t \land Y < F(Y) \land [t < |F(Y)| \lor F(F(Y)) = F(Y)]$$

43
holds. Now there are two cases to consider. First, assume that
\[ Y = \emptyset \land F(Y) = Y. \]
In this case, note that \( F_0(Y) \) always holds, since \( \emptyset \) is always a candidate solution for any instance of \( Q \). Because \( F(Y) = Y \), by the definition of \( F \), we have \( N_Q(Y) = Y \). Therefore, \( G(Y) \) is a solution to \( Q(\bar{x}, \bar{X}) \), by the definition of \( G \).

Next, assume
\[ |Y| \leq t \land Y < F(Y) \land [t < |F(Y)| \lor F(F(Y)) = F(Y)] \]
and observe that \( t < |F(Y)| \) cannot happen. Now, since \( Y < F(Y) \), by the definition of \( F \),
\[
F(Y) = N_Q(Y), \\
F_0(Y) \land N_Q(Y) \neq Y.
\]
Hence, \( F_0(N_Q(Y)) \) holds. Because \( F(F(Y)) = F(Y) \), we have
\[
F(N_Q(Y)) = N_Q(Y).
\]
Therefore, by the definition of \( F \), we have
\[
N_Q(N_Q(Y)) = N_Q(Y).
\]
By the definition of \( G \), it follows that \( G(Y) \) is a solution to \( Q(\bar{x}, \bar{X}) \). This finishes the proof.

From Lemmas 4.5 and 4.6, we immediately obtain the following corollary:

**Corollary 4.7** \( \text{IITER} \) is \( AC^0 \)-many-one complete for \( IPLS \).

Our next goal is to show that \( \text{IITER} \) is also \( AC^0 \)-many-one complete for \( SPP \), the subclass of \( \text{TFNP} \) solvable in polynomial-time. In order to achieve this goal, we first show how to represent an \( FP \)-function by an \( \text{IITER} \) problem with a single solution. From this, it follows that \( SPP \) is \( AC^0 \)-many-one reducible to \( \text{IITER} \). Then we show how to solve an \( \text{IITER} \) problem by an \( FP \)-function, hence, making \( \text{IITER} \subseteq SPP \). Thus, proving that \( \text{IITER} \) is complete for \( SPP \).

But first, we need to define the following \( AC^0 \)-functions:

**Definition 4.8** ([CN10]) The **string concatenation** function \((X \ast_z Y)\) concatenates the first \( z \) bits of \( X \) with \( Y \) and has the following \( \Sigma^R_0 \)-bit defining axiom:
\[
(X \ast_z Y)(i) \leftrightarrow i < z + |Y| \land [(i < z \land X(i)) \lor (z \leq i \land Y(i - z))],
\]  
(4.5)
where the function \( x \div y \) is the **arithmetical subtraction** and is defined as follows:
\[ z = x \div y \leftrightarrow ((y + z = x) \lor (x \leq y \land z = 0)). \]

The function \((X \ast_z Y)\) can be recursively extended as follows:
\[
(Z_0 \ast_{z_0} Z_1 \ast_{z_1} \ldots \ast_{z_k} Z_{k+1}) = (Z_0 \ast_{z_0} Z_1 \ast_{z_1} \ldots \ast_{z_{k-1}} Z_k) \ast_{z_k} Z_{k+1}
\]
We usually write \( Z_0 \ast_{z_0} Z_1 \ast_{z_1} \ldots \ast_{z_k} Z_{k+1} \) for \( (Z_0 \ast_{z_0} Z_1 \ast_{z_1} \ldots \ast_{z_k} Z_{k+1}) \).
4.1. Inflationary Polynomial Local Search and Iteration Problems

Definition 4.9 Let \( b = b(\bar{x}, \bar{X}) \) be an \( \mathcal{L}_2^t \)-term and \( Y \) be a string of the form
\[
Y_0 \ast_b Y_1 \ast_{2b} \ldots \ast_{lb} Y_l,
\]
where \( l \geq 0 \). Then \( (\bar{x}, \bar{X}, Y) \) is an FAC\(^0\)-function that is defined so that
\[
(\bar{x}, \bar{X}, Y)_j = \begin{cases} 
Y_j & \text{if } j \leq l \\
\emptyset & \text{otherwise},
\end{cases}
\]
We call \( (\bar{x}, \bar{X}, Y)_j \) the \( j \)-th component of \( Y \).

We now show how to represent an FP-function by an IITER problem with a single solution:

Lemma 4.10 Let \( G \) be an FP-function. Then there is an FAC\(^0\)-function \( H \) and an IITER problem \( Q_F \) such that for every \( \bar{x}, \bar{X} \), the problem \( Q_F(\bar{x}, \bar{X}) \) has a unique solution \( Y \) and \( H(\bar{x}, \bar{X}, Y) = G(\bar{x}, \bar{X}) \).

Proof. Let \( M \) be a polynomial-time Turing machine which computes \( G \). We will use the computation of \( M(\bar{x}, \bar{X}) \) in order to set up an IITER problem \( Q_F \) such that \( G(\bar{x}, \bar{X}) \) can be extracted from a solution to \( Q_F(\bar{x}, \bar{X}) \) by using an FAC\(^0\)-function.

Assume an efficient method of encoding \( M \)'s configurations so that, for a configuration \( Z \) of \( M(\bar{x}, \bar{X}) \), there is an \( \mathcal{L}_2^t \)-term \( b = b(\bar{x}, \bar{X}) \) such that \(|Z| \leq b \). Next, let \( \text{Init}_M(\bar{x}, \bar{X}), \text{Next}_M(Z) \) and \( \text{Out}_M(Z) \) be some FAC\(^0\)-functions such that:

- \( \text{Init}_M(\bar{x}, \bar{X}) \) computes the initial configuration of \( M(\bar{x}, \bar{X}) \);
- \( \text{Next}_M(Z) \) computes \( M(\bar{x}, \bar{X}) \)'s next configuration that comes after \( Z \), if \( Z \) encodes a non-final configuration of \( M(\bar{x}, \bar{X}) \), or is the identity if \( Z \) codes the final configuration of \( M(\bar{x}, \bar{X}) \), or equals \( \emptyset \) if \( Z \) does not code a configuration of \( M(\bar{x}, \bar{X}) \);
- \( \text{Out}_M(Z) \) outputs the tape contents of \( M(\bar{x}, \bar{X}) \) from \( Z \), if \( Z \) encodes a configuration of \( M(\bar{x}, \bar{X}) \), or outputs \( \emptyset \) otherwise.

We may assume that \( M \) halts with \( G(\bar{x}, \bar{X}) \) equal to the contents of its tape, hence, \( \text{Out}(Z) = G(\bar{x}, \bar{X}) \). Let \( u = u(\bar{x}, \bar{X}) \) be a bound on the runtime of \( M(\bar{x}, \bar{X}) \). Assume that for all \( \bar{x}, \bar{X}, M(\bar{x}, \bar{X}) \) runs for exactly \( u \) many steps. Now, suppose that the computation of \( M(\bar{x}, \bar{X}) \) is encoded by a sequence of configurations
\[
Z = (Z^0, Z^1, \ldots, Z^u),
\]
where \( Z^0 = \text{Init}_M(\bar{x}, \bar{X}), Z^{i+1} = \text{Next}_M(Z^i) \) and \( Z^u \) is the final configuration of \( M(\bar{x}, \bar{X}) \). Without loss of generality, assume that \( Z^0 \neq \emptyset \) and \( 0 < u \). We are now ready to define the IITER problem \( Q_F \). If \( Y = \emptyset \), then
\[
F(\bar{x}, \bar{X}, Y) = \text{Init}_M(\bar{x}, \bar{X}).
\]
For \( Y = Z_0 \ast_b Z_1 \ast_{2b} \ldots \ast_{ib} Z_i \), where \( i < u \) and \( Z_0 = \text{Init}_M(\bar{x}, \bar{X}) \) and for all \( j < i \), \( Z_{j+1} = \text{Next}_M(Z_j) \), we define
\[
F(\bar{x}, \bar{X}, Y) = Y \ast_{(i+1)b} \text{Next}_M(Z_i).
\]
4. Provably Total Search Problems for Polynomial-time

Otherwise, \( F(\vec{x}, \vec{X}, Y) = Y \). Finally, let \( t(\vec{x}, \vec{X}) = (u+1) \cdot b \) and \( Q_F \) be specified by \( F \) and \( t \).

From the definition of \( F \), one can easily see that \( F \) is inflationary. Hence, we only have to show that \( F \) is an \( \text{FAC}^0 \)-function. For that we need to prove that the conditions that need to be satisfied in every case, in the definition of \( F \), can be expressed by a \( \Sigma^B_0 \)-formula (that is to say, an \( \text{AC}^0 \)-property, by the \( \Sigma^B_0 \) representation theorem). Again, it is easy to see that every condition is expressed by a \( \Sigma^B_0 \)-formula. Since \( \text{FAC}^0 \)-functions are closed under definition by cases, it follows that \( F \) is an \( \text{FAC}^0 \)-function.

Note that \( \emptyset \) cannot be a solution to \( Q_F \). Hence, a solution to \( Q_F \) is a string \( Y \) such that

\[
|Y| \leq t(\vec{x}, \vec{X}) \land Y < F(\vec{x}, \vec{X}, Y) \land F(\vec{x}, \vec{X}, Y) = F(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}, Y)).
\]

Therefore, the only possible solution to \( Q_F \) is (omitting the subscripts to \(*\))

\[
Z_0 * \ldots * Z_{u-1},
\]

where \( Z_0 = \text{Init}_M(\vec{x}, \vec{X}) \) and \( Z_j, Z_{j+1} \) are two consecutive configurations of \( M(\vec{x}, \vec{X}) \). It is now easy to see that from that unique solution to \( Q_F(\vec{x}, \vec{X}) \) one can extract \( G(\vec{x}, \vec{X}) \). Namely, \( \text{Out}_M(\text{Next}_M(Z_{u-1})) = G(\vec{x}, \vec{X}) \).

From Lemma 4.10, we immediately obtain the following corollary:

**Corollary 4.11** Every SPP problem is \( \text{AC}^0 \)-many-one reducible to an \( \text{IITER} \) problem.

We are now left with proving that \( \text{IITER} \subseteq \text{SPP} \); hence, showing \( \text{IITER} \) is \( \text{AC}^0 \)-many-one complete for SPP:

**Lemma 4.12** Let \( Q_F \) be an \( \text{IITER} \) problem specified by \( F, t \). Then \( Q_F \in \text{SPP} \).

**Proof.** To show that \( Q_F \in \text{SPP} \), we need to argue that \( Q_F \in \text{TFNP} \) and there is a polynomial-time algorithm that solves \( Q_F \).

By Lemma 4.5, we have \( \text{IITER} \subseteq \text{IPLS} \). Since \( \text{IPLS} \subseteq \forall \exists \text{AC}^0 \) (see Observation 4.3) and \( \forall \exists \text{AC}^0 \subseteq \text{TFNP} \), it follows that \( \text{IITER} \subseteq \text{TFNP} \). Hence, we are left with proving the existence of a polynomial-time algorithm \( A \) that solves \( Q_F \).

The algorithm \( A(\vec{x}, \vec{X}) \) first checks whether \( \emptyset \) is a solution. If it is, then \( A(\vec{x}, \vec{X}) \) outputs \( \emptyset \). Otherwise, \( A(\vec{x}, \vec{X}) \) will iterate \( F \) on \( \emptyset \) until it finds a solution \( F^k(\emptyset) \) to \( Q_F(\vec{x}, \vec{X}) \). Note that \( A(\vec{x}, \vec{X}) \) will eventually find \( F^k(\emptyset) \) because, if \( F \) does not loop on \( \emptyset \), then the finite but strictly increasing sequence \( \emptyset, F(\emptyset), \ldots, F^k(\emptyset), \ldots \) is guaranteed to be formed, for some \( k \geq 0 \), where \( F^k(\emptyset) \) is the largest element in that sequence such that \( |F^k(\emptyset)| \leq t \) and \( F^k(\emptyset) < F^{k+1}(\emptyset) \).

Next, we argue that, for some polynomial \( p(\vec{x}, |\vec{X}|) \), \( A(\vec{x}, \vec{X}) \) iterates \( F \) on \( \emptyset \) for at most \( p(\vec{x}, |\vec{X}|) \). Actually, we will show that \( k \) is less than or equal to \( t = t(\vec{x}, \vec{X}) \). This will suffice to prove that \( A(\vec{x}, \vec{X}) \) is a polynomial-time algorithm. Let \( \text{numones}(y, Y) \) be the function that computes the total number of elements in \( Y \) that are strictly less than \( y \). Then observe that for all \( i \leq k \), we have that \( \text{numones}(t, F^i(\emptyset)) \leq t \). Now, we prove by induction on \( i \) that for all \( i \leq k \), we have that \( i \leq \text{numones}(t, F^i(\emptyset)) \). For \( i = 0 \), it is straightforward. For the induction step, let \( i \) be an arbitrary number strictly less than \( k \) such that \( i \leq \text{numones}(t, F^i(\emptyset)) \). Then \( \text{numones}(t, F^i(\emptyset)) + 1 \leq \text{numones}(t, F^{i+1}(\emptyset)) \), since \( F^i(\emptyset) \subset F^{i+1}(\emptyset) \). It follows that \( i + 1 \leq \text{numones}(t, F^{i+1}(\emptyset)) \). We conclude that \( k \leq t \). □
From Corollary 4.11 and Lemma 4.12, we obtain the following corollary:

**Corollary 4.13** IITER is $\mathcal{AC}^0$-many-one complete for SPP.

Let us now briefly recapitulate what we have shown so far with regard to IITER and IPLS. Basically, we have shown that IPLS $\subseteq \forall \exists \exists \mathcal{AC}^0$, which is Observation 4.3, and IITER is $\mathcal{AC}^0$-many-one complete for IPLS and SPP, see Corollary 4.7 and 4.13, respectively. As a side remark, Observation 4.3 easily follows from Corollary 4.7 and 4.13 and by observing that a solution to an IPLS problem can be verified in $\mathcal{AC}^0$. However, we have decided to give a separate proof of Observation 4.3, for illustration purposes. Also, note that from Corollary 4.7 and 4.13, the following corollary directly follows:

**Corollary 4.14** IPLS is $\mathcal{AC}^0$-many-one complete for SPP.

### 4.2 Constructions of Inflationary Iteration Problems

In order to present the proof of the new-style witnessing theorem for $V^1$ clearly and swiftly, we show in this section how to combine “smaller” IITER problems to make “bigger” ones. The idea of the constructions we will present in this section was originally introduced in [BB09] in the language of first-order bounded arithmetic $S_2^1$. Before we start the constructions, we first adopt some conventions so as to help with the modular design of IITER problems.

In this section, whenever $Z$ is a candidate solution (see Definition 4.2) to an IITER problem $Q_F(\bar{x}, \bar{X})$, then we always ensure that $Z$ is of the form

$$\langle \bar{x}, \bar{X} \rangle *_{b} Y *_{2b} \delta_1 *_{3b} \delta_2 *_{4b} \ldots *_{(k+1)b} \delta_k$$

for some suitable constant $k$ and $\mathcal{L}^2$-term $b = b(\bar{x}, \bar{X})$ such that $|\langle \bar{x}, \bar{X} \rangle|, |Y|, |\delta_1|, \ldots, |\delta_{k-1}|$ are less than or equal to $b$ and $\delta_k$ is a string small enough so that the length of $Z$ never exceeds the bounding term $t(\bar{x}, \bar{X})$ which bounds the length of all candidate solutions to $Q_F(\bar{x}, \bar{X})$. Furthermore, when $Z$ is a solution to $Q_F(\bar{x}, \bar{X})$, then $Y$ is viewed as the “output” of $Q_F(\bar{x}, \bar{X})$, denoted $Y = Q_F(\bar{x}, \bar{X})$; that is to say,

$$Y = Q_F(\bar{x}, \bar{X}) \iff \exists Z[Q_F(\bar{x}, \bar{X}, Z) \land Y = (\bar{x}, \bar{X}, Z)_1],$$

where $(\bar{x}, \bar{X}, Z)_j$ is the $\mathcal{AC}^0$-function from Definition 4.9 that extracts the $j$-th component of $Z$ (that is to say, $(\bar{x}, \bar{X}, Z)_0 = (\bar{x}, \bar{X}), (\bar{x}, \bar{X}, Z)_1 = Y$, etc).

**Notation 4.15** We write $Y_0 * \ldots * y * \ldots * Y_k$ for

$$Y_0 * \ldots * Y * \ldots * Y_k,$$

where $Y$ is the string representing the unary notation of the number value $y$. Thus, if $Z$ is a solution to an IITER problem $Q_F(\bar{x}, \bar{X})$ such that $Z$ is of the form (omitting the subscripts to $*$)

$$\langle \bar{x}, \bar{X} \rangle * y * \delta_1 * \delta_2 * \ldots * \delta_k,$$

then the output of $Q_F(\bar{x}, \bar{X})$, denoted $y = Q_F(\bar{x}, \bar{X})$, is the string that represents the unary representation of $y$.  

47
4. Provably Total Search Problems for Polynomial-time

FAC⁰-functions as IITER problems. In this paragraph, we will show how to represent FAC⁰-functions by IITER problems. There are two cases that need to be considered: the case of number functions and the one for string functions. Suppose that \( g(\vec{x}, \vec{X}) \) is an FAC⁰-function and let \( b = b(\vec{x}, \vec{X}) \) be an \( \mathcal{L}_{AC}^2 \)-term such that \( g(\vec{x}, \vec{X}) \leq b(\vec{x}, \vec{X}) \) and \( |\langle \vec{x}, \vec{X} \rangle| \leq b(\vec{x}, \vec{X}) \). We want to construct an IITER problem \( Q_{F_g} \) so that \( Z \) is a solution to \( Q_{F_g}(\vec{x}, \vec{X}) \) if, and only if, \( Z \) is of the form

\[
\langle \vec{x}, \vec{X} \rangle \ast b \ast g(\vec{x}, \vec{X}) \ast 1.
\]

We define the inflationary AC⁰-function \( F_{g} \) that makes up \( Q_{F_g} \) as follows (omitting the subscripts to *):

\[
F_{g}(\vec{x}, \vec{X}, Z) = \begin{cases} 
\langle \vec{x}, \vec{X} \rangle \ast g(\vec{x}, \vec{X}) \ast 1 & \text{if } Z = \emptyset \\
Z \ast 1 & \text{if } Z = \langle \vec{x}, \vec{X} \rangle \ast g(\vec{x}, \vec{X}) \ast 1 \\
Z & \text{otherwise}
\end{cases}
\]

Then define \( t = t(\vec{x}, \vec{X}) = 2b + 1 \) and let \( F_{g} \) and \( t \) specify \( Q_{f_g} \).

Note that the only possible solution to \( Q_{F_g}(\vec{x}, \vec{X}) \) is (4.6). The case of a string function \( G(\vec{x}, \vec{X}) \) is defined similarly so that the only possible solution to \( Q_{F_g}(\vec{x}, \vec{X}) \) is

\[
\langle \vec{x}, \vec{X} \rangle \ast b \ast G(\vec{x}, \vec{X}) \ast 2b \ast 1,
\]

where \( |G(\vec{x}, \vec{X})| \leq b \), in addition to \( |\langle \vec{x}, \vec{X} \rangle| \leq b \).

Combining IITER problems. Let \( Q_{F_1}(\vec{x}, \vec{X}, Z) \) and \( Q_{F_1}(\vec{x}, \vec{X}) \) be IITER problems. Then the composition of \( Q_{F_1} \) and \( Q_{F_1} \), denoted

\[
Q_{F_2}(\vec{x}, \vec{X}, Q_{F_1}(\vec{x}, \vec{X}))
\]

is an IITER problem \( Q(\vec{x}, \vec{X}) \), which is defined so that \( Y = Q(\vec{x}, \vec{X}) \) if, and only if, there is a \( Y \) such that \( Y_1 = Q_{F_1}(\vec{x}, \vec{X}) \) and \( Y = Q_{F_1}(\vec{x}, \vec{X}, Y_1) \).

Let \( Q_{F_1}(\vec{x}, \vec{X}) \) and \( Q_{F_1}(\vec{x}, \vec{X}) \) be IITER problems. Then the composition of \( Q_{F_2} \) and \( Q_{F_1} \), denoted

\[
Q_{F_2}(Q_{F_1}(\vec{x}, \vec{X}), \vec{x}, \vec{X}),
\]

is an IITER problem \( Q(\vec{x}, \vec{X}) \), which is defined so that \( Y = Q(\vec{x}, \vec{X}) \) if, and only if, there is a \( Y \) such that \( |Y_1| = Q_{F_1}(\vec{x}, \vec{X}) \) and \( Y = Q_{F_1}(|Y_1|, \vec{x}, \vec{X}) \).

Next, we formally define what it means to “pair” two IITER problems \( Q_{F_1}(\vec{x}, \vec{X}) \) and \( Q_{F_2}(\vec{x}, \vec{X}) \). Assume that \( Q_{F_1} \) and \( Q_{F_2} \) are IITER problems. Then the pairing of \( Q_{F_1} \) and \( Q_{F_2} \), denoted

\[
(Q_{F_1}(\vec{x}, \vec{X}), Q_{F_2}(\vec{x}, \vec{X})),
\]

is an IITER problem \( Q(\vec{x}, \vec{X}) \), which is defined so that \( Y = Q(\vec{x}, \vec{X}) \) if, and only if, there exists a \( Y_1 \) and \( Y_2 \) such that \( Y_1 = Q_{F_1}(\vec{x}, \vec{X}) \) and \( Y_2 = Q_{F_2}(\vec{x}, \vec{X}) \) and \( Y = (Y_1, Y_2) \).
4.2. Constructions of Inflationary Iteration Problems

One can unify the notion of composition and pairing into one single notion, called a $\mathcal{F}\mathcal{G}$-combination. The reason why it is called so is because it is a “combination” of IITER problems, plus the application of some FAC$^0$-functions. Let $Q_{F_1}(\vec{x},\vec{X})$ and $Q_{F_2}(\vec{x},\vec{X})$ be IITER problems. Furthermore, let $\mathcal{G} = \vec{g}(\vec{x},\vec{X},Z), \vec{G}(\vec{x},\vec{X},Z)$ and $\mathcal{F} = F(\vec{x})$ be FAC$^0$-functions. Then the $\mathcal{F}\mathcal{G}$-combination of $Q_{F_1}$ and $Q_{F_2}$, denoted

$$F((Q_{F_1}(\vec{x},\vec{X}),Q_{F_2}(\vec{g}(\vec{x},\vec{X},Q_{F_1}(\vec{x},\vec{X})),\vec{G}(\vec{x},\vec{X},Q_{F_1}(\vec{x},\vec{X})))))$$

is an IITER problem $Q$, which is defined so that $Y = Q(\vec{x},\vec{X})$ if, and only if, there is some $Y_1 = Q_{F_1}(\vec{x},\vec{X})$ and some $Y_2 = Q_{F_2}(\vec{g}(\vec{x},\vec{X},Y_1),\vec{G}(\vec{x},\vec{X},Y_1))$ such that $Y = F(Y_1,Y_2)$.

For the sake of illustration, let us demonstrate how to obtain the composition of $Q_{F_1}(\vec{x},\vec{X})$ and $Q_{F_2}(\vec{x},\vec{X},Z)$ using $\mathcal{F}\mathcal{G}$-combination. For that, let $\mathcal{G} = \vec{g}, \vec{G}$ and $\mathcal{F} = F$ be defined so that $g_i(\vec{x},\vec{X},Z) = x_i$, $G_j(\vec{x},\vec{X},Z) = X_j$, $G(\vec{x},\vec{X},Z) = Z$ and $F(U) = U[1]$. Then it is clear that the $\mathcal{F}\mathcal{G}$-combination of $Q_{F_1}(\vec{x},\vec{X})$ and $Q_{F_2}(\vec{x},\vec{X},Z)$ is $Q_{F_3}(\vec{x},\vec{X},Q_{F_1}(\vec{x},\vec{X}))$.

In fact, an $\mathcal{F}\mathcal{G}$-combination is powerful enough that it can even define IITER problems that are defined by case distinctions. More precisely, let Cond($X,Y,Z$) be an FAC$^0$-function that is defined so that it is equal to $Y$, if $|X| = 0$, and is equal to $Z$, otherwise. Now, let $Q_{F_1}(\vec{x},\vec{X}),Q_{F_2}(\vec{x},\vec{X})$ and $Q_{F_3}(\vec{x},\vec{X})$ be IITER problems. Then

$$\text{Cond}(Q_{F_1}(\vec{x},\vec{X}),Q_{F_2}(\vec{x},\vec{X}),Q_{F_3}(\vec{x},\vec{X}))$$

can be defined in the following way using an $\mathcal{F}\mathcal{G}$-combination. First, we define

$$Q(\vec{x},\vec{X}) = (Q_{F_1}(\vec{x},\vec{X}),Q_{F_2}(\vec{x},\vec{X})).$$

Next, let $\mathcal{G} = \vec{g}, \vec{G}$ such that $g_i(\vec{x},\vec{X},Z) = x_i$ and $G_j(\vec{x},\vec{X},Z) = X_j$, and $\mathcal{F} = F$ such that

$$F(U) = \text{Cond}(U[0],[U[1]]^0,[U[1]]^1).$$

Then it is clear that the $\mathcal{F}\mathcal{G}$-combination

$$F((Q_{F_1}(\vec{x},\vec{X}),Q(\vec{g}(\vec{x},\vec{X},Q_{F_1}(\vec{x},\vec{X})),\vec{G}(\vec{x},\vec{X},Q_{F_1}(\vec{x},\vec{X})))))$$

of $Q_{F_1}$ and $Q$ is $\text{Cond}(Q_{F_1}(\vec{x},\vec{X}),Q_{F_2}(\vec{x},\vec{X}),Q_{F_3}(\vec{x},\vec{X}))$.

We are now going to show how to construct the $\mathcal{F}\mathcal{G}$-combination of two IITER-problems. For that, let $\mathcal{G} = \vec{g}(\vec{x},\vec{X},Z), \vec{G}(\vec{x},\vec{X},Z)$ and $\mathcal{F} = F(\vec{x})$ be FAC$^0$-functions. Furthermore, let $Q_{F_1}(\vec{x},\vec{X})$ and $Q_{F_2}(\vec{x},\vec{X})$ be IITER problems. Before we formally show how to construct the $\mathcal{F}\mathcal{G}$-combination $Q$ of $Q_{F_1}$ and $Q_{F_2}$, let us first explain how $Q$ works. Suppose that we start with an “initial candidate solution” to $Q(\vec{x},\vec{X})$ of the form

$$Z_0 = \langle \vec{x},\vec{X} \rangle *_b Y *_{2b} \gamma_1 *_{3b} \gamma_2 *_{4b} \delta,$$

where $Y$ is intended to be $\emptyset$ at the beginning of the computation and intended to store the output of $Q(\vec{x},\vec{X})$ at the end of the computation; $\gamma_i$ is intended to be a “candidate solution” to $Q_{F_i}$, for $i = 1, 2$; $\delta$ is a sort of a “marker” that is equal to 1 at the beginning of the computation and is updated, when necessary, in order for $Q$ to stay inflationary throughout the computation;
$b = b(\vec{x}, \vec{X})$ is an $L^2$-term large enough so that $|\langle \vec{x}, \vec{X} \rangle|, |Y|, |\gamma_1|, |\gamma_2|$ and $|\delta|$ are all less than or equal to $b$. The search problem $Q$ proceeds by repeatedly using $Q_{F_1}$’s transition function until it eventually finds a solution to $Q_{F_1}(\vec{x}, \vec{X})$. Once a solution $S_1$ to $Q_{F_1}(\vec{x}, \vec{X})$ is found, then $\delta$ is updated and $Q$ repeatedly uses $Q_{F_1}$’s transition function in order to find a solution to $Q_{F_1}(\vec{g}(\vec{x}, \vec{X}, Y_1), \vec{G}(\vec{x}, \vec{X}, Y_1))$, where $Y_1 = (S_1)_1 = (\vec{x}, \vec{X}, S_1)_1 = Q_{F_1}(\vec{x}, \vec{X})$. Eventually, after $u$ many steps, for some $u > 0$, a solution $S_2$ to $Q_{F_2}(\vec{g}(\vec{x}, \vec{X}, Y_1), \vec{G}(\vec{x}, \vec{X}, Y_1))$ will be found. Then the components of $Z_a$ will again be adjusted, so that $Y$ stores the output of $Q(\vec{x}, \vec{X})$ and $Q$ remains inflationary. Finally, $\delta$ is adjusted once more, which marks the end of the computation.

We are now ready to formally define $Q$. Therefore, let $F_i$ and $t_i$ be the components of $Q_{F_i}$ and $\psi_{F_i}$ be its graph, for $i = 1, 2$. First, let us define $Q$’s transition function $F_Q(\vec{x}, \vec{X}, Z)$. In what follows, we omit the parameters $\vec{x}, \vec{X}$ and the subscripts to $\ast$. Assume that $Z$ is of the form $\langle \vec{x}, \vec{X} \rangle \ast Y \ast \gamma_1 \ast \gamma_2 \ast \delta$. If $Z = \emptyset$, then

$$F_Q(Z) = \langle \vec{x}, \vec{X} \rangle \ast \emptyset \ast \emptyset \ast \emptyset \ast 1. \quad (4.7)$$

Now, assume that $Z \neq \emptyset$. If $|\gamma_1| \leq t_1 \land \gamma_1 < F_1(\gamma_1) \land \neg \psi_{F_1}(\gamma_1)$ holds and $\gamma_2 = Y = \emptyset$ and $\delta = 1$, then

$$F_Q(Z) = \langle \vec{x}, \vec{X} \rangle \ast Y \ast F_1(\gamma_1) \ast \gamma_2 \ast \delta. \quad (4.8)$$

If $\psi_{F_1}(\gamma_1)$ and $|\gamma_2| \leq t_2 \land \gamma_2 < F_2(\vec{g}(\gamma_1)_1, \vec{G}(\gamma_1)_1, \gamma_2) \land \psi_{F_2}(\vec{g}(\gamma_1)_1, \vec{G}(\gamma_1)_1, \gamma_2)$ hold and $Y = \emptyset$ and $\delta = 1$, then

$$F_Q(Z) = \langle \vec{x}, \vec{X} \rangle \ast Y \ast \gamma_1 \ast F_2(\vec{g}(\gamma_1)_1, \vec{G}(\gamma_1)_1) \ast \delta. \quad (4.9)$$

If $\psi_{F_1}(\gamma_1)$ and $\psi_{F_2}(\vec{g}(\gamma_1)_1, \vec{G}(\gamma_1)_1, \gamma_2)$ hold and $Y = \emptyset$ and $\delta = 1$, then

$$F_Q(Z) = \langle \vec{x}, \vec{X} \rangle \ast \gamma_1 \ast F(\gamma_1)_1 \ast \gamma_2 \ast 3. \quad (4.10)$$

If $\psi_{F_1}(\gamma_1)$ and $\psi_{F_2}(\vec{g}(\gamma_1)_1, \vec{G}(\gamma_1)_1, \gamma_2)$ hold and $Y = F(\gamma_1)_1, (\gamma_2)_1)$ and $\delta = 3$, then

$$F_Q(Z) = \langle \vec{x}, \vec{X} \rangle \ast Y \ast \gamma_1 \ast \gamma_2 \ast 7. \quad (4.11)$$

In all other cases, we have that $F_Q(Z) = Z$. The bounding term $t_Q(\vec{x}, \vec{X})$ is defined to be equal to $4 \cdot b(\vec{x}, \vec{X}) + 3$. Let $Q$ be specified by $F_Q(\vec{x}, \vec{X}, Z)$ and $t_Q(\vec{x}, \vec{X})$.

The following lemma states that if $Q$ is an $\mathcal{F}G$-combination of two IITER problems $Q_{F_1}$ and $Q_{F_2}$, then a solution to $Q_{F_1}$ and $Q_{F_2}$ can be extracted from a solution to $Q$ using some $\text{FAC}^0$-functions, provable in the theory $\overline{\text{V}}^0$:

**Lemma 4.16** Let $G = \vec{g}(\vec{x}, \vec{X}, Z), \vec{G}(\vec{x}, \vec{X}, Z)$ and $F = F(X)$ be some $\text{FAC}^0$-functions. Furthermore, for $i = 1, 2$, let $Q_{F_i}(\vec{x}, \vec{X})$ be some IITER problem, with graph $\psi_{F_i}$, and let $Q$ be the $\mathcal{F}G$-combination of $Q_{F_1}$ and $Q_{F_2}$, with graph $\psi_{Q}$. Then there are $\text{FAC}^0$-functions $\Pi_1(\vec{x}, \vec{X}, Z)$ and $\Pi_2(\vec{x}, \vec{X}, Z)$ such that $\overline{\text{V}}^0$ proves

$$\psi_{Q}(\vec{x}, \vec{X}, Z) \supseteq \psi_{F_1}(\vec{x}, \vec{X}, \Pi_1(\vec{x}, \vec{X}, Z)) \land \psi_{F_2}(\vec{g}(\vec{x}, \vec{X}, (\Pi_1(\vec{x}, \vec{X}, Z))_1, \vec{G}(\vec{x}, \vec{X}, (\Pi_1(\vec{x}, \vec{X}, Z))_1), \Pi_2(\vec{x}, \vec{X}, Z)). \quad (4.12)$$
Proof. We reason in $\forall^0$. Let $F_Q$ and $t_Q$ be the components of $Q$. Remember that a string $Z$ is a solution to $Q(\vec{x}, \vec{X})$ if either $Z = \emptyset$ and $F_Q(Z) = Z$, or

$$|Z| \leq t_Q \land Z < F_Q(Z) \land [|F_Q(Z)|] > t_Q \lor F_Q(F_Q(Z)) = F_Q(Z).$$

Suppose that $Z$ is a solution to $Q(\vec{x}, \vec{X})$. Observe that $Z$ cannot be $\emptyset$. To see this, note that if $Z = \emptyset$, then $F_Q(Z)$ is of the same form as (4.7). Therefore,

$$Z < F_Q(Z) \land |F_Q(Z)| \leq t_Q.$$

Now, one of (4.8), (4.9) and (4.10) must apply to $F_Q(Z)$. This implies

$$F_Q(Z) < F_Q(F_Q(Z)).$$

Hence, we know that $Z \neq \emptyset$. Next, observe that the conditions that imply (4.8), (4.9) and (4.10) cannot hold for $Z$. Since $Z < F_Q(Z)$, the only possibility is for $Z$ to satisfy the conditions that imply (4.11). This means that $Z$ has to be of the form

$$\langle \vec{x}, \vec{X} \rangle \ast F(\langle \gamma_1, \gamma_2 \rangle) \ast \gamma_1 \ast \gamma_2 \ast 3,$$

where $\gamma_1$ is a solution to $Q_{R_1}(\vec{x}, \vec{X})$ and $\gamma_2$ is a solution to $Q_{R_2}(\vec{g}(\vec{x}, \vec{X}, (\gamma_1)_1), \vec{g}(\vec{x}, \vec{X}, (\gamma_1)_1))$. □

4.3 Inflationary Polynomial Local Search and $V^1$

In Section 4.1 we showed that the class IITER is $AC^0$-many-one complete for IPLS, which in turn is $AC^0$-many-one complete for SPP (see Definition 2.50). In this section, we characterize the provably total search problems in $V^1$ in terms of IITER, where the reduction is provable in $V^0$. This section goes as follows. First, we show that IITER is provably total in $V^1$. Then we prove the other direction, that is to say, the new-style witnessing theorem for $V^1$, which relies on a new-style witnessing lemma.

Theorem 4.17 Let $Q$ be an IITER problem. Then $Q$ is provably total in $V^1$.

Proof. By Lemma 4.12, let $F$ be an FP-function such that $F$ solves $Q$. By Theorem 2.43, the function $F$ is provably total in $V^1$. Therefore, let $\varphi(\vec{x}, \vec{X}, Y)$ be a $\Sigma^0_1$-formula such that

$$V^1 \vdash \forall \vec{x} \forall \vec{X} \exists! Y \varphi(\vec{x}, \vec{X}, Y), \quad (4.13)$$

$$\varphi(\vec{x}, \vec{X}, Y) \leftrightarrow Y = F(\vec{x}, \vec{X}, Y). \quad (4.14)$$

Since $F$ solves $Q$, the following formula holds:

$$Y = F(\vec{x}, \vec{X}) \supset Q(\vec{x}, \vec{X}, Y). \quad (4.15)$$

From (4.13), (4.14), and (4.15), it easily follows that $Q$ is provably total in $V^1$. □

We next prove the converse of Theorem 4.17, that is to say, the new-style witnessing theorem for $V^1$. Basically, it states that the class of provably total search problems in $V^1$ is $AC^0$-many-one reducible to IITER, where the reduction is provable in the theory $\forall^0$. This extends and improves Buss’s results [Bus86], who showed that the class of provably total search problems in $V^1$ is $AC^0$-many-one reducible to FP, where the reduction is provable in $V^1$.
4. Provably Total Search Problems for Polynomial-time

**Theorem 4.18** (New-style Witnessing Theorem for $V^1$) Suppose that $\varphi(\vec{x}, \vec{X}, Y)$ is a $\Sigma^1_1$-formula such that

$$V^1 \vdash \forall \vec{x} \forall \vec{X} \exists Y \varphi(\vec{x}, \vec{X}, Y).$$

Then there is an IITER problem $Q_F$ with graph $\psi_F(\vec{x}, \vec{X}, Y)$ (as in (4.1)) and an $\text{FAC}^0$-function $G(\vec{x}, \vec{X}, Y)$ such that

$$\nabla^0 \vdash \psi_F(\vec{x}, \vec{X}, Y) \supset \varphi(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}, Y)).$$

(4.16)

In what follows, when we say that a theory $\mathcal{T}$ proves a sequent

$$\varphi_1, \ldots, \varphi_k \rightarrow \psi_1, \ldots, \psi_l,$$

we mean that $\mathcal{T}$ proves

$$\bigwedge_{i=1}^k \varphi_i \supset \bigvee_{j=1}^l \psi_j.$$

Buss [Bus86] originally proved his witnessing theorem for $V^1$ via a witnessing lemma. Here, we do the same; that is to say, we use a new-style witnessing lemma in order to prove Theorem 4.18.

**Lemma 4.19** (New-style Witnessing Lemma for $V^1$) Suppose that the theory $V^1$ proves a sequent $\Gamma(\vec{a}, \vec{A}) \rightarrow \Delta(\vec{a}, \vec{A})$ of the form

$$\ldots, \exists \gamma \varphi_i(\vec{x}_i), \ldots, \Lambda \rightarrow \Pi, \ldots, \exists \gamma \psi_j(\vec{y}_j), \ldots$$

(4.17)

where $\varphi_i, \psi_j, \Lambda$ and $\Pi$ are $\Sigma^0_1$-formulae. Then there is an IITER problem $Q_F$ with graph $\Psi_F$ and $\text{FAC}^0$-functions $\bar{G}$ such that $\nabla^0$ proves the sequent $\Gamma' \rightarrow \Delta'$, which is

$$\ldots, \varphi_i(\vec{A}_i), \ldots, \Lambda, \psi_F(\vec{a}, \vec{A}, \vec{\beta}, \gamma) \rightarrow \Pi, \ldots, \psi_j(G_j(\vec{a}, \vec{A}, \vec{\beta}, \gamma)), \ldots$$

(4.18)

**Proof Sketch of Theorem 4.18 from Lemma 4.19.** The approach we follow here is similar to the one used to prove Theorem V.5.1 [CN10]. For the sake of simplicity, assume that $\varphi(\vec{x}, \vec{X}, Y)$ is of the form $\exists Z \psi(\vec{x}, \vec{X}, Y, Z)$, where $\psi \in \Sigma^0_1$. The trick is to encode $Y$ and $Z$ into a single string $W = (Y, Z)$ using the $\text{FAC}^0$-function $\text{Row}$. By Lemma 2.38, this will result in

$$V^1(\text{Row}) \vdash \exists W \psi(\vec{x}, \vec{X}, W^{(0)}, W^{(1)}).$$

Since $V^1(\text{Row})$ is a conservative extension of $V^1$ and the $\Sigma^0_1(\text{Row})$-formula $\psi$ is equivalent to a $\Sigma^0_1$-formula $\psi'(\vec{x}, \vec{X}, W)$, provable in $V^0(\text{Row})$ (see Theorem 2.36), it follows that

$$V^1 \vdash \exists W \psi'(\vec{x}, \vec{X}, W).$$

By Lemma 4.19, we obtain an IITER problem $Q_F$ with graph $\psi_F$ and an $\text{FAC}^0$-function $H$ such that the theory

$$\nabla^0 \vdash \psi_F(\vec{x}, \vec{X}, Z) \supset \psi'(\vec{x}, \vec{X}, H(\vec{x}, \vec{X}, Z)).$$

(4.19)

Since $\nabla^0$ is a universal conservative extension of $V^0(\text{Row}, H)$, it follows that

$$\nabla^0 \vdash \psi_F(\vec{x}, \vec{X}, Z) \supset \psi(\vec{x}, \vec{X}, H(\vec{x}, \vec{X}, Z)^{[0]}, H(\vec{x}, \vec{X}, Z)^{[1]}).$$
4.3. Inflationary Polynomial Local Search and V^1

To complete the proof, we define G as follows:

\[ G(\vec{x}, \vec{X}, Z) = H(\vec{x}, \vec{X}, Z)^{[0]}. \]

Then

\[ \nabla^0 \vdash \psi_F(\vec{x}, \vec{X}, Z) \supset \varphi(\vec{x}, \vec{X}, G(\vec{x}, \vec{X}, Z)). \]

A first attempt at proving Lemma 4.19 is to consider an LK-V^1 proof of (4.17). Then, by structural induction on the depth of a sequent S in \( \pi \), we try to show the conclusion of Lemma 4.19. The problem with this approach is that the \( \Sigma^B_1 \)-COMP axiom

\[ (\exists X \leq t)(\forall z < y)(X(z) \leftrightarrow \varphi(z)), \]

where \( \varphi \in \Sigma^B_1 \), is in general not equivalent to a \( \Sigma^B_1 \)-formula. As a result, the proof \( \pi \) may contain formulae that are not \( \Sigma^B_1 \). To circumvent this obstacle, we need to work with a slightly different theory \( \tilde{V}^1 \) equivalent to \( V^1 \). For that, first consider the following definitions:

**Definition 4.20** ([CN10]) Let \( \psi(\vec{x}, \vec{X}) \) be an \( \mathcal{L}^2_A \)-formula. Then \( \psi \) is a single-\( \Sigma^1_1 \)-formula if \( \psi \) is of the form \( \exists Y \varphi(\vec{x}, \vec{X}, Y) \), where \( \varphi \) is a \( \Sigma^B_0 \)-formula. If \( \psi \) is of the form \( (\exists Y \leq t) \varphi(\vec{x}, \vec{X}, Y) \), where \( \varphi \) is a \( \Sigma^B_0 \)-formula and \( t \) is an \( \mathcal{L}^2_A \)-term not involving \( Y \), then \( \psi \) is a single-\( \Sigma^B_1 \)-formula.

**Definition 4.21** ([CN10]) The theory \( \tilde{V}^1 \) is axiomatized by the axioms of \( V^0 \) plus the single-\( \Sigma^B_1 \)-IND axiom scheme.

Below, we merely state that \( \tilde{V}^1 = V^1 \) without proof. A full proof of it can be found in [CN10, Theorem VI.4.8].

**Theorem 4.22** ([CN10]) The theories \( \tilde{V}^1 \) and \( V^1 \) are the same.

The sequent calculus LK-\( \tilde{V}^1 \) for \( \tilde{V}^1 \) is essentially LK-V^0 (cf. Definition 2.8) augmented with the single-\( \Sigma^B_1 \)-IND rule, which is

\[
\frac{\chi(b), \Gamma \rightarrow \Delta, \chi(b+1)}{\chi(0), \Gamma \rightarrow \Delta, \chi(t)}
\]

where \( b \) is an eigenvariable and cannot appear in the lower sequent.

The sequent calculus LK-\( \tilde{V}^1 \) satisfies the following property, whose proof can be found in [CN10]:

**Theorem 4.23** ([CN10]) Suppose that \( \tilde{V}^1 \) proves a sequent \( \Gamma \rightarrow \Delta \) consisting only of single-\( \Sigma^1_1 \)-formulae. Then there is an LK-V^1 proof \( \pi \) of \( \Gamma \rightarrow \Delta \) such that every formula in \( \pi \) is a single-\( \Sigma^1_1 \)-formula.

We are now ready to prove Lemma 4.19. The proof technique we use to prove Lemma 4.19 is similar to the one used for Theorem VI.4.1 in [CN10, page 154] (which is a witnessing
theorem for $V^1$), which adopts the same proof technique as Buss (cf. [Bus86, Theorem 5]). Again, we note that the constructions we developed in Section 4.2 are merely used here in order to present the proof of Lemma 4.19 clearly and swiftly. In particular, the output of the IITER problem $Q_F(\tilde{a}, \tilde{A}, \tilde{\beta})$, from the statement of Lemma 4.19, does not play a direct role in finding a witness to one of the formulae in the succedent of the sequent $\Gamma' \rightarrow \Delta'$ in (4.18). Instead, a witness to one of the formulae in $\Delta'$ is extracted from a solution $\gamma$ of $Q_F(\tilde{a}, \tilde{A}, \tilde{\beta})$ by using an FAC$^0$-function $G_j$ in $\tilde{G}$ (see Lemma 4.19).

Proof of Lemma 4.19. Since $\tilde{V}^1$ and $V^1$ are the same, it follows that $\tilde{V}^1$ proves (4.17). By Theorem 4.23, let $\pi$ be an LK-$\tilde{V}^1$ proof of (4.17) such that every formula in $\pi$ is a single-$\Sigma_1$-formula. We show that $V^0$ proves the conclusion of Lemma 4.19 by induction on the depth of a sequent $S$ in $\pi$. The inductive proof splits into cases, depending on whether $S$ is an initial sequent or generated by the use of an inference rule. The most crucial case is the case of the single-$\Sigma_1$-IND rule. Nevertheless, we will cover most cases here, for the sake of illustration.

Suppose that $S$ is an initial sequent. If $S$ is a logical axiom, then it is straightforward. So, assume that $S$ is a non-logical one. If $S$ is an equality axiom, then it is also straightforward. Hence, consider the case when $S$ consists of a $\Sigma_0$-COMP axiom of the form

$$\rightarrow (\exists X \leq b)(\forall y \leq b)[X(y) \leftrightarrow \psi(y, b, \tilde{a}, \tilde{A})].$$

We want to show that there is an IITER problem $Q_F$ and an FAC$^0$-function $G$ such that $\tilde{V}^0$ proves that if $\gamma$ is a solution to $Q_F(b, \tilde{a}, \tilde{A})$, then $G(b, \tilde{a}, \tilde{A}, \gamma)$ witnesses the formula on the right-hand side of $S$. For that, we define $G$ as follows:

$$G(b, \tilde{a}, \tilde{A}, \gamma)(y) \leftrightarrow y < |A| \land A(y).$$

Note that $G$ is an FAC$^0$-function, since it is $\Sigma_0^b$-bit definable. Then we let $Q_F$ be specified by an FAC$^0$-function $F$ and a term $t$ such that the only solution to $Q_F$ is $\emptyset$ and $F(\emptyset) = \emptyset$.

Suppose that $S$ is obtained by the application of an inference rule. The weakening rules are easy. The case of the exchange and the $\neg$ introduction rules are straightforward. The contraction rules can be derived from cut and exchanges. Hence, first suppose that $S$ is obtained by the application of the $\exists$-right string rule. Therefore, $S$ is the bottom sequent of

$$\Lambda \rightarrow \Pi, \psi(A)$$

$$\Lambda \rightarrow \Pi, \exists Y \psi(Y)$$

The variable $A$ is not an eigenvariable, therefore, without loss of generality, appears in the lower sequent. By the induction hypothesis, let $Q_{F_1}$ be the IITER problem for the upper sequent. We define the FAC$^0$-function $G$ that may witness the new quantifier $\exists Y$ as follows:

$$G(\tilde{a}, \tilde{A}, A, \tilde{\beta}, \gamma)(y) \leftrightarrow y < |A| \land A(y).$$

The IITER problem $Q_F$ for the lower sequent is defined to be $Q_{F_1}$. 
Suppose that $S$ is obtained by the application of the $\exists$-left string rule. Hence, $S$ is the bottom sequent of

$$
\phi(B), \Lambda \rightarrow \Pi \\
\exists X \phi(X), \Lambda \rightarrow \Pi
$$

Here, $B$ is an eigenvariable and cannot occur in the lower sequent. By the induction hypothesis, let $Q_{F_1}$ be the IITER problem and $G^1_1, \ldots, G^1_j$ be the witnessing functions for the upper sequent. Note here that $Q_{F_1}$ and each $G^1_j$ will take as inputs $\vec{a}, \vec{A}, \vec{B}, \vec{\tilde{B}}$ and $\vec{a}, \vec{A}, \vec{B}, \vec{\tilde{B}}, \gamma$, respectively. Then the IITER problem $Q_F(\vec{a}, \vec{A}, \vec{B}, \vec{\tilde{B}})$ is defined to be $Q_{F_1}(\vec{a}, \vec{A}, \vec{B}, \vec{\tilde{B}})$, where $\vec{B}$ is intended to be a witness for the new quantifier $\exists X$, and each witnessing function $G_j(\vec{a}, \vec{A}, \vec{B}, \vec{\tilde{B}}, \gamma)$ for the lower sequent is defined to be $G^1_j(\vec{a}, \vec{A}, \vec{B}, \vec{\tilde{B}}, \gamma)$.

Suppose that $S$ is obtained by the application of the $\forall$-right number rule (the case of the $\exists$-left number rule is treated similarly). Hence, $S$ is the bottom sequent of

$$
b \leq t, \Lambda \rightarrow \Pi, \psi(b) \\
\Lambda \rightarrow \Pi, (\forall y \leq t) \psi(y)
$$

Note that $b$ is an eigenvariable, therefore, it cannot occur in the lower sequent. Now, let $Q_{F_1}$ and $G^1_1, \ldots, G^1_j$ be the IITER problem and the witnessing functions, respectively, for the upper sequent obtained by induction hypothesis. Let $g(\vec{a}, \vec{A})$ be an $\text{FAC}^0$-function that computes the least $y < t$ such that $\neg \psi(y)$ is true, or $t$, if no such $y$ exists. Then $g$ can be defined as follows:

$$y = g(\vec{a}, \vec{A}) \leftrightarrow y \leq t \land [y < t \rightarrow \neg \psi(y)] \land (\forall y < t) \psi(y).$$

By the constructions in Section 4.2, let $Q_{F_2}$ be an IITER problem that computes $g$. Thus, $Q_{F_2}(\vec{a}, \vec{A}) = g(\vec{a}, \vec{A})$. Then $Q_F$ is the composition of $Q_{F_2}$ and $Q_{F_1}$:

$$Q_F(\vec{a}, \vec{A}, \vec{\tilde{B}}) = Q_{F_1}(Q_{F_2}(\vec{a}, \vec{A}), \vec{a}, \vec{A}, \vec{\tilde{B}}).$$

We will now define the witnessing functions $G_j$ for the lower sequent. But first, let $\psi_{F_2}, \psi_{F_1}$ and $\psi_{F_1}$ be the graphs of $Q_F, Q_{F_1}$ and $Q_{F_2}$, respectively. Then by Lemma 4.16, let $\Pi_1(\vec{a}, \vec{A}, \vec{\tilde{B}}, \gamma)$ and $\Pi_2(\vec{a}, \vec{A}, \vec{\tilde{B}}, \gamma)$ be some $\text{AC}^0$-functions such that (omitting the parameters $\vec{a}, \vec{A}, \vec{\tilde{B}}$)

$$\bigvee^1 \vdash \psi_{F_2}(\gamma) \supset \psi_{F_1}(\Pi_1(\gamma)) \land \psi_{F_1}((\Pi_1(\gamma))_1, \Pi_2(\gamma)).$$

Note here that $(\Pi_1(\gamma))_1 = g(\vec{a}, \vec{A})$. Then we define

$$G_j(\gamma) = G^1_j((\Pi_1(\gamma))_1, \Pi_2(\gamma)).$$

Suppose that $S$ is either obtained by the application of the $\lor$-left rule or the $\land$-right rule, both are handled in a similar way. Hence, we only consider the case when $S$ is obtained by the application of the $\land$-right rule. Therefore, $S$ is the bottom sequent of

$$S_1 \equiv \Lambda \rightarrow \Pi, \varphi_1 \quad S_2 \equiv \Lambda \rightarrow \Pi, \varphi_2$$

$$\Lambda \rightarrow \Pi, (\varphi_1 \land \varphi_2)$$
Note that, for any $\Sigma^0_1$-formula $\phi$, there is an $\text{lITER}$ problem $Q_\phi$ that “evaluates” $\phi$. To see this, one can easily define an $\text{FAC}^0$-function $f_\phi$ that outputs 1, if $\phi$ is true, and outputs 0, otherwise. Then define $Q_\phi$ to be the $\text{lITER}$ problem that computes $f_\phi$. Now, let $Q_{\phi_1}$ and $Q_{\phi_2}$ be the $\text{lITER}$ problems that evaluate $\phi_1$ and $\phi_2$, respectively, and let $Q(\vec{a}, \vec{A}, \vec{B})$ be any $\text{lITER}$ problem. Furthermore, let $Q_{F_i}$ be the $\text{lITER}$ problem obtained by the induction hypothesis for $S_i$, for $i = 1, 2$. We are now going to define the $\text{lITER}$ problem $Q_F$ for the bottom sequent. Intuitively, $Q_F$ first runs $Q_{\phi_1}$. If $Q_{\phi_1}$ says that $\phi_1$ is false, then $Q_F$ runs $Q_{F_1}$ and finishes. Otherwise, it continues its process by running $Q_{\phi_2}$. If $Q_{\phi_2}$ says that $\phi_2$ is false, then $Q_F$ finishes its computation by running $Q_{F_2}$. Otherwise, it runs $Q(\vec{a}, \vec{A}, \vec{B})$ and finishes. More formally, let $P$ be defined by

$$P(\vec{a}, \vec{A}, \vec{B}) = \begin{cases} Q_{F_1}(\vec{a}, \vec{A}, \vec{B}) & \text{if } Q_{\phi_1}(\vec{a}, \vec{A}) = 0 \\ Q(\vec{a}, \vec{A}, \vec{B}) & \text{otherwise,} \end{cases}$$

Then we define $Q_F$ in the following way:

$$Q_F(\vec{a}, \vec{A}, \vec{B}) = \begin{cases} Q_{F_1}(\vec{a}, \vec{A}, \vec{B}) & \text{if } Q_{\phi_1}(\vec{a}, \vec{A}) = 0 \\ P(\vec{a}, \vec{A}, \vec{B}) & \text{otherwise,} \end{cases}$$

Observe that if $\neg \phi_1$ is true, then, applying Lemma 4.16 accordingly, there is an $\text{FAC}^0$-function $H_1$ such that, from a solution $\gamma$ to $Q_F(\vec{a}, \vec{A}, \vec{B})$, we get a solution $H_1(\vec{a}, \vec{A}, \vec{B}, \gamma)$ to $Q_{F_1}(\vec{a}, \vec{A}, \vec{B})$. Similarly, if $\neg \phi_2$ is true, then from $\gamma$, we get a solution $H_2(\vec{a}, \vec{A}, \vec{B}, \gamma)$ to $Q_{F_2}(\vec{a}, \vec{A}, \vec{B})$. We then define the witnessing functions $G_j$ for the lower sequent as follows (omitting the parameters $\vec{a}, \vec{A}, \vec{B}$):

$$G_j(\gamma) = \begin{cases} G^j_1(H_1(\gamma)) & \text{if } \neg \phi_1 \\ G^j_2(H_2(\gamma)) & \text{if } \neg \phi_2 \\ \gamma & \text{otherwise,} \end{cases}$$

Suppose that $S$ is obtained by the application of the single-$\Sigma^0_1$-IND rule. Then $S$ is the bottom sequent of

$$\psi(0), \Lambda \longrightarrow \Pi, \psi(b+1)$$
$$\psi(0), \Lambda \longrightarrow \Pi, \psi(b+1)$$

where (omitting the parameters $\vec{a}, \vec{A}, \vec{B}$) $\psi(b)$ is of the form $\exists \lambda \leq r(b) \psi_0(b, X)$ and

$$\Pi = \Pi', \exists \lambda \psi'_1(Y_1), \ldots, \exists \lambda \psi'_j(Y_j).$$

Here $\Pi', \psi'_1, \ldots, \psi'_j$ is a sequence of $\Sigma^0_1$-formulae. Let $\eta(b, \beta)$ denote the formula $|\beta| \leq r(b) \land \psi_0(b, \beta)$. By the induction hypothesis, let $Q_{F_i}$ be an $\text{lITER}$ problem specified by $F_i$ and $t_i$, with graph $\psi_{F_i}$ and $G^j_1, \ldots, G^j_i$ and $G^j_{i+1}$ be the witnessing functions for the formulae in $\Lambda, \psi(b+1)$ such that $\hat{\psi}^j$ proves the following (omitting the parameters $\vec{a}, \vec{A}$, and $\vec{A}$, where $\vec{A}$ are witnesses for the formulae in $\Lambda$):

$$\eta(b, \beta), \Lambda', \psi_{F_i}(b, \beta, \gamma) \longrightarrow \Pi'^j(G^j_i(b, \beta, \gamma)) \land \eta(b+1, G^j_{i+1}(b, \beta, \gamma))$$

(4.20)
4.3. Inflationary Polynomial Local Search and $\forall^1$

where $\Lambda'$ is the result of witnessing $\Sigma^1_1$-formulae in $\Lambda$ and leaving the rest unchanged and
\[
\Pi''(G_j^1(b, \beta, \gamma)) = \Pi', \psi'_i(G_j^1(b, \beta, \gamma)), \ldots, \psi'_i(G_j^1(b, \beta, \gamma)).
\]

Our goal is to construct an ITER problem $Q_f$ (with graph $\psi_f$) and FAC$^0$-functions $G_1, \ldots, G_l$ and $G_{l+1}$ such that $\nabla^0$ proves the following:
\[
\eta(0, \beta_0, \Lambda'), \psi_f(\beta_0, \gamma) \rightarrow \Pi''(G_j(\beta_0, \gamma)). \eta(t, G_{l+1}(\beta_0, \gamma)).
\] (4.21)

The intuitive idea behind the definition of $Q_f$ is that, assuming that $\eta(0, \beta_0)$ is true, we will repeatedly use $Q_{f_1}$ and $G_{l+1}$ in order to generate witnesses $\beta_1, \ldots, \beta_n$ for $\psi(1), \ldots, \psi(n)$, respectively, for $n \leq t$. If $n < t$, then $Q_{f_1}$ failed to generate a witness to $\psi(n + 1)$. Therefore, assuming that the hypothesis for (4.21) is true and using (4.20), we obtain our desired goal.

We assume that the search variable for $Q_f$ is of the form
\[
\gamma = (\bar{a}, \bar{A}, \beta_0, \bar{\lambda}) * \bar{s}_0 * \bar{s}_1 * \bar{s}_2 * \bar{s}_3 * \ldots * \bar{s}_{(m+1)}, S_m,
\]
where $s$ ($s$ is obtained from $t$ and the bounding term $r$, in the induction-formula $\psi$, and the bounding term $t_1$ for $Q_{f_1}$) is a suitable $L^2_A$-term that bounds $|\bar{a}, \bar{A}, \beta_0, \bar{\lambda}|, |\bar{s}_0|, \ldots, |S_m|$: the symbol $S_i$ denotes $i * \beta_i * \gamma * 1$ and $m \leq t$. Note here that, even though we omitted the subscripts to $\gamma$ in $S_i$, they are somehow implicit. Let us now define the transition function $F$ for $Q_f$. In the following, we again omit the parameters $\bar{a}, \bar{A}, \bar{\lambda}$ for $F$. As usual, we will drop the subscripts to $*$ in $F(\beta_0, \gamma)$. If $\gamma = 0$, then
\[
F(\beta_0, \gamma) = (\bar{a}, \bar{A}, \beta_0, \bar{\lambda}) * 0 * \beta_0 * \theta * 1.
\] (4.22)

Assume now that $\gamma \neq 0$ and suppose that $m < t$ and $\eta(m, \beta_m)$ is true. Then there are two cases to consider. First, if $|\gamma_m| \leq t_1 \land \gamma_m < F_i(m, \beta_m, \gamma_m)$ is true, then
\[
F(\beta_0, \gamma) = (\bar{a}, \bar{A}, \beta_0, \bar{\lambda}) * \bar{s}_0 * \ldots * \bar{s}_{m-1} * m * \beta_m * F_i(m, \beta_m, \gamma_m) * 1.
\] (4.23)

Second, if $\psi_{f_1}(m, \beta_m, \gamma_m)$, then
\[
F(\beta_0, \gamma) = (\bar{a}, \bar{A}, \beta_0, \bar{\lambda}) * \bar{s}_0 * \ldots * \bar{s}_m * (m + 1) * G_{l+1}(m, \beta_m, \gamma_m) * \theta * 1.
\] (4.24)

In all other cases, $F(\beta_0, \gamma) = \gamma$. Let $t_{q_f}$ be $(t + 2) \cdot s$ and $Q_f$ be specified by $F$ and $t_{q_f}$. Finally, we define the FAC$^0$-functions $G_j$, for $i = 1, \ldots, l + 1$, as follows:
\[
G_j(\beta_0, \gamma) = \begin{cases} 
\beta_0 & \text{if } t = 0 \\
G_j^1(m, \beta_m, \gamma_m) & \text{otherwise},
\end{cases}
\]

The fact that $\nabla^0$ proves (4.21) follows from (4.21)'s assumptions, from the following claim, the induction hypothesis and the definition of $G_j$ above. As a side remark, note that if $t = 0$, then $\nabla^0$ proves (4.21) trivially.

Claim 4.24 We reason in $\nabla^0$. Suppose that $t \neq 0$, $\eta(0, \beta_0)$ is true and
\[
\gamma = (\bar{a}, \bar{A}, \beta_0, \bar{\lambda}) * \bar{s}_0 * \ldots * \bar{s}_m
\]
is a solution to $Q_f(\beta_0)$, where $S_i$ is again of the form $i * \beta_i * \gamma * 1$. Then $\eta(m, \beta_m)$ is true; $\gamma_m$ is a solution to $Q_i(m, \beta_m)$; and either $\eta(m + 1, G_{l+1}(m, \beta_m, \gamma_m))$ or $\eta(t, G_{l+1}(\beta_0, \gamma))$ is true.

57
Proof of Claim 4.24. Since $\gamma$ is a solution to $Q_F(\beta_0)$, then we have two possibilities: either $\gamma = 0$ and $F(\beta_0, \gamma) = \gamma$, or

$$|\gamma| \leq t_QF \land \gamma < F(\beta_0, \gamma) \land |F(\beta_0, \gamma)| > t_QF \lor F(\beta_0, F(\beta_0, \gamma)) = F(\beta_0, \gamma).$$

Note that, by the definition of $F$, $\theta$ cannot be a solution to $Q_F(\beta_0)$ and $|F(\beta_0, \gamma)| \leq t_QF$. Therefore, we have that

$$\gamma \neq 0 \land \gamma < F(\beta_0, \gamma) = F(\beta_0, F(\beta_0, \gamma)). \quad (4.25)$$

The only way for (4.25) to hold is if (4.24) is true. This implies that $\eta(m, \beta_m)$ holds and $\psi_F(m, \beta_m, \gamma_m)$ is true; that is to say, $\gamma_m$ is a solution $Q_{F_i}(m, \beta_m)$. Hence, we are left with proving the following:

$$\neg \eta(m + 1, G_{t+1}(m, \beta_m, \gamma_m)) \lor \eta(t, G_{t+1}(\beta_0, \gamma)).$$

If $m + 1 = t$, then we are done. So, assume that $m + 1 < t$. For the sake of contradiction, assume that $\eta(m + 1, G_{t+1}(m, \beta_m, \gamma_m))$ holds. This means that $F(\beta_0, \gamma) < F(\beta_0, F(\beta_0, \gamma))$, which is a contradiction. Thus, we are done with the proof of the claim.

The case when $S$ is obtained by the application of the cut rule is treated in a similar way as the case of the single-$\Sigma_1^B$-IND rule. The other cases are straightforward (that is to say, the $\land$-left, the $\lor$-right, the $\exists$-right (number) and the $\forall$-left number rules).

Finally, combining Theorems 4.17 and 4.18 and the fact that $V^0$ is a universal conservative extension of $V^0$, we obtain the following theorem:

**Theorem 4.25** IITER is $\text{AC}^0$-many-one complete for the provably total search problems in $V^1$. Furthermore, the reduction is provable in the theory $V^0$. 

58
Generalized Polynomial Local Search and Improved Witnessing Theorem

Beckmann and Buss [BB09] introduced a class of total search problems called $\Pi^p_k$-PLS with $\Pi^p_g$-goals, which is a generalization of PLS problems: the function computing the initial solution, the profit function and the neighborhood function are polynomial-time functions; however, the predicate $F$ defining the set of candidate solutions is of $\Pi^p_k$-complexity; also, it comes with a stopping condition called "goal predicate", which is defined by a $\Pi^p_g$-predicate. They argued that these $\Pi^p_k$-PLS problems with $\Pi^p_g$-goals are "formalizable" in the theory $V^1$. Using these formalized $\Pi^p_k$-PLS problems with $\Pi^p_g$-goals, they characterized the class of $\Sigma^B_{g+1}$-definable search problems in $TV^{k+1}$, where the reduction is provable in $V^1$ (we stress here that their reduction is a $P$-many-one reduction). Furthermore, they showed that these formalized $\Pi^p_k$-PLS problems with $\Pi^p_g$-goals can be defined in $V^1$ in Skolem form: the defining properties can be proved in a Skolemized form, where the Skolem functions are simple polynomial-time functions. As a result, they obtained a stronger characterization of the class of $\Sigma^B_{g+1}$-definable search problems in $TV^{k+1}$. Namely, they showed that the class of $\Sigma^B_{g+1}$-definable search problems in $TV^{k+1}$ is characterized by the class of Skolemizable $\Pi^p_k$-PLS problems with $\Pi^p_g$-goals, where the reduction is provable in $V^1$. Then, using the definition of Skolemized $\Pi^p_k$-PLS problems with $\Pi^p_g$-goals as a template, they came up with a relativized $\forall\Sigma^B_1(\alpha)$-principle $P_{\Pi^p_k}(\alpha)$, which is provable in $TV^{k+1}(\alpha)$, but conjectured not to be a the-
orem of TV\(^k(\alpha)\).

In this chapter, we improve on the results of Beckmann and Buss [BB09] in several ways. As mentioned in the previous paragraph, Beckmann and Buss proved that these formalized \(\Pi^p_k\)-PLS problems with \(\Pi^p_k\)-goals characterize the \(\Sigma^B_{g+1}\)-definable search problems in \(TV^{k+1}\), over \(V^1\). The proof of their characterization relied on a new-style witnessing lemma (see Lemma 5 in [BB09]). Unfortunately, the proof of their new-style witnessing lemma contains an error (the case of the induction rule). To correct this, in Section 5.1, we redefine the class \(\Pi^p_k\)-PLS so that the predicate \(F\) defining the set of feasible points is of \(\Sigma^B_{k+1}\)-complexity, rather than \(\Pi^p_k\). Then we show that the class \(\Sigma^B_{k+1}\)-PLS, which we call \(\Sigma^B_{k+1}\)-PLS, has a complete problem class: namely, the \(\Sigma^B_{k+1}\)-ITER problems, which are a generalization of the iteration principle [CK98] (Beckmann and Buss did not use iteration-type problems in [BB09]; they stuck to \(\Pi^p_k\)-PLS problems all the way through). In Section 5.2, we show that, for \(0 \leq g \leq k\), these \(\Sigma^B_{k+1}\)-ITER problems with \(\Pi^p_k\)-goals are formalizable in \(V^0\) and that the totality of the resulting formalizable iteration problems can be expressed by a \(\forall\Sigma^B_{g+1}\)-formula that is provable in \(TV^{k+1}\). Additionally, in Section 5.2, we state an improved new-style witnessing theorem for \(TV^{k+1}\), which says that the class of \(\Sigma^B_{g+1}\)-definable search problems in \(TV^{k+1}\) is \(AC^0\)-many-one reducible to the class of formalizable \(\Sigma^B_{k+1}\)-ITER problems with \(\Pi^p_k\)-goals, where the reduction is provable in \(V^0\). In Section 5.3, we argue that the constructions of the formalizable \(\Pi^p_k\)-PLS problems described in [BB09, Section 3] can be carried out in \(V^0\), rather than \(V^1\). This is possible here, since we use \(\Sigma^B_{k+1}\)-ITER problems instead \(\Sigma^B_{k+1}\)-PLS problems (we do not know how this could be possible to complete using \(\Sigma^B_{k+1}\)-PLS; this is partly due to the nature of PLS problems, which use the cost function explicitly). Section 5.4 is devoted to the proof of the improved new-style witnessing theorem for \(TV^{k+1}\). In Section 5.5, we state another improvement on Beckmann and Buss’s results. Namely, we show that formalizable \(\Sigma^B_{k+1}\)-ITER problems with \(\Pi^p_k\)-goals can be defined in \(V^0\) in Skolem form, where the Skolem functions are \(AC^0\)-functions. Then we use these Skolemizable \(\Sigma^B_{k+1}\)-ITER problems with \(\Pi^p_k\)-goals in order to obtain a stronger characterization of the \(\Sigma^B_{g+1}\)-definable search problems in \(TV^{k+1}\), where the reduction is provable in \(V^0\). From the definition of Skolemizable \(\Sigma^B_{k+1}\)-ITER problems with \(\Pi^p_k\)-goals, we obtain a generic \(\forall\Sigma^B_{g+1}\)-principle \(Silter_{\alpha}\) from which we derive a class \(Silter(k)\) of \(\forall\Sigma^B_{g+1}\)-principles, which can be viewed as a subclass of \(\forall\exists AC^0\). Then, we proceed by showing that \(Silter(k)\) is \(AC^0\)-many-one complete for the \(\forall\Sigma^B_{g+1}\)-theorems of \(TV^{k+1}\), where the reduction is provable in \(V^0\).

5.1 Generalized Polynomial Local Search and Iteration Problems

As previously stated, Beckmann and Buss [BB09] defined the class \(\Pi^p_k\)-PLS of search problems. In the following definition, we will make slight changes to their definition and take the profit and the neighborhood functions to be \(AC^0\)-functions, instead of polynomial-time functions. Furthermore, we want the predicate defining the set of feasible points to now be given by a \(\Sigma^B_{k+1}\)-formula rather than a \(\Pi^p_k\)-relation. Also, Beckmann and Buss’s \(\Pi^p_k\)-PLS problems are minimization problems (that is to say, the goal is to find a solution with a minimal cost (or
5.1. Generalized Polynomial Local Search and Iteration Problems

conditions: AC

A concentrate on a subclass called where

F

PLS

problem with an additional

a

Σ

Let 0

Definition 5.2

The following definition makes that concept formal:

The PLS problem is in the class \( \Pi^0_k \), for some \( g < k \). In fact, the class \( \Sigma^B_{k+1} \) forms a kind of a hierarchy arranged according to the real difficulty of recognizing a valid solution. For this reason, Beckmann and Buss [BB09] defined the notion of PLS problems with goal predicates. The following definition makes that concept formal:

Definition 5.1 A \( \Sigma^B_{k+1} \)-PLS problem \( Q(\vec{x}, \vec{X}, Z) \) is specified by a \( \Sigma^B_{k+1} \)-formula \( F_Q(\vec{x}, \vec{X}, Z) \), AC\(^0\)-functions \( N_Q(\vec{x}, \vec{X}, Z) \) and \( P_Q(\vec{x}, \vec{X}, Z) \) and an \( \mathcal{L}^2_A \)-term \( t_Q(\vec{x}, \vec{X}) \) satisfying the following conditions:

1. For all \( \vec{x}, \vec{X} \) and \( Z \), if \( F_Q(\vec{x}, \vec{X}, Z) \) is true, then \( |Z| \leq t_Q(\vec{x}, \vec{X}) \).
2. For all \( \vec{x} \) and \( \vec{X} \), \( F_Q(\vec{x}, \vec{X}, \emptyset) \) is always true,
3. For all \( \vec{x}, \vec{X} \) and \( Z \), if \( F_Q(\vec{x}, \vec{X}, Z) \) is true, then \( F_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Z)) \) also holds,
4. For all \( \vec{x}, \vec{X} \) and \( Z \),

\[
N_Q(\vec{x}, \vec{X}, Z) = Z \lor P_Q(\vec{x}, \vec{X}, Z) < P_Q(\vec{x}, \vec{X}, N_Q(\vec{x}, \vec{X}, Z)).
\]

A solution to \( Q(\vec{x}, \vec{X}) \) is any string \( Z \) such that \( F_Q(\vec{x}, \vec{X}, Z) \) holds and \( N_Q(\vec{x}, \vec{X}, Z) = Z \).

As the reader may have noted, even though the complexity of verifying a valid solution to a \( \Sigma^B_{k+1} \)-PLS problem is in the class \( \Sigma^B_{k+1} \), it does not mean that it cannot be done in some lower quantifier complexity class \( \Pi^B_g \), for some \( g < k \). We will not be working directly with \( \Sigma^B_{k+1} \)-PLS problems. As in the case of IPLS, we will concentrate on a subclass called \( \Sigma^B_{k+1} \)-ITER, which is complete for \( \Sigma^B_{k+1} \)-PLS.

Definition 5.3 A \( \Sigma^B_{k+1} \)-ITER problem \( Q(\vec{x}, \vec{X}, Z) \) is specified by a \( \Sigma^B_{k+1} \)-formula \( C(\vec{x}, \vec{X}, Z) \), which defines the set \( \{ Z: C(\vec{x}, \vec{X}, Z) \} \) of candidate solutions to \( Q(\vec{x}, \vec{X}) \), an \( \mathcal{L}^2_A \)-term \( t(\vec{x}, \vec{X}) \) and an AC\(^0\)-function \( F(\vec{x}, \vec{X}, Z) \) (usually called \( Q \)’s transition function) satisfying the following conditions:

(a) For all \( \vec{x}, \vec{X}, Z \), if \( C(\vec{x}, \vec{X}, Z) \) is true, then \( |Z| \leq t(\vec{x}, \vec{X}) \),
(b) For all \( \vec{x}, \vec{X} \), \( C(\vec{x}, \vec{X}, \emptyset) \) is always true,
(c) For all \( \vec{x}, \vec{X}, Z \), if \( C(\vec{x}, \vec{X}, Z) \) is true, then so is \( C(\vec{x}, \vec{X}, F(\vec{x}, \vec{X}, Z)) \).
A solution to $Q(\bar{x}, \bar{X})$ is any string $Z$ such that $C(\bar{x}, \bar{X}, Z)$ is true and $F(\bar{x}, \bar{X}, Z) \leq Z$. We call $C, F$ and $t$ the components of $Q$. We will sometimes denote $Q$ by $(C, F, t)$.

The following two lemmas state that $\Sigma_{k+1}^B$-ITER is AC$^0$-many-one complete for $\Sigma_{k+1}^B$-PLS. However, their proofs are straightforward and simply remark that the definition of a $\Sigma_{k+1}^B$-ITER is very similar to that of a $\Sigma_{k+1}^B$-PLS. In the definition of $\Sigma_{k+1}^B$-ITER, we sort of combined a candidate solution to a $\Sigma_{k+1}^B$-PLS problem and its profit into one string by merely concatenating them using the concatenating function $X *_2 Y$.

**Lemma 5.4** Every $\Sigma_{k+1}^B$-ITER problem is a $\Sigma_{k+1}^B$-PLS problem.

**Lemma 5.5** Every $\Sigma_{k+1}^B$-PLS problem is AC$^0$-many-one reducible to a $\Sigma_{k+1}^B$-ITER problem.

Naturally, one can extend the definition of a $\Sigma_{k+1}^B$-ITER problem to also include goal predicates:

**Definition 5.6** Let $0 \leq g \leq k$. Then we define a $\Sigma_{k+1}^B$-ITER problem with $\Pi_k^B$-goal to be a $\Sigma_{k+1}^B$-ITER problem $Q$ with an additional $\Pi_k^B$-formula $G$ satisfying the following conditions:

$$
\forall \forall \forall \forall Z \forall G(\bar{x}, \bar{X}, Z) \supset C(\bar{x}, \bar{X}, Z) \land F(\bar{x}, \bar{X}, Z) \leq Z,
$$

$$
\forall \forall \forall \forall Z \forall C(\bar{x}, \bar{X}, Z) \land F(\bar{x}, \bar{X}, Z) \leq Z \supset G(\bar{x}, \bar{X}, Z),
$$

where $C$ and $F$ are two of $Q$’s components. $Q$ will sometimes be denoted by $(C, F, t, G)$.

### 5.2 Generalized Iteration Problems and TV$^{\infty}$

We start this section by defining what it means for $\Sigma_{k+1}^B$-ITER problems with $\Pi_k^B$-goals to be formalizable in $V^0$. Then we show that the resulting formalizable problems are $\Sigma_{k+1}^B$-definable in TV$^{k+1}$. Finally, we state that the converse holds as well, which is the new-style witnessing theorem for TV$^{k+1}$.

**Definition 5.7** A $\Sigma_{k+1}^B$-ITER problem $Q$ is said to be *formalizable* in $V^0$ (or just formalizable) if conditions (a), (b) and (c) in Definition 5.3 are provable in $V^0$. Additionally, if $Q$ comes with a $\Pi_k^B$-goal $G$, then we require that conditions (d) and (e) in Definition 5.6 be also provable in $V^0$, in addition to (a), (b) and (c). The class of formalizable $\Sigma_{k+1}^B$-ITER problems is denoted by $[\Sigma_{k+1}^B]$-ITER]. Similarly, $[\Sigma_{k+1}^B]_{\exists}$-ITER problems with $\Pi_k^B$-goals] denotes the class of formalizable $\Sigma_{k+1}^B$-ITER problems with $\Pi_k^B$-goals.

**Theorem 5.8** Let $Q \in [\Sigma_{k+1}^B]$-ITER]. Then TV$^{k+1}$ proves that $Q$ is total.

**Proof**. Let $C, F$ and $t$ be the components of $Q$. We want to show that TV$^{k+1}$ proves that for all $\bar{x}, \bar{X}$, there always exists a $Z$ such that $C(\bar{x}, \bar{X}, Z)$ and $F(\bar{x}, \bar{X}, Z) \leq Z$. Observe that the maximal $Z_{\text{max}}$ such that $C(Z_{\text{max}})$ holds is always a solution. Since $C(\emptyset)$ is provable in $V^0$, using $\Sigma_{k+1}^B$-SMAX, TV$^{k+1}$ is able to find $Z_{\text{max}}$. We now need to show that $F(Z_{\text{max}}) \leq Z_{\text{max}}$. For the sake of contradiction, assume that $Z_{\text{max}} < F(Z_{\text{max}})$. Because $V^0$ proves $C(Z_{\text{max}}) \supset C(F(Z_{\text{max}}))$, we
end up with a contradiction to the maximality of $Z_{\max}$ in $C$. Thus, we have proved that $TV^{k+1}$ proves the totality of $Q$.

**Remark 5.9** Let $Q$ be a formalizable $\Sigma^B_{k+1}$-iter problem with $\Pi^g_k$-goal $G$. Then $TV^{k+1}$ proves the totality of $Q$ if, and only if, $TV^{k+1}$ proves $\forall \bar{x} \forall \bar{X} \exists Z G(\bar{x}, \bar{X}, Z)$.

Note that by the previous remark, problems that are in $[\Sigma^B_{k+1}$-iter with $\Pi^g_k$-goals], $\mathcal{F}$ are $\Sigma^B_{k+1}$-definable in $TV^{k+1}$. The following theorem, whose proof is carried out in Section 5.4, shows that the converse also holds:

**Theorem 5.10** (New-style Witnessing Theorem for $TV^{k+1}$) Let $k \geq 0$ and $0 \leq g \leq k$. Suppose that $\varphi(\bar{x}, \bar{X}, Y)$ is a $\Sigma^B_{g+1}$-formula and

$$TV^{k+1} \vdash \forall \bar{x} \forall \bar{X} \exists Y \varphi(\bar{x}, \bar{X}, Y).$$

Then there is a $\Sigma^B_{g+1}$-iter problem $Q$ with $\Pi^g_k$-goal $G$ and an $AC^0$-function $H$ such that $Q$ is formalizable and $\overline{V^0}$ proves

$$G(\bar{x}, \bar{X}, Z) \supset \varphi(\bar{x}, \bar{X}, H(\bar{x}, \bar{X}, Z)). \quad (5.1)$$

Our new-style witnessing theorem for $TV^{k+1}$ above is an improvement over Beckmann and Buss’s (see [BB09, Theorem 2]) in the following ways. Firstly, for $g = 0$, our characterization of the provably total search problems in $TV^{k+1}$ is in terms of a subclass of $\forall \exists AC^0$ (Beckmann and Buss [BB09] characterized the provably total search problems in $TV^{k+1}$ in terms of a subclass of TFNP, but not $\forall \exists AC^0$). Second, our reduction is stronger: our characterization uses $AC^0$-many-one reduction instead of $P$-many-one reduction. Finally, our reduction is provable over $\overline{V^0}$, instead of $V^1$.

Now, combining Theorem 5.8, Remark 5.9 and Theorem 5.10, and the fact that $\overline{V^0}$ is a universal conservative extension of $V^0$, we obtain the following corollary:

**Corollary 5.11** Let $0 \leq g \leq k$. Then $[\Sigma^B_{k+1}$-iter problems with $\Pi^B_k$-goals], $\mathcal{F}$ is $AC^0$-many-one complete for the $\Sigma^B_{g+1}$-definable search problems in $TV^{k+1}$, over $\overline{V^0}$.

When proving Theorem 5.10 above, it is convenient to consider the “single” version\(^1\) of $TV^{k+1}$, which is denoted $\overline{TV}^{k+1}$. Thus, the following definitions:

**Definition 5.12** Let $\mathcal{L} \supseteq \mathcal{L}_A^2$. Then

1. $\hat{\Sigma}^B_{=0}(\mathcal{L}) = \tilde{\Pi}^B_{=0}(\mathcal{L}) = \Sigma^B_{A} (\mathcal{L})$,

2. For $k \geq 0$, $\varphi \in \hat{\Sigma}^B_{=k+1}(\mathcal{L})$ if $\varphi$ is of the form $(\exists Y \leq k) \psi(\bar{x}, \bar{X}, Y)$, where $\psi \in \hat{\Pi}^B_{=k}(\mathcal{L})$,

3. For $k \geq 0$, $\hat{\Pi}^B_{=k+1}(\mathcal{L})$ is defined dually to $\hat{\Sigma}^B_{=k+1}(\mathcal{L})$.

---

\(^1\)Historically, the idea of having “single” versions of bounded arithmetic theories originated from [Pol97], but in the context of first-order bounded arithmetic.
5. Generalized Polynomial Local Search and Improved Witnessing Theorem

Definition 5.13 Let $\mathcal{L} \supseteq \mathcal{L}^2_A$. Then $\hat{\Sigma}_B^V(\mathcal{L}) = \Sigma_B^V(\mathcal{L})$. A formula $\varphi \in \hat{\Sigma}_B^{k+1}(\mathcal{L})$ if either $\varphi \in \hat{\Sigma}_B^k(\mathcal{L})$ or there is a $j < k + 1$ such that $\varphi \in (\hat{\Sigma}_B^j \cup \hat{\Pi}_B^j)(\mathcal{L})$. Similarly, a formula $\varphi \in \hat{\Pi}_B^{k+1}(\mathcal{L})$ if either $\varphi \in \hat{\Pi}_B^k(\mathcal{L})$ or there is a $j < k + 1$ such that $\varphi \in (\hat{\Sigma}_B^j \cup \hat{\Pi}_B^j)(\mathcal{L})$.

Again, in Definition 5.12 and 5.13, if $\mathcal{L} = \mathcal{L}^2_A$, then we shall drop mention of $\mathcal{L}$.

The following lemma is easily provable from Lemma 2.38:

Lemma 5.14 For every $\Sigma_B^{k+1}$-formula $\varphi$, there is a $\hat{\Sigma}_B^{k+1}$-formula $\varphi'$ such that $V^0(\text{Row})$ proves $\varphi \leftrightarrow \varphi'$.

Definition 5.15 The theory $\hat{T}V^{k+1}$ is axiomatized by the axioms of $V^0$ and the $\hat{\Sigma}_B^{k+1}$-SIND axiom scheme.

Clearly, the class $\hat{\Sigma}_B^{k+1} \subseteq \Sigma_B^{k+1}$. Thus, $\hat{T}V^{k+1} \subseteq TV^{k+1}$. By Lemma 5.14, the following corollary follows:

Corollary 5.16 $\hat{T}V^{k+1} = TV^{k+1}$.

5.3 Some More Constructions of Generalized Iteration Problems

The constructions that we present in this section are a generalization of the constructions in Section 4.2. However, here, we do not have to worry about making sure that the problems are inflationary, which makes things slightly easier. Even though the constructions will be cleaner, we need to deal with more of them. Namely, we need to deal with the constructions of pseudo-iteration of formalizable $\Sigma_B^{k+1}$-ITER problems and formalizable $\Sigma_B^{k+1}$-ITER problems that “decide” the truth of $\Sigma_B^k$- and $\Pi_B^k$-properties, in addition to formalizable $\Sigma_B^{k+1}$-ITER problems that represent $AC^0$-functions and formalizable $\Sigma_B^{k+1}$-ITER problems that are obtained by $\mathcal{F} \mathcal{G}$-combinations.

The idea behind the constructions in Section 4.2, and what will follow, originated from [BB09]. However, the ones in this section will be very close to the ones in [BB09], in the sense that the definitions of the components of formalizable $\Sigma_B^{k+1}$-ITER problems that we shall see in this section are very similar to the ones found in [BB09]. This similarity is very important later when we discuss Skolemization. The minor differences are that now we will be using iteration problems, instead of PLS problems, and we will be in the two-sorted setting.

Iteration problems have certain advantages over PLS problems, mainly that a candidate solution and its profit is now concatenated together into just one string. This ability to abstract away the profit of a candidate solution is very important. If we were to use formalizable $\Sigma_B^{k+1}$-PLS instead of $\Sigma_B^{k+1}$-ITER problems in the following constructions, then, in some cases, the cost function may involve the multiplication operation over strings, which is not in $AC^0$.

In what follows, we adopt the same conventions as in Section 4.2.
AC^0-functions as formalized ITER problems. Note that lITER is a subset of ITER. In the previous chapter, we showed how AC^0-functions are represented by lITER problems. Therefore, it is clear that an AC^0-function \( g \) can be represented by an ITER problem \( Q_G \) so that the only possible solution to \( Q_G \) is a string \( Z \) such that \( Z \) is of the form
\[
\langle \vec{x}, \vec{X} \rangle \ast_s G(\vec{x}, \vec{X}) \ast_{2s} 1.
\]
Again, \( s = s(\vec{x}, \vec{X}) \) is an \( \mathcal{L}_A^2 \)-term that bounds the length of every component of \( Z \). Also, it is clear that an AC^0-function \( g \) can be represented by an ITER problem \( Q_g \) so that the only possible solution to \( Q_g \) is a string \( Z \) such that \( Z \) is of the form
\[
\langle \vec{x}, \vec{X} \rangle \ast_s g(\vec{x}, \vec{X}) \ast_{2s} 1
\]

Combining formalized \( \Sigma^B_{k+1} \)-ITER problems. In this paragraph, we show how to formally define the \( \mathcal{F} \mathcal{G} \)-combination of two \( \Sigma^B_{k+1} \)-ITER problems. Let \( \mathcal{F} = F(\vec{X}) \) and
\[
\mathcal{G} = g(\vec{x}, \vec{X}, Z), G(\vec{x}, \vec{X}, Z)
\]
be AC^0-functions. Furthermore, let \( Q_1(\vec{x}, \vec{X}) \) and \( Q_2(\vec{x}, \vec{X}) \) be \( \Sigma^B_{k+1} \)-ITER problems. Then the \( \mathcal{F} \mathcal{G} \)-combination of \( Q_1(\vec{x}, \vec{X}) \) and \( Q_2(\vec{x}, \vec{X}) \), denoted
\[
F(\langle Q_1(\vec{x}, \vec{X}), Q_2(g(\vec{x}, \vec{X}, Q_1(\vec{x}, \vec{X}))), G(\vec{x}, \vec{X}, Q_1(\vec{x}, \vec{X})))
\]
is a \( \Sigma^B_{k+1} \)-ITER problem \( Q \) that is defined so that \( Y = Q(\vec{x}, \vec{X}) \) if, and only if, there is some \( Y_1 = Q_1(\vec{x}, \vec{X}) \) and some \( Y_2 = Q_2(g(\vec{x}, \vec{X}, Q_1(\vec{x}, \vec{X}))), G(\vec{x}, \vec{X}, Q_1(\vec{x}, \vec{X}))) \) such that \( Y = F(\langle Y_1, Y_2 \rangle) \).

As a reminder, \( \mathcal{F} \mathcal{G} \)-combination is powerful enough in order to obtain pairing and composition of lITER problems. In the context of \( \Sigma^B_{k+1} \)-ITER problems, nothing changes: from \( \mathcal{F} \mathcal{G} \)-combination, we still are able to obtain pairing and composition. Also, \( \mathcal{F} \mathcal{G} \)-combination is powerful enough in order to obtain \( \Sigma^B_{k+1} \)-ITER problems that are defined by case distinctions (cf. Section 4.2).

Now, let \( Q_1 = (C_1, F_1, t_1) \) and \( Q_2 = (C_2, F_2, t_2) \). The idea behind the definition of \( Q \) is similar to the definition of \( \mathcal{F} \mathcal{G} \)-combination of lITER problems. A candidate solution \( Z \) to \( Q \) will either be \( \emptyset \) or of the form
\[
\langle \vec{x}, \vec{X} \rangle \ast_b Y \ast_{2b} Z_1 \ast_{3b} Z_2 \ast_{4b} \delta,
\]
where \( b = b(\vec{x}, \vec{X}) \) is a suitable \( \mathcal{L}_A^2 \)-term that bounds the components of \( Z \). Intuitively, when \( \delta \) is equal to 1, then \( Z_1 \) is a candidate solution to \( Q_1(\vec{x}, \vec{X}) \). The search problem \( Q \) will then iterate \( F_1 \) on \( Z_1 \) until a solution to \( Q_1(\vec{x}, \vec{X}) \) is found. At that point, \( \delta \) is then incremented to 2. That incremental action signals the beginning of the process of finding a solution to \( Q_2(g(\vec{x}, Z_1)), G(\vec{x}, Z_1)) \) (omitting the parameters \( \vec{x}, \vec{X} \)). At that stage, the search problem \( Q \) will iterate \( F_2 \) on a candidate solution \( Z_2 \) of \( Q_2(g(\vec{x}, Z_1)), G(\vec{x}, Z_1) \) until a solution is found. When that happens, then \( \delta \) is again incremented to 3 and \( Y \) will now store the output of \( Q(\vec{x}, \vec{X}) \). That final step marks the end of the computation.

In what follows, we omit the subscript to \( \ast \). Formally, the formula \( C(\vec{x}, \vec{X}, Z) \) is defined so that (omitting the parameters \( \vec{x}, \vec{X} \)):
5. Generalized Polynomial Local Search and Improved Witnessing Theorem

- $C(\emptyset)$ is true if, and only if, $C_1(\emptyset)$ holds;
- $C(\langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast Z_1 \ast \emptyset \ast 1)$ is true if, and only if, $C_1(Z_1)$ holds;
- $C(\langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast Z_1 \ast Z_2 \ast 2)$ is true if, and only if, $C_1(Z_1) \land F_1(Z_1) \leq Z_1$ holds (that is to say, $Z_1$ is a solution to $Q_1(\langle \bar{x}, \bar{X} \rangle)$ and $C_2(\bar{g}((Z_1)_1), \bar{G}((Z_1)_1), Z_2)$ also holds;
- $C(\langle \bar{x}, \bar{X} \rangle \ast Y \ast Z_1 \ast Z_2 \ast 3)$ is true if and only if
  \[
  C_1(Z_1) \land F_1(Z_1) \leq Z_1 \\
  \land C_2(\bar{g}((Z_1)_1), \bar{G}((Z_1)_1), Z_2) \land F_2(\bar{g}((Z_1)_1), \bar{G}((Z_1)_1), Z_2) \leq Z_2 \\
  \land Y = \mathcal{F}((Z_1)_1, (Z_2)_1)).
  \]

Next, we define $Q$’s transition function $F(\langle \bar{x}, \bar{X}, Z \rangle)$ so as to satisfy the following (again, omitting the parameters $\bar{x}, \bar{X}$):

- $F(\emptyset) = \langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast \emptyset \ast 1$,
- For $Z = \langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast Z_1 \ast \emptyset \ast 1$,
  \[
  F(Z) = \begin{cases} 
  \langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast F_1(Z_1) \ast \emptyset \ast 1 & \text{if } Z_1 < F_1(Z_1) \\
  \langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast Z_1 \ast F_2(\bar{g}((Z_1)_1), \bar{G}((Z_1)_1), \emptyset) \ast 2 & \text{otherwise},
  \end{cases}
  \]
- For $Z = \langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast Z_1 \ast Z_2 \ast 2$,
  \[
  F(Z) = \begin{cases} 
  \langle \bar{x}, \bar{X} \rangle \ast \emptyset \ast Z_1 \ast F_2(\bar{g}((Z_1)_1), \bar{G}((Z_1)_1), Z_2) \ast 2 & \text{if } Z_2 < F_2(\bar{g}((Z_1)_1), \bar{G}((Z_1)_1), Z_2) \\
  \langle \bar{x}, \bar{X} \rangle \ast \mathcal{F}((Z_1)_1, (Z_2)_1)) \ast Z_1 \ast Z_2 \ast 3 & \text{otherwise},
  \end{cases}
  \]
- For all other $Z$, $F(Z) = Z$.

Finally, we define $t(\bar{x}, \bar{X}) = 4b(\bar{x}, \bar{X}) + |\text{POW}2(3)|$. Then let $Q = (C, F, t)$.

It is straightforward to verify that $Q$ is a $\Sigma^B_{k+1}$-ITER problem that correctly defines the $\mathcal{F}$$\mathcal{G}$-combination of $Q_1$ and $Q_2$. Furthermore, if $Q_1$ and $Q_2$ are formalizable, then so is $Q$.

**Pseudo-iteration of formalized $\Sigma^B_{k+1}$-ITER problems** The following construction is needed in the proof of the new-style witnessing lemma for Theorem 5.10. The critical case is the induction one, which will require some kind of iteration of a $\Sigma^B_{k+1}$-ITER problem $P_1$. The problem is that the number of iterations that we will face is of exponential-length. This means that a second-order variable is not enough to encode the entire computation of all the steps of the iteration. Instead, we use a side property $H$ that is preserved by iteration of $P_1$ to indirectly describe the result of an exponentially long iteration. This is called “pseudo-iteration”, a term coined by Beckmann and Buss [BB09].

Let $P_1$ be a $\Sigma^B_{k+1}$-ITER problem, $H(I, X, Y)$ be a $\Sigma^B_{k+1}$-formula and $p_H$ be an $\mathcal{L}^2_H$-term satisfying the following conditions:
5.3. Some More Constructions of Generalized Iteration Problems

(t1) For all $X$, $H(\emptyset, X, X)$ is always true,

(t2) For all $I, X, Y$ and $Z$, if $H(I, X, Y)$ is true and $Z = P_I(Y)$, then $H(I + 1, X, Z)$ holds,

(t3) For all $I, X$ and $Y$, if $H(I, X, Y)$ is true, then $|Y| \leq p_H(|X| + |I|)$.

Our goal is to define a $\Sigma_{k+1}^B$-ITER problem $Q$ so that if $Y = Q(M, X)$, then we want to make sure that $H(M, X, Y)$ holds. Let $m = (\omega(M))_2$ (see Definition 2.9). Note that $m$ may be a value that is exponential in the length of $X$. Then, intuitively, $X$ will be the input on which $P_I$ will be iterated and $m$ is the number of iterations. Basically, we wish to compute a sequence $Y_0, Y_1, \ldots, Y_m$ of strings such that $Y_0 = X$ and $Y_{i+1} = P_I(Y_i)$, for $i < m$. In the end, we want to have that $Y_m = Q(M, X)$. The search problem $Q$ will be denoted $PsIter[P_I, H]$.

Let $P_I = (C_1, F_1, t_1)$. The formula $C(M, X, Z)$ is defined so that $C(M, X, Z)$ is true if, and only if, either $Z = \emptyset$ or $Z = \langle M, X \rangle \ast b Y \ast 2b Z_1 \ast 3b I \ast 4b 1$ and

$$I \leq M \land H(I, X, Y) \land [I < M \supset C_1(Y, Z_1)].$$

Here, $b = b(M, X)$ is a suitable $\mathcal{L}_\Sigma^2$-term that bounds the length of the components of $Z$.

Next, define the function $F(M, X, Z)$ so as to satisfy the following conditions (omitting the parameters $M, X$ and subscripts to $\ast$):

- For $Z = \emptyset$, $F(Z) = \langle M, X \rangle \ast X \ast \emptyset \ast \emptyset \ast 1$,
- For $Z = \langle M, X \rangle \ast Y \ast Z_1 \ast I \ast 1$, $F(Z) = \begin{cases} \langle M, X \rangle \ast Y \ast F_1(Y, Z_1) \ast I \ast 1 & \text{if } I < M \land Z_1 < F_1(Y, Z_1) \\ \langle M, X \rangle \ast (Z_1) \ast 1 \ast \emptyset \ast I + 1 \ast 1 & \text{if } I < M \land F_1(Y, Z_1) \leq Z_1, \end{cases}$

- In all other cases, $F(Z) = Z$.

Finally, we define $t(M, X) = 4b(M, X) + 1$. Then let $Q = (C, F, t)$.

It is straightforward to verify that $Q$ is a $\Sigma_{k+1}^B$-ITER such that if $Y = Q(M, X)$, then $H(M, X, Y)$ is true. Furthermore, if $P_I$ is formalizable and conditions (t1), (t2) and (t3) are provable in $\mathcal{V}^0$, then $Q$ is also formalizable.

Deciding $\Sigma_{k+1}^B$- and $\Pi_{k+1}^B$-properties. We also want to have a formalizable $\Sigma_{k+1}^B$-ITER problem that decides the validity of either a $\Sigma_{k}^B$-formula or a $\Pi_{k}^B$-formula. More specifically, let $A(\bar{x}, \bar{X})$ be a $\Sigma_{k}^B$-formula of the form $\langle \exists Y \leq t \rangle B(\bar{x}, \bar{X}, Y)$. Then we want to define a formalizable $\Sigma_{k+1}^B$-ITER problem $Q_A$ such that if $A(\bar{x}, \bar{X})$ is true, then $Q_A(\bar{x}, \bar{X}) = \langle 1, Y \rangle$, where $Y$ is the least string that witnesses $A(\bar{x}, \bar{X})$; otherwise, $Q_A(\bar{x}, \bar{X}) = \langle 0, POW(2(t)) \rangle$.

For $A(\bar{x}, \bar{X})$ of the form $\langle \forall Y \leq t \rangle B(\bar{x}, \bar{X}, Y)$, where $B \in \Sigma_{k-1}^B$, one can define a formalizable $\Sigma_{k+1}^B$-ITER problem $Q_A$ that decides the validity of $A$ in a similar way. More precisely, if $A(\bar{x}, \bar{X})$ is true, then $Q_A(\bar{x}, \bar{X}) = \langle 1, POW(2(t)) \rangle$; otherwise, $Q_A(\bar{x}, \bar{X}) = \langle 0, Y \rangle$, where $Y$ is the least string such that $|Y| \leq t \land \neg B(\bar{x}, \bar{X}, Y)$.  

67
5. Generalized Polynomial Local Search and Improved Witnessing Theorem

The definition of $Q_A$ proceeds by induction on $k$. For $k = 0$, $A$ is a $\Sigma^B_k$-formula. Therefore, there is an $AC^0$-function $F_A(x, \tilde{X})$ such that $F_A(x, \tilde{X})$ is equal to $\langle 1, \emptyset \rangle$, if $A$ is true, and equal to $\langle 0, \emptyset \rangle$, otherwise. Then let $Q_A$ be an $\text{ITER}$ that represents $F_A$.

Assume that $k > 0$ and consider the case when $A$ is a $\Sigma^B_k$-formula; that is to say, $A$ is of the form $(\exists Y \leq t)B(x, \tilde{X}, Y)$. The case when $A$ is a $\Pi^B_k$-formula is dealt in a similar way. By the induction hypothesis, we have a formalizable $\Sigma^B_k$-$\text{ITER}$ problem $Q_B$ that decides the validity of $B$. First, we define an intermediate $\Sigma^B_{k+1}$-$\text{ITER}$ problem $Q$ (formalizable) so that

$$Q((\langle \tilde{x}, \tilde{X} \rangle, \langle 0, Y \rangle)) = \begin{cases} \langle \langle \tilde{x}, \tilde{X} \rangle, \langle 0, \text{Y} + 1 \rangle \rangle & \text{if } (Q_B(\langle \tilde{x}, \tilde{X} \rangle, Y))^{[0]} = 0 \\ \langle \langle \tilde{x}, \tilde{X} \rangle, \langle 1, Y \rangle \rangle & \text{otherwise} \end{cases}$$

Intuitively, $Q$ will start on the initial input $\langle \langle \tilde{x}, \tilde{X} \rangle, \langle 0, \emptyset \rangle \rangle$. Then, after $2^r$-many pseudo-iterations of $Q$, if $A$ is true, then we obtain $\langle \langle \tilde{x}, \tilde{X} \rangle, \langle 1, Y \rangle \rangle$, where $Y$ is the least string that witnesses $A$; otherwise, we get $\langle \langle \tilde{x}, \tilde{X} \rangle, \langle 0, \text{POW}(2r) \rangle \rangle$. Next, we define the side property $H$ so that

$$H(I, \langle \langle \tilde{x}, \tilde{X} \rangle, \langle 0, 0 \rangle \rangle, \langle \langle \tilde{x}, \tilde{X} \rangle, \langle 1, I \rangle \rangle) \leftrightarrow I < J \wedge B(I) \wedge (\forall Y < I) \neg B(Y)$$

and, in all other cases, $H(I, X, Y)$ is false. Before, we carry on, it is important to note that $H(I, X, Y)$ can easily be rewritten as a $\Pi^B_k$-formula (so can $(\forall Y' < I) \neg B(Y')$). Therefore, $H$ is a $\Pi^B_k$-property. Consequently, we define $R$ to be

$$\text{PsIter}[Q, H](\text{POW}(2r), \langle \langle \tilde{x}, \tilde{X} \rangle, \langle 0, 0 \rangle \rangle).$$

Finally, $Q_A$ is defined to be $\langle R(\tilde{x}, \tilde{X}) \rangle^{[1]}$.

Clearly, $Q_A$ is a $\Sigma^B_{k+1}$-$\text{ITER}$ problem that decides the validity of $A$ and such that if $A$ is true, then $Q_A$ finds the least $Y$ that witnesses $A$. Furthermore, $Q_A$ is formalizable, since conditions $(t_1)$, $(t_2)$ and $(t_3)$ (and the above definition of $H$) are provable in $\emptyset\emptyset$.

5.4 Proof of the New-style Witnessing Theorem for $\text{TV}^k$

This section is devoted to the proof of Theorem 5.10, which relies on a new-style witnessing lemma. But first, let us state the simple form of Theorem 5.10, which implies the more general form:

**Lemma 5.17** Let $k \geq 0$ and $0 \leq g \leq k$. Suppose that $\varphi(\tilde{x}, \tilde{X}, Y)$ is a $\Pi^B_k$-formula and

$$\text{TV}^{k+1} \vdash \forall x \forall \tilde{X} \exists Y \varphi(\tilde{x}, \tilde{X}, Y).$$

Then there is a $\Sigma^B_{k+1}$-$\text{ITER}$-problem $Q$ with $\Pi^B_g$-goal $G$ and an $AC^0$-function $H$ such that $Q$ is formalizable and

$$\emptyset\emptyset \vdash G(\tilde{x}, \tilde{X}, Z) \supset \varphi(\tilde{x}, \tilde{X}, H(\tilde{x}, \tilde{X}, Z)).$$
5.4. Proof of the New-style Witnessing Theorem for $\mathsf{TV}^k$

Proof Sketch of Theorem 5.10 from Lemma 5.17. Assume that $\mathsf{TV}^{k+1}$ proves $\forall \vec{x} \forall \vec{Y} \exists Y \psi(\vec{x}, \vec{X}, Y)$, where $\psi \in \Sigma_{k+1}^B$. By Theorem 5.14, we can assume that $\psi \in \Sigma_{k+1}^B$ and $\psi$ is of the form $\exists Y \psi(\vec{x}, \vec{X}, Y)$, where $\psi \in \Pi_k^B$. By the equivalence of $\mathsf{TV}^{k+1}$ and $\mathsf{TV}^{k+1}$ (see Corollary 5.16),

$$\mathsf{TV}^{k+1} \vdash \forall \vec{x} \forall \vec{X} \exists Y \phi(\vec{x}, \vec{X}, Y).$$

From here, the proof steps are very similar to the proof of Theorem 4.18 from Lemma 4.19. Here, we apply Lemma 5.17, instead of Lemma 4.19, for the case of Theorem 4.18. □

In what follows, we will follow the notations set in [BB09] to ease up the overloading of symbols:

Let $\phi(\vec{a}, \vec{\alpha})$ be a $\Sigma_{k+1}^B$-formula of the form $(\exists Y \leq t) \phi'(Y)$. Then $\mathsf{Wit}_\phi(\vec{a}, \vec{\alpha}, \beta)$ denotes the formula $|\beta| \leq t \land \phi'(\beta)$. If $\phi$ is a $\Pi_k^B$-formula, then $\mathsf{Wit}_\phi(\vec{a}, \vec{\alpha}, \beta)$ is $\phi$ itself.

Next, suppose that $\Gamma(\vec{a}, \vec{\alpha}) = \phi_0, \ldots, \phi_{m-1}$ is a subsequence of the antecedent of a sequent. Then

$$\mathsf{Wit}_\Gamma(\vec{a}, \vec{\alpha}, \beta) \equiv \bigwedge_{i=0}^{m-1} \mathsf{Wit}_\phi(\vec{a}, \vec{\alpha}, \beta_i).$$

Now, assume that $\Delta(\vec{a}, \vec{\alpha}) = \phi_0, \ldots, \phi_{p-1}$ is a subsequence of the succedent of a sequent. Then $\mathsf{Wit}_\Delta(\vec{a}, \vec{\alpha}, \gamma)$ states that $\gamma$ is a pair $(i, Z)$, where $i$ points to $\phi_i$ and $Z$ witnesses $\phi_i$. More specifically,

$$\mathsf{Wit}_\Delta(\vec{a}, \vec{\alpha}, \gamma) \equiv \bigvee_{i=0}^{p-1} (i = \gamma[0] \land \mathsf{Wit}_\phi(\vec{a}, \vec{\alpha}, \gamma[1])).$$

We are now ready to formulate the new-style witnessing lemma for $\mathsf{TV}^{k+1}$, which is stated in terms of $\mathsf{TV}^{k+1}$:

**Lemma 5.18** (New-style Witnessing Lemma for $\mathsf{TV}^{k+1}$) Suppose that $\mathsf{TV}^{k+1}$ proves a sequent $\Gamma(\vec{a}, \vec{A}) \rightarrow \Delta(\vec{a}, \vec{A})$ consisting only of $\Sigma_{k+1}^B$-formulæ. Then there is a formalizable $\Sigma_{k+1}^B$-ITER problem $Q$ such that $\mathsf{TV}^{k+1}$ proves

$$\mathsf{Wit}_\Gamma(\vec{a}, \vec{A}, \vec{B}) \land Y = Q(\vec{a}, \vec{A}, \vec{B}) \supset \mathsf{Wit}_\Delta(\vec{a}, \vec{A}, Y).$$

(5.2)

Originally, Lemma 5.18 was stated in terms of a formalizable $\Pi_k^B$-PLS problem (see [BB09, Lemma 5]). Unfortunately, the proof of Lemma 5 [BB09] contains an error: the case of the induction inference. Basically, the tool that Beckmann and Buss [BB09] used to deal with the case of the induction inference was pseudo-iteration. The problem is that the side property $H$ is of $\Sigma_{k+1}^B$ complexity, but not of $\Pi_k^B$ complexity. Therefore, one cannot get a $\Pi_k^B$-PLS problem using $H$. For this reason, we redefined Beckmann and Buss’s $\Pi_k^B$-PLS so that the predicate defining the set of feasible solutions is of $\Sigma_{k+1}^B$ complexity.

**Proof of Lemma 5.17 from Lemma 5.18.** Let $\phi^*(\vec{x}, \vec{X}, Y)$ be a $\Pi_k^B$ obtained from $\phi$ by simply adding vacuous quantifiers. Hence, $\mathsf{TV}^{k+1}$ proves the sequent

$$\rightarrow (\exists Y \leq s) \phi^*(\vec{a}, \vec{A}, Y),$$

(5.3)
5. Generalized Polynomial Local Search and Improved Witnessing Theorem

where \( s = s(\bar{a}, \bar{A}) \) is a term suggested by Parikh’s theorem (Theorem 2.26). By Lemma 5.18, let \( Q = (C,F,t) \) be a formalizable \( \Sigma^B_{k+1} \)-ITER problem such that \( \overline{V}^0 \) proves

\[
W = Q(\bar{a}, \bar{A}) \supset \text{Wit}(\exists Y \leq s) \varphi^*(\bar{a}, \bar{A}, W).
\]

Here, \( \text{Wit}(\exists Y \leq s) \varphi^*(\bar{a}, \bar{A}, W) \) means that \( W = \langle 0, W_1 \rangle \) such that \( |W_1| \leq s \land \varphi^*(\bar{a}, \bar{A}, W_1) \). We next define a formalizable \( \Sigma^B_{k+1} \)-ITER problem \( Q' \), using \( Q \). The idea behind \( Q' \) is that on any candidate solution to \( Q \), \( Q' \) behaves just like \( Q \). But once a solution \( Z \) to \( Q(\bar{a}, \bar{A}) \) is found, then \( Q' \) concatenates 1 to \( Z \) (that is to say, \( F'(\bar{a}, \bar{A}, Z) = Z \ast \bar{s}, 1 \)) and then behaves as the identity (that is to say, \( F'(\bar{a}, \bar{A}, Z) = Z \)). Note that, by doing this, the length of a solution to \( Q' \) is strictly bigger than the length of any candidate solution to \( Q \). Formally, we define \( C'(\bar{a}, \bar{A}, Z) \) to hold if, and only if,

\[
C(\bar{a}, \bar{A}, Z) \lor |Z| \leq t + 1 \land \varphi^*(\bar{a}, \bar{A}, ((\bar{a}, \bar{A}, Z)_1)^{\{1\}})
\]

holds. Then we define \( F' \) so that, for all \( Z \) of length less than or equal to \( t \),

\[
F'(\bar{a}, \bar{A}, Z) = \begin{cases} 
F(\bar{a}, \bar{A}, Z) & \text{if } Z < F(\bar{a}, \bar{A}, Z) \\
Z \ast \bar{s}, 1 & \text{otherwise,}
\end{cases}
\]

In all other cases, \( F'(\bar{a}, \bar{A}, Z) = Z \). Next, we define \( t' \) to be \( t + 1 \) and let \( P = Q' = (C', F', t') \).

Finally, we define the goal predicate \( G(\bar{a}, \bar{A}, Z) \) to hold if, and only if, the following formula holds:

\[
|Z| = t + 1 \land \varphi(\bar{a}, \bar{A}, ((\bar{a}, \bar{A}, Z)_1)^{\{1\}})
\]

and \( H(\bar{a}, \bar{A}, Z) \) is defined to be \(((\bar{a}, \bar{A}, Z)_1)^{\{1\}}\).

It is straightforward to verify that \( P \) is a formalizable \( \Sigma^B_{k+1} \)-ITER problem with \( \Pi^B_S \)-goal such that \( \overline{V}^0 \) proves that if \( G(\bar{x}, \bar{X}, Z) \) is true, then \( \varphi(\bar{x}, \bar{X}, H(\bar{x}, \bar{X}, Z)) \) holds. \( \square \)

The sequent calculus \( \text{LK-}\overline{TV}^{k+1} \) for \( \overline{TV}^{k+1} \) is essentially \( \text{LK-}V^0 \) (cf. Definition 2.8) augmented with the \( \Sigma^B_{k+1} \)-\text{SIND rule}, which is

\[
\frac{\psi(B), \Gamma \rightarrow \Delta, \psi(S(B))}{\psi(\emptyset), \Gamma \rightarrow \Delta, \psi(T)}
\]

where \( B \) is an eigenvariable and cannot appear in the lower sequent (as a reminder, \( S(B) \) is the binary successor function whose bit defining axiom is (2.30)).

The sequent calculus \( \text{LK-}\overline{TV}^{k+1} \) satisfies the following property, whose proof can be found in [CN10]:

**Theorem 5.19** Suppose that \( \overline{TV}^{k+1} \) proves a sequent \( \Gamma \rightarrow \Delta \) consisting only of \( \Sigma^B_{k+1} \)-formulae. Then there is an \( \text{LK-}\overline{TV}^{k+1} \) proof \( \pi \) of \( \Gamma \rightarrow \Delta \) such that every formula in \( \pi \) is a \( \Sigma^B_{k+1} \)-formula.
5.4. Proof of the New-style Witnessing Theorem for $TV^k$

Proof of Lemma 5.18. This Lemma is proven in the same way as Lemma 4.19. Hence, by Theorem 5.19, let $\pi$ be an LK-$TV^{k+1}$ proof of $\Gamma \to \Delta$ such that every formula in $\pi$ is a $\hat{\Sigma}_{k+1}$-formula. Again, we show that $\Gamma^0$ proves (5.2) by induction on the depth of a sequent $S$ in $\pi$.

The inductive proof splits into cases, depending on whether $S$ is an initial sequent or generated by the use of an inference rule. Here, we will only deal with the case of the $\hat{\Sigma}_{k+1}$-SIND rule, since the cut rule is treated in a similar way as the case of the $\hat{\Sigma}_{k+1}$-SIND rule and the other cases are either straightforward or almost similar to the cases in the proof of Lemma 4.19.

Suppose that $S(a, A)$ is obtained by the application of the $\hat{\Sigma}_{k+1}$-SIND rule. Then $S$ is the bottom sequent of

\[
\frac{\psi(B), \Gamma \to \Delta, \psi(S(B))}{\psi(\emptyset), \Gamma \to \Delta, \psi(T)}
\]

Consider the case when $\psi$ is a $\hat{\Sigma}_{k+1}$-formula. By the induction hypothesis, let $Q(a, A, B, \beta)$ be a formalizable $\Sigma_{k+1}$-ITER problem such that $\Gamma^0$ proves

\[
\text{Wit}_{\psi(B), \Gamma}(a, A, B, \beta) \wedge Q(a, A, B, \beta) = Y \supset \text{Wit}_{\Delta, \psi(S(B))}(a, A, B, Y)
\]

Let $t = (w(T))_2$ (see Definition 2.9). Our goal is to define a formalizable $\Sigma_{k+1}$-ITER problem $P$ such that $\Gamma^0$ proves

\[
\text{Wit}_{\psi(\emptyset), \Gamma}(a, A, \beta, \beta) \wedge P(a, A, \beta, \beta) = Y \supset \text{Wit}_{\Delta, \psi(T)}(a, A, Y).
\]

Let $\Delta = \psi_0, \ldots, \psi_{p-1}$ (that is to say, $p$ is the number of formulae in $\Delta$). The idea is essentially the same as in the induction case in the proof of Lemma 5 in [BB09]. That is to say, we will use a variant $P_1$ of $Q$ and iterate $P_1$ for $t$-many times on an initial input $(\beta, A, \emptyset, Y, \beta)$, where $\emptyset$ will signal that we are at the beginning of the iteration, $Y$ is intended to be a pair $(p, \beta)$ such that $\text{Wit}_{\psi(\emptyset)}(a, A, \beta)$ holds and $\beta$ is intended to be a sequence of strings such that $\text{Wit}_{\Gamma}(a, A, \beta)$ holds. At the end of the $t$-many iterations of $P_1$ on $(\beta, A, \emptyset, Y, \beta)$, we are guaranteed to get a witness to either a formula in $\Delta$ or $\psi(T)$. Formally, we define $P_1$ so that

\[
P_1((\beta, A, B, Y, \beta)) = \begin{cases} 
(\beta, A, S(B), Y, \beta) & \text{if } Y^{[0]} < p \\
(\beta, A, S(B), Q(a, A, B, Y^{[1]}), \beta) & \text{otherwise},
\end{cases}
\]

Next, the side property $H$ is defined just as in [BB09]; that is to say, $H(J, X, Z)$ is true if, and only if, $X = (\beta, A, \emptyset, Y, \beta)$ and $Z = (\beta, A, B, W, \beta)$ and the following formula is true:

\[
\text{Wit}_1(\beta, A, \beta) \wedge \text{Wit}_{\Delta, \psi(\emptyset)}(a, A, Y) \supset \text{Wit}_1(\beta, A, \beta) \wedge \text{Wit}_{\Delta, \psi(B)}(a, A, B, W) \wedge B = J,
\]

where the formula $\text{Wit}_{\Delta, \psi(B)}(a, A, B, W)$ means that either $W^{[0]} < p \wedge \text{Wit}_1(a, A, W)$ holds or $W^{[0]} = p \wedge \text{Wit}_{\psi(B)}(a, A, B, W^{[1]})$ holds. Then the formula $\text{Wit}_{\Delta, \psi(\emptyset)}(a, A, Y)$ is defined to be $\text{Wit}_{\Delta, \psi(B)}(a, A, \emptyset, Y)$. 

71
Finally, we define the $\Sigma_{k+1}^B$-ITER problem $P$ for the lower sequent as follows:

$$P(\bar{a}, \bar{A}, \beta, \bar{\beta}) = (PsIter|\Pi|H)(T, (\bar{a}, \bar{A}, \emptyset, Y, \bar{\beta}))^{[l+1]},$$

where $Y = (p, \beta)$ and $l$ is the number of variables in $\bar{a}, \bar{A}$. Thus, assuming that $\beta$ satisfies $\text{Wit}_{\psi(\emptyset)}(\bar{a}, \bar{A}, \beta)$ and the sequence $\bar{\beta}$ satisfies $\text{Wit}_{\Delta}(\bar{a}, \bar{A}, \bar{\beta})$, it follows that $(\ldots)^{[l+1]}$ extracts the string $W$ which satisfies $\text{Wit}_{\Delta, \psi(T)}(\bar{a}, \bar{A}, W)$.

It is straightforward to check that $P$ is formalizable, since the conditions $(t_2)$ and $(t_3)$ for $H$ (see the paragraph on the constructions of pseudo-iteration of formalized $\Sigma_{k+1}^B$-ITER problems in Section 5.3) are easily provable in $V^0$. Furthermore, it is easy to check that $P$ is a correct formalizable $\Sigma_{k+1}^B$-ITER for the lower sequent. 

\[5.5\text{ Discussion on Skolemized Iteration Problems}\]

In the previous sections, we argued that $[\Sigma_{k+1}^B$-ITER problems with $\Pi^B$-goals]$_X$ characterizes the class of $\Sigma_{g+1}^B$-definable search problems in $TV^{k+1}$, where the reduction is provable in $V^0$. In this section, we show how to obtain a stronger characterization of the $\Sigma_{g+1}^B$-definable search problems in $TV^{k+1}$ in terms of Skolemizable $\Sigma_{k+1}^B$-ITER problems with $\Pi^B$-goals (see [BB09], for a full exposition of how to Skolemize $\Sigma_{k+1}^B$-ITER problems with $\Pi^B$-goals). Then we use the definition of Skolemized $\Sigma_{k+1}^B$-ITER problems with $\Pi^B$-goals in order to derive a class $\text{SIter}(k)$ of $\forall \Sigma^B_1$-principles, which arises in a uniform way: one generic formula $\text{SIter}_k(\mathcal{P})$ is used in order to define $\text{SIter}(k)$, where $\mathcal{P}$ is a sequence of relation and function symbols. Finally, we show that $\text{SIter}(k)$ is $AC^0$-many-one complete for the $\forall \Sigma^B_1$-theorems of $TV^{k+1}$, where the reduction is provable over $V^0$.

\textbf{Skolemized iteration problems and the $\forall \Sigma^B_{g+1}$-consequences of $TV^{k+1}$}. Suppose that $\bar{V}^0$ proves a formula $\varphi$ of the form

$$\forall \bar{x} \forall \bar{X} (Q_1Y_1 \leq s_1)(Q_2Y_2 \leq s_2) \ldots (Q_lY_l \leq s_l) \psi(\bar{x}, \bar{X}, \bar{Y}),$$

where, without loss of generality, we assume that the terms $s_j$ do not involve any of the variables $Y_j$, the formula $\psi$ is a $\Sigma^B_2$-formula and $l \geq 0$. Here, the quantifiers in $\varphi$ alternate between $\exists$ and $\forall$. For the sake of illustration, assume that $Q_i = \exists$, when $i$ is odd, and $Q_i = \forall$, when $i$ is even. Hence, $\varphi$ is of the form

$$\forall \bar{x} \forall \bar{X} (\exists Y_1 \leq s_1)(\forall Y_2 \leq s_2) \ldots (\forall Y_l \leq s_l) \psi(\bar{x}, \bar{X}, \bar{Y}). \tag{5.3}$$

Then, in some cases, $\bar{V}^0$ proves a Skolemized form of (5.3). More specifically, there are $L_{\forall \bar{y}}$-terms $T_i = T_i(\bar{x}, \bar{X}, Y_2, Y_4, \ldots, Y_{l-1})$, for $i$ odd, such that $\bar{V}^0$ proves

$$\forall \bar{x} \forall \bar{X} \forall Y_2 \leq s_2(\forall Y_4 \leq s_4) \ldots (\forall Y_{l-1} \leq s_{l-1}) \big[ T_1 \leq s_1 \land T_2 \leq s_3 \land \ldots \land T_l \leq s_l \land \psi(\bar{x}, \bar{X}, T_2, Y_2, T_3, \ldots, Y_{l-1}, T_l) \big] \tag{5.4}$$
5.5. Discussion on Skolemized Iteration Problems

Now, note that the Skolemized form (5.4) is stronger in the sense that it logically implies (5.3) (this fact will be used later, therefore, it is worth remembering). Before we go any further, first consider the following definition ([BB09] describes a step-by-step process of how to Skolemize conditions (a)-(e) from Definition 5.3 and 5.6):

**Definition 5.20** Let \( Q = (C, F, t, G) \) be a \( \Sigma_{k+1}^B \)-ITER problem with \( \Pi_g^B \)-goal \( G \). Then \( Q \) is said to be *formalizable in Skolem form* in \( \mathcal{V}^0 \) (or just *Skolemizable*) if the following conditions are satisfied:

1. The function \( F \) is given by an \( \mathcal{L}_{\mathcal{V}^0} \)-term,
2. The predicates \( C \) and \( G \) are given by a \( \hat{\Sigma}_{k+1}^B \)-formula and \( \hat{\Pi}_g^B \)-formula, respectively,
3. Skolemized versions of (a)-(e) (from Definition 5.3 and 5.6) are provable in \( \mathcal{V}^0 \), where the Skolem functions are given by \( \mathcal{L}_{\mathcal{V}^0} \)-terms.

The class of Skolemizable \( \Sigma_{k+1}^B \)-ITER problems with \( \Pi_g^B \)-goals will sometimes be denoted by \([\Sigma_{k+1}^B \text{-ITER problems with } \Pi_g^B \text{-goals}]_{\text{Sk}}\).

The following theorem easily follows from Theorem 5.8, which states that \( TV^{k+1} \) proves the totality of formalizable \( \Sigma_{k+1}^B \)-ITER problems with \( \Pi_g^B \)-goals, and the fact that Skolemized versions of (a)-(e) are stronger than (a)-(e):

**Theorem 5.21** Let \( Q \) be a search problem in \([\Sigma_{k+1}^B \text{-ITER problems with } \Pi_g^B \text{-goals}]_{\text{Sk}}\). Then \( TV^{k+1} \) proves the totality of \( Q \).

The following theorem is the converse of the previous theorem:

**Theorem 5.22** (Skolemized New-style Witnessing Theorem for \( TV^{k+1} \)) Let \( k \geq 0 \) and \( 0 \leq g \leq k \). Suppose that \( \phi(\bar{x}, \bar{X}, Y) \) is a \( \Sigma_{g+1}^B \)-formula and

\[
TV^{k+1} \vdash \forall \bar{x} \forall \bar{X} \exists Y \phi(\bar{x}, \bar{X}, Y).
\]

Then there is a \( \Sigma_{k+1}^B \)-ITER problem \( Q \) with \( \Pi_g^B \)-goal \( G \) and an \( \mathcal{L}_{\mathcal{V}^0} \)-term \( H \) such that \( Q \) is formalizable in Skolem form and \( \mathcal{V}^0 \) proves a Skolem version of

\[
G(\bar{x}, \bar{X}, Z) \supset \phi(\bar{x}, \bar{X}, H(\bar{x}, \bar{X}, Z)), \quad (5.5)
\]

where the Skolem functions are given by \( \mathcal{L}_{\mathcal{V}^0} \)-terms.

Beckmann and Buss [BB09] originally proved a version of Theorem 5.22 over the base theory \( V^1 \) instead of \( \mathcal{V}^0 \) (see [BB09, Theorem 3]). The arguments used in the proof of Theorem 3 in [BB09] can easily be adapted to prove Theorem 5.22. This is because the constructions we showed in Section 5.3 are similar to the ones in [BB09] (one could call \( \Sigma_{k+1}^B \)-ITER problems just \( \Sigma_{k+1}^B \)-PLS problems with no cost function (the cost of a candidate solution is just its length)). Therefore, the constructions in Section 5.3 are Skolemizable. Furthermore, \( \mathcal{V}^0 \) proves
5. Generalized Polynomial Local Search and Improved Witnessing Theorem

a Skolemized version of (5.1). By analyzing the constructions of $\Pi^b_k$-PLS in [BB09] and the proof of Theorem 3 in [BB09], one can also remark that the Skolem functions generated have very low computational complexity. In fact, they are all AC^0-functions. Therefore, they can be expressed by $Z_q^\varphi$-terms.

By combining Theorem 5.21 and Theorem 5.22 and the fact that $V^0$ is a universal conservative extension of $V^0$, we obtain the following corollary:

**Corollary 5.23** Let $0 \leq g \leq k$. Then $[\Sigma^B_{k+1}-ITER with \Pi^B_k$-goals] is $AC^0$-many-one-complete for the set of all $\Sigma^B_{k+1}$-definable search problems in $TV^{k+1}$, where the reduction is provable in the theory $V^0$.

**Silter($k$) is complete for the $\forall \Sigma^B_1$-theorems of $TV^{k+1}$**. Consider Definition 5.20 of a Skolemizable $\Sigma^B_{k+1}$-ITER problem $P = (C,F,d,G)$ with $\Pi^B_0$-goal $G$ and let us give a reminder of what the conditions (a)-(e) state again (for the sake of simplicity, assume that there is only one input variable $X$ instead of $x, \bar{x}$):

(a) $\alpha \equiv C(X,Z) \supset |Z| \leq d(X)$,
(b) $\beta \equiv C(X,\emptyset)$,
(c) $\gamma \equiv C(X,Z) \supset C(X,F(X,Z))$,
(d) $\delta \equiv G(X,Z) \supset C(X,Z) \land F(X,Z) \leq Z$,
(e) $\epsilon \equiv C(X,Z) \land F(X,Z) \leq Z \supset G(X,Z)$.

Our goal is to describe a relativized $\forall \Sigma_1^B$-principle by using the above definition as a template. Without loss of generality, we can always assume that $C(X,Z)$ is a $\Sigma^B_{k+1}$-formula. Hence, $C$ is of the form

$$(\exists Y_1 \leq t_1)(\forall Y_2 \leq t_2) \ldots (QY_{k+1} \leq t_{k+1})C_0(X,Z,\bar{Y}),$$

where, without loss of generality, we can also assume that the terms $t_j$ do not involve any of the variables $Y_i$; where $Q$ is either $\exists$ or $\forall$, depending on whether $k$ is odd or even. Now, consider the Skolemization $\alpha_{Sk} \gamma_{Sk}$ of $\alpha \epsilon$, as shown in [BB09]. For instance, a Skolemization of $\gamma$ is obtained by first turning $\gamma$ into a suitable prenex form as given by (5) in [BB09]. Then a Skolemization of $\gamma$ uses functional substitutions for the existentially quantified variables (see (5.4), for example). Then let $Cond_k(d,\bar{t},C_0,G,F,T_\alpha,T_\beta,\ldots,T_\epsilon,X,Z)$ be the formula

$$\alpha_{Sk}(d,\bar{t},C_0,T_\alpha,X,Z) \land \beta_{Sk}(\bar{t},C_0,T_\beta,X,Z) \land \ldots \land \epsilon_{Sk}(\bar{t},C_0,G,F,T_\epsilon,X,Z).$$

Here, the symbols $T_\alpha,\ldots,T_\epsilon$ are sequences (whose lengths depend on $k$) of new function symbols added to the language and are understood to be the functions used for Skolemizing the formulae $\alpha \epsilon$; also, $d,\bar{t},C_0,G$ and $F$ are now new symbols adjoined to the language. Note that

$$Cond_k(d,\bar{t},C_0,G,F,T_\alpha,T_\beta,\ldots,T_\epsilon,X,Z)$$
5.5. Discussion on Skolemized Iteration Problems

is a $\Sigma_0^B(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon})$-formula. We define the formula

$$\text{Silter}_k(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon})$$

to be

$$\forall X[(\forall Z \leq d)\text{Cond}_k(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon},X,Z) \supset (\exists Y \leq d)G(X,Y)].$$

Observe that the formula $\text{Silter}_k$ is equivalent to a $\forall\Sigma_1^B(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon})$-formula, provable in $TV^{k+1}$ (relativized).

**Definition 5.24** We define $\text{Silter}(k)$ to be the class of all formulae obtained from

$$\text{Silter}_k(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon})$$

by replacing $d,\bar{t}$ with $\mathcal{L}_A^2$-terms, $C_0, G$ with $\Sigma_0^B$-formulae and $F, T_{\alpha}, \ldots, T_{\varepsilon}$ with $\mathcal{L}_{\varphi}$-terms.

**Corollary 5.25** $TV^{k+1}(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon}) \vdash \text{Silter}_k(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon}).$

The above corollary follows directly from the relativized version of Theorem 5.8, which states that $TV^{k+1}$ proves the totality of formalizable $\Sigma_{k+1}^B$-ITER problems with $\Pi_0^B$-goals, and the fact that $\forall\alpha \cdot \forall\varepsilon$ logically follow from $\forall\alpha_{\text{Sk}} \cdot \forall\varepsilon_{\text{Sk}}$, as discussed earlier in this section.

The following theorem is a converse of Corollary 5.25:

**Theorem 5.26** The set of all $\forall\Sigma_1^B$-theorems of $TV^{k+1}$ is $A^0\text{-many-one reducible to } \text{Silter}(k)$, provable in $\neg\Sigma_0^0$.

**Proof Sketch.** Suppose that $TV^{k+1}$ proves $\forall X \exists Y \phi(X,Y)$, where $\phi$ is assumed to be a $\Sigma_0^B$-formula, without loss of generality. Then, by the Skolemized new-style witnessing theorem for $TV^{k+1}$, we obtain a Skolemizable $\Sigma_{k+1}^B$-ITER problem $Q$ with $\Pi_0^B$-goal $G$ whose solution can then be used in order to obtain a witness to the outermost existential quantifier in $\exists Y \phi(X,Y)$ (see Theorem 5.22 for more details). Since $Q$ is Skolemizable, it follows that conditions $\alpha \cdot \varepsilon$ are Skolemizable; that is to say, $\neg\Sigma_0^0$ proves

$$\forall\alpha_{\text{Sk}}(d,\bar{t},C_0,T_{\alpha},X,Z) \cdot \forall\varepsilon_{\text{Sk}}(\bar{t},C_0,G,F,T_{\varepsilon},X,Z),$$

where $d,\bar{t}$ are $\mathcal{L}_A^2$-terms, $C_0$ is $\Sigma_0^B$-formulae and $F, T_{\alpha}, \ldots, T_{\varepsilon}$ are $\mathcal{L}_{\varphi}$-terms. Then we use $\text{Silter}_k(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon})$ in order to obtain the desirable reduction, where

$$\text{Silter}_k(d,\bar{t},C_0,G,F,T_{\alpha},\ldots,T_{\varepsilon})$$

can be rewritten as follows:

$$\forall X \exists Z[\neg(Z \leq d(X) \supset \text{Cond}_k(d,\bar{t},C_0,G,F,T_{\alpha},T_{\beta},\ldots,T_{\varepsilon},X,Z)) \lor (Z \leq d(X) \land G(X,Z))]$$

Finally, combining Corollary 5.25, Theorem 5.26 and the fact that $\neg\Sigma_0^0$ is a universal conservative extension of $V^0$, we obtain the following theorem:
5. Generalized Polynomial Local Search and Improved Witnessing Theorem

Theorem 5.27 SiIter\((k)\) is $AC^0$-many-one complete for the set of all $\forall \Sigma^B_1$-theorems of $TV^{k+1}$, where the reduction is provable over $V^0$.

The following conjecture is justified by Theorem 5.27:

Conjecture 5.28 $TV^k(d, \vec{t}, C_0, G, F, T_\alpha, \ldots, T_\epsilon) \not\vdash SiIter_k(d, \vec{t}, C_0, G, F, T_\alpha, \ldots, T_\epsilon)$. 
Kołodziejczyk et al. [KNT11] introduced the linear local improvement principle, which is about labelings of a directed acyclic graph $G$ on vertices $\{0, 1, \ldots, a-1\}$ and directed edges $(x-1, x)$, for $1 \leq x \leq a-1$. Initially, every node in $G$ has a label with score 0. Then sweeping forward through $G$ (from vertex 0 to vertex $a-1$) allows scores to increase from even to odd, and sweeping backwards (from vertex $a-1$ to vertex 0) allows scores to increase from odd to even. Thus, under these conditions, scores can increase without bound. There are different types of linear local improvement principle. In this chapter, we are mainly interested in $\text{LLI}$ and $\text{LLI}_{\log}$, where we are only allowed to improve the score up to a score bound $c$ and $\log c$, respectively. Kołodziejczyk et al. [KNT11] showed that $\text{LLI}_{\log}$ is $P$-many-one complete for the $\forall \Sigma^b_1$-theorems of $U^1_2$, where the reduction is provable in $S^1_2$. Their results were later improved by Beckmann and Buss [BB14], who showed that $U^2_2$ already proves $\text{LLI}$.

In this chapter, we introduce a new class of TFNP search problems that we call extended linear local improvement principle, which can be viewed as an extension of the linear local improvement principle in the sense that a vertex $x$ in $G$ may now have polynomially-many edges coming into it, instead of just the one from $x-1$. Similar to the linear local improvement principle, there are different types of extended linear local improvement principle: $\text{ELLI}$, $\text{ELLI}_{\log}$, and $\text{ELLI}_k$, where $k$ is a constant – these principles are defined analogously to $\text{LLI}$ and $\text{LLI}_{\log}$ by putting bounds on scores. For some constant $m$, which depends on the constant number of witness queries made by a polynomial-time Turing machine to some oracle, we show that, for all $k \geq m$, $\text{ELLI}_k$ is $P$-many-one complete for the $\forall \Sigma^b_1$-theorems of $U^1_2$, where the reduction is
6. Extended Linear Local Improvement Principles and $U^1_2$

provable in $S^1_2$. Furthermore, we show that, for all $d \geq 1$ and all $k \geq m$, $\text{ELLI}_{d}^{\log^d}$ and $\text{ELLI}_{k}$ are $P$-equivalent and this equivalence is provable in $S^1_2$. Additionally, we define a new theory that we call WT, which is weaker than $S^1_2$. Using WT as a base theory, we obtain an improved new-style witnessing theorem for $U^1_2$ (our base theory WT is weaker than $S^1_2$, the base theory used by Beckmann and Buss [BB14] for their new-style witnessing theorem for $U^1_2$).

This chapter is organized as follows. We first start with some preliminaries, since we now switch from the two-sorted context of [CN10] to the one-sorted (and two-sorted) context of Buss [Bus86]. We explain how elements of a complexity class are viewed in Buss’s setting and provide the necessary definitions in order to carry out the arguments in this chapter. In Section 6.2, we define our base theory WT and prove a witnessing theorem for it. Section 6.3 is devoted to the proof of the improved new-style witnessing theorem for $U^1_2$. Finally, Section 6.4 discusses our results that relate the $\forall \Sigma^b_1$-theorems of $U^1_2$ and the extended linear local improvement principle.

6.1 Preliminaries

One-sorted Logic. One-sorted logic is a contraction of two-sorted logic (cf. Section 2.1): terms and formulae are built from an infinite set of variables $x, y, z, \ldots$, which are intended to range over $\mathbb{N}$; from the propositional connectives $\neg, \land, \lor$ and logical constants $\bot$ and $\top$; from the existential quantifier $\exists$ and universal quantifier $\forall$; from function and relation (or predicate) symbols and from parentheses $( )$.

Naturally, function and relation symbols only have one type of arguments and there is only one kind of function symbols. Again, we use $f, g, h, \ldots$ as meta-symbols for function symbols and $P, Q, R, \ldots$ as meta-symbols for relation symbols.

The vocabulary $\mathcal{L}_{S_2}$ of one-sorted bounded arithmetic that we use here is the one from Pollett [Pol97, Pol99] and consists of the non-logical function symbols

$$0, S, +, \cdot, |x|, \lfloor x/2 \rfloor, \#, \text{MSP}, \neg$$

and the non-logical relation symbol $\leq$. These symbols are intended to be applied to non-negative integers (numbers) and take their usual meaning in the standard model, that is to say, $0, S, +, \cdot, \neg$ are the zero constant, the successor function, addition, multiplication and arithmetical subtraction, respectively; $|x|$ denotes the length of the binary representation of $x$; $\lfloor x/2 \rfloor$ denotes the greatest integer less than or equal to $x/2$; $x\#y$ (smash function) is defined to be $2^{\lfloor |x|/\lfloor y \rfloor \rfloor}$; finally, if $x = (b_{n-1} \ldots b_1 b_0)_2$, then $\text{MSP}(x, i)$ (most significant part) is defined to be $(b_{n-1} \ldots b_{i+1} b_i)_2$, where $(b_{n-1} \ldots b_1 b_0)_2$ is defined to be

$$b_{n-1} \cdot 2^{n-1} + \ldots + b_1 \cdot 2^1 + b_0 \cdot 2^0,$$

and $b_{n-1}, \ldots, b_1, b_0 \in \{0, 1\}$. 78
Terms over $L_{\Delta^b_2}$ (or its extension) are defined in the same way as the ones over $L^*_{\Delta^b_2}$ (with obvious modifications) and we denote terms by $r, s, t, \ldots$. Also, notions such as bounded quantifiers and bounded formulae are the same as before, except now there is only one type of bounded quantifiers. For one-sorted BA, there is another form of quantifiers called \textit{sharply bounded quantifiers}, which are of the form $(\forall x \leq |t|)$ or $(\exists x \leq |t|)$, where $t$ is any term not involving $x$.

**Definition 6.1** ([Bus86]) The class $\Sigma^b_k$ and $\Pi^b_k$ are defined as follows:

1. $\Sigma^b_0 = \Pi^b_0 = \Delta^b_0$ is the class of all formulae whose quantifiers are all sharply bounded.

2. For $k > 0$, $\Sigma^b_{k+1}$ is defined to be the smallest class such that $\Pi^b_k \subseteq \Sigma^b_{k+1}$ and closed under $\wedge, \vee, (\forall x \leq |t|), (\exists x \leq |t|)$. Furthermore, if $\phi \in \Sigma^b_{k+1}$, then $\neg \phi \in \Pi^b_{k+1}$.

3. For $k > 0$, $\Pi^b_{k+1}$ is defined to be the smallest class such that $\Sigma^b_k \subseteq \Pi^b_{k+1}$ and closed under $\wedge, \vee, (\exists x \leq |t|), (\forall y \leq |t|)$. Additionally, if $\phi \in \Pi^b_{k+1}$, then $\neg \phi \in \Sigma^b_k$.

**One-sorted complexity classes and search problems.** In the context of one-sorted BA, elements of a complexity class are relations $R(x)$ over $\mathbb{N}$. Here, the input $x$ is presented in binary to the accepting machine. Thus, in this setting, for example, a relation $R(x)$ is in $P$ if, and only if, there is a Turing machine $M$ that decides $R$ in polynomial-time in the length of the input.

Here, the notion of a search problem is the same as the one in the two-sorted context, but with obvious modifications. Thus, a total search problem is a relation $R(x, y)$ such that $(\forall x)(\exists y)R(x, y)$ is true. Furthermore, $R$ is a TFNP problem if $R$ is a polynomial-time relation such that if $R(x, y)$ holds, then $|y|$ is bounded by the length of the binary representation of $x$.

A total search $R$ is provably total in a theory $\mathcal{T}$ if $R$ is $\Sigma^b_1$-definable in $\mathcal{T}$, where the notion of definability in a theory in this context is the same as Definition 2.48, but with obvious modifications.

Also, the notion of many-one reducibility is similar to Definition 2.49 (again, with obvious modifications). However, in this chapter, whenever we talk about reduction, it is always with respect to P-many-one reduction. Therefore, we just drop any mention of P and simply say, for example, that a class of $\forall \Sigma^b_1$-principles is many-one reducible to another.

**Notation 6.2** We denote by PSPACE the class of relations computable in polynomial-space by a Turing machine.

**One-sorted bounded arithmetic $S_2$ and its extension $U_2^1$.** In this chapter, our notion of a theory is the same the one in the two-sorted context: a theory is a set of formulae that is closed under substitution of terms for variables and also closed under logical consequence.

The purpose of this paragraph is to define Buss’s theory $S_2'$, which is the one-sorted version of the theory $V'$, and the two-sorted theory $U_2^1$ corresponding to PSPACE. For the two-sorted theory $U_2^1$, second-order objects are used differently to the previous chapters: in the previous chapters, second-order objects are the main objects and presented to the accepting machines
in binary; whereas here, second-order objects are used to reason about exponentially-large objects, such as the case of PSPACE computations, and are presented to the accepting machines as oracles.

In order to define the theories $S^1_2$ and $U^1_2$, we first need to define $BASIC$, which provides definitions for the non-logical symbols of $L_{S^2_2}$. Our definition of the theory $BASIC$ follows [Pol97] and consists of the following 33 axioms that can be traced back to Buss [Bus86] and Takeuti [Tak95]:

B1. $y \leq x \supset y \leq S(x)$

B2. $x \neq S(x)$

B3. $0 \leq x$

B4. $(x \leq y \land x \neq y) \leftrightarrow S(x) \leq y$

B5. $x \neq 0 \supset 2 \cdot x \neq 0$

B6. $y \leq x \lor x \leq y$

B7. $(x \leq y \land y \leq x) \supset x = y$

B8. $(x \leq y \land y \leq z) \supset x \leq z$

B9. $|0| = 0$

B10. $x \neq 0 \supset \|2 \cdot x| = S(|x|) \land |S(2 \cdot x)| = S(|x|)$

B11. $|1| = 1$

B12. $x \leq y \supset |x| \leq |y|$

B13. $|x\#y| = S(|x| \cdot |y|)$

B14. $0\#y = 1$

B15. $x \neq 0 \supset [1\#(2 \cdot x) = 2(1\#x) \land 1\#(S(2 \cdot x)) = 2(1\#x)]$

B16. $x\#y = y\#x$

B17. $|x| = |y| \supset x\#z = y\#z$

B18. $|x| = |u| + |v| \supset x\#y = (u\#y) \cdot (v\#y)$

B19. $x \leq x + y$

B20. $x + 0 = x$

B21. $x + S(y) = S(x + y)$
6.1. Preliminaries

B22. \((x + y) + z = x + (y + z)\)

B23. \(x + y \leq x + z \iff y \leq z\)

B24. \(x \cdot 0 = 0\)

B25. \(x \cdot S(y) = (x \cdot y) + x\)

B26. \(x \cdot y = y \cdot x\)

B27. \(x \cdot (y + z) = (x \cdot y) + (x \cdot z)\)

B28. \(1 \leq x \supset (x \cdot y \leq x \cdot z \iff y \leq z)\)

B29. \(x \neq 0 \supset |x| = S([|x/2|])\)

B30. \(x = [x/2] \leftrightarrow (2 \cdot x = y \lor S(2 \cdot x) = y)\)

B31. \(\text{MSP}(a, 0) = a\)

B32. \(\text{MSP}(a, i + 1) = [\frac{1}{2} \text{MSP}(a, i)]\)

B33. \(x \cdot -y = z \leftrightarrow (y + z = x \lor (z = 0 \land x \leq y))\)

**Definition 6.3 ([Bus86])** For \(k \geq 0\), the theory \(S^{b}_2\) is axiomatized by the axioms of BASIC and the **length induction scheme** for \(\Sigma^{b}_k\), denoted \(\Sigma^{b}_k\text{-LIND}\), which is

\[
\phi(0) \land \forall x (\phi(x) \supset \phi(x + 1)) \supset \forall x \phi([x])
\]  

(6.1)

where \(\phi(x) \in \Sigma^{b}_k\) and is allowed to have free variables other than \(x\). In general, for a class \(\Phi\) of formulae over a vocabulary that extends \(L_{S^{2}}\), the length induction scheme for \(\Phi\) is denoted \(\Phi\text{-LIND}\) and has the form of (6.1), but with \(\phi \in \Phi\).

The theory \(S^{b}_2\) is then defined as follows:

\[
S^{b}_2 = \bigcup_{k \geq 1} S^{b}_k
\]  

(6.2)

In this dissertation, we are particularly interested in the theory \(S^{0}_2\) and its extension \(S^{0}_2(PV)\), which includes all polynomial-time functions in its vocabulary (see [Bus86]).

Our exposition of the two-sorted BA theory \(U^{1}_2\) follows [Bus86, Chapter 9]. For this purpose, we extend one-sorted logic with second-order variables \(X, Y, Z, \ldots\), which are intended to range over sets of numbers. The predicate symbol \(\in\), which takes a number and a set as arguments, is added to \(L_{S^{2}}\) and is intended to denote set membership. Again, we write \(X(i)\) for \(i \in X\).

For the rest of this chapter, the classes \(\Sigma^{b}_k\) and \(\Pi^{b}_k\) will be allowed to have second-order variables, but no second-order quantifiers. Likewise, let \(\Sigma^{b}_k\) be defined so that second-order variables are allowed to occur in formulae (including the induction axioms), but second-order
quantifiers are not allowed. Usually, the extension of $S^k_2$ with second-order variables is denoted by $S^k_2^+$. However, since there will be no confusion, the superscript + will be omitted. Likewise, we omit the superscript + in $\Sigma^k_2^+$, $\Pi^k_2^+$ and $S^k_2^+$.

**Definition 6.4** ([Bus86]) The classes $\Sigma^1_0$, $\Sigma^1_1$ and $\Pi^1_1$ are defined as follows:

1. The class $\Sigma^1_0$ is defined to be

$$\bigcup_{k \geq 0} \Sigma^k_1$$

2. The class $\Sigma^1_1$ is the smallest class such that $\Sigma^1_0 \subseteq \Sigma^1_1$ and closed under

$$\land, \lor, (\forall x \leq t), (\exists x \leq t), \exists X.$$ 

Furthermore, if $\phi \in \Sigma^1_1$, then $\neg \phi \in \Pi^1_1$.

3. The class $\Pi^1_1$ is the smallest class such that $\Sigma^1_0 \subseteq \Pi^1_1$ and closed under

$$\land, \lor, (\forall x \leq t), (\exists x \leq t), \forall X.$$ 

Furthermore, if $\phi \in \Pi^1_1$, then $\neg \phi \in \Sigma^1_1$.

**Definition 6.5** ([Bus86]) $U^1_2$ consists of $\Sigma_2^1$, the $\Sigma^1_1$-LIND axioms and the comprehension axiom scheme for $\Sigma^1_0$, denoted $\Sigma^1_0$-comp, which is

$$(\forall x)(\forall X)(\forall y \leq t)[y \in Z \leftrightarrow \phi(y, \bar{x}, \bar{X})],$$

(6.3)

where $\phi \in \Sigma^1_0$ and $t$ is a term.

In general, for a class $\Phi$ of formulae over a vocabulary which extends $L_{S^2_2}$, the comprehension axiom scheme for $\Phi$ has the form of (6.3), but with $\phi \in \Phi$ and is denoted by $\Phi$-comp.

**Sequent calculus for $U^1_2$.** For this chapter, our notion of an LK-$\Phi$ proof is the same as in the previous chapters (cf Section 2.1), but with obvious modifications.

The sequent calculus LK-$U^1_2$ for $U^1_2$ consists of the usual rules of inference: the structural rules, the propositional rules, the rules for bounded number quantifiers and the string quantifier rules $\exists$-left and $\exists$-right. Additionally, it has the $\Sigma^1_1$-LIND rule, which is:

$$\frac{\phi(b), \Gamma \rightarrow \Delta, \phi(b + 1)}{\phi(0), \Gamma \rightarrow \Delta, \phi([t])}$$

for every $\phi$ in $\Sigma^1_0$, where $b$ is eigenvariable and does not appear in the lower sequent of the inference. Non-logical axioms in an LK-$U^1_2$ proof are either of the form

$$\rightarrow (\exists Z)(\forall y \leq t)[y \in Z \leftrightarrow \phi(y, \bar{x}, \bar{X})],$$

(6.4)

where $\phi \in \Sigma^1_0$, or of the form $\rightarrow \phi$, where $\phi$ is an instance of one of the BASIC axioms.
In general, for a class \( \Phi \) of formulae, the \( \Phi \)-LIND rule is constructed in the same way as the \( \Sigma^1_{1,b} \)-LIND rule, but with \( \phi \in \Phi \) instead of \( \Sigma^1_{1,b} \).

Let us next define the notion of “strict” \( \Sigma^1_{1,b} \)-formulae, since during the proof of the new-style witnessing theorem for \( U^1_2 \), it is useful to restrict proofs to only contain “strict” \( \Sigma^1_{1,b} \)-formulae.

**Definition 6.6** ([BB14]) Let \( \phi \in \Sigma^1_{1,b} \). Then \( \phi \) is **strict**, denoted \( s \Sigma^1_{1,b} \), if either \( \phi \in \Sigma^0_{1,b} \) or \( \phi \) is of the form \( (\exists X) \phi \), where \( \phi \in \Sigma^1_{1,b} \).

**Notation 6.7** In what follows, we write \( \phi(x, \{z\} Z(x,z)) \) for the formula obtained from \( \phi(x,Y) \) by replacing every occurrence of the subformula \( Y(z) \) in \( \phi(x,Y) \) by \( Z(x,z) \).

Again, we write \( Z(x_1,\ldots,x_k) \) for \( Z(\langle x_1,\ldots,x_k \rangle) \), where \( \langle x_1,\ldots,x_k \rangle \) is \( (2.19) \) and \( \langle x, z \rangle \) is \( (2.18) \), which is twice the Cantor’s pairing function.

Even though \( U^1_2 \) proves that every \( \Sigma^1_{1,b} \)-formula is equivalent to a \( s \Sigma^1_{1,b} \)-formula [Bus86] – by using the replacement axiom scheme for \( \Sigma^1_{1,b} \), which is

\[
(\forall x \leq t)(\exists Y)\phi(x,Y) \supset (\exists Z)(\forall x \leq t)\phi(x,\{z\} Z(x,z)),
\]

where \( \phi \in \Sigma^1_{1,b} \) and \( t \) a term – unfortunately, the proof in \( U^1_2 \) of the replacement axioms for \( \Sigma^1_{1,b} \)-formulae requires length induction on non-strict \( \Sigma^1_{1,b} \)-formulae (see Theorem 16 of Chapter 9 [Bus86]). As a result, with the current version of the sequent calculus \( LK-U^1_2 \) for \( U^1_2 \), it is not guaranteed to have proofs containing only \( s \Sigma^1_{1,b} \)-formulae. For this reason, we need to work with a slightly modified version of \( LK-U^1_2 \) called \( LK-U^1_{2*} \) [BB14].

**Definition 6.8** ([Pol99]) The classes \( s \Sigma^b_{1,k} \) and \( s \Pi^b_{1,k} \) are defined as follows:

1. \( s \Sigma^b_{1,0} = s \Pi^b_{1,0} = \Sigma^b_{1,0} \).

2. For \( k \geq 0 \), a formula \( \phi(x,\bar{x}) \in s \Sigma^b_{1,k+1} \) if \( \phi \) is of the form \( (\exists y \leq t)\psi(y,\bar{x},\bar{X}) \), where \( \psi \in s \Pi^b_{1,k} \).

3. For \( k \geq 0 \), the class \( s \Pi^b_{1,k+1} \) is defined dually to \( s \Sigma^b_{1,k+1} \).

**Definition 6.9** ([Pol99]) Define \( s \Sigma^0_{1,k} = \Sigma^0_{1,k} \). For \( k \geq 0 \), a formula \( \phi \in s \Sigma^b_{1,k+1} \) if either \( \phi \in s \Sigma^b_{1,k} \) or there is a \( j < k+1 \) such that \( \phi \in (s \Sigma^b_{1,j} \cup s \Pi^b_{1,j}) \). Similarly, \( \phi \in s \Pi^b_{1,k+1} \) if either \( \phi \in s \Pi^b_{1,k+1} \) or there is a \( j < k+1 \) such that \( \phi \in (s \Sigma^b_{1,j} \cup s \Pi^b_{1,j}) \).

**Definition 6.10** The theory \( U^1_{2*} \) is defined to be \( U^1_2 \), but with \( s \Sigma^1_{1,b} \)-LIND instead of \( \Sigma^1_{1,b} \)-LIND, and \( s \Pi^b_{1,\text{comp}} \) instead of \( \Sigma^1_{0,b} \)-comp.

We note that our definition of the theory \( U^1_{2*} \) above is slightly different from the original one [BB14], which uses \( \Sigma^1_{1,b} \)-comp instead of \( s \Pi^b_{1,\text{comp}} \). However, using \( s \Pi^b_{1,\text{comp}} \) instead of \( \Sigma^1_{0,b} \)-comp is perfectly fine, since we can always bring \( \Sigma^1_{0,b} \)-comp down to \( s \Pi^b_{1,\text{comp}} \) by using nested comprehension. For the sake of illustration, let \( \phi(y) \) (omitting the parameters \( \bar{x},\bar{X} \)) be
the $\Sigma_{0}^{1, b}$-formula from a $\Sigma_{0}^{1, b}$-comp axiom (see Definition 6.5) and, without loss of generality, assume that $\phi(y)$ is of the form

$$
(\forall y_{0} \leq t)(\exists y_{1} \leq t) \phi_{0}(y, y_{0}, y_{1}),
$$

where $\phi_{0}$ is a quantifier-free formula. Let $\phi_{1}(y, y_{0})$ be $(\forall y_{1} \leq t)\neg \phi_{0}(y, y_{0}, y_{1})$ (that is to say, $\phi_{1} \in s\Pi_{1}^{0}$). By the $s\Pi_{1}^{0}$-COMP axioms, let $Z_{1}$ be such that the following formula holds:

$$
Z_{1}(y_{0}) \leftrightarrow \phi_{1}(y, y_{0}).
$$

Thus, $\phi(y)$ holds if, and only if, the $s\Pi_{1}^{0}$-formula $(\forall y_{0} \leq t)\neg Z_{1}(y_{0})$ holds.

**Theorem 6.11** ([BB14]) $U_{1}^{1, b} = U_{2}^{1, b}$.

**Definition 6.12** ([BB14]) The sequent calculus $LK_{U_{1}^{1, b}}$ is defined to be $LK_{U_{2}^{1, b}}$, but with the $s\Sigma_{1}^{1, b}$-LIND rule instead of the $\Sigma_{1}^{1, b}$-LIND rule; additionally, $LK_{U_{2}^{1, b}}$ is equipped with an additional rule of inference called the $s\Sigma_{1}^{1, b}$-repl-$\forall$ rule, which is

$$
\frac{a \leq t, \Gamma \rightarrow \Delta, (\exists X)\phi(a, X)}{\Gamma \rightarrow \Delta, (\exists Y)(\forall x \leq t)\phi(x, \{z\}Y(x, z))}
$$

where $\phi$ is a $\Sigma_{0}^{1, b}$-formula and $a$ is an eigenvariable and cannot occur in the lower sequent.

**Theorem 6.13** ([BB14]) Suppose that $U_{2}^{1, b}$ proves a sequent $\Gamma \rightarrow \Delta$ consisting only of $s\Sigma_{1}^{1, b}$-formulae. Then there is an $LK_{U_{2}^{1, b}}$ proof $\pi$ of $\Gamma \rightarrow \Delta$ such that every formula in $\pi$ is a $s\Sigma_{1}^{1, b}$-formula.

### 6.2 Witnessing Theorem for WT

In this section, we define a theory that we call $WT$, which will be used later as a base theory for the new-style witnessing theorem for $U_{2}^{1}$. We then prove a witnessing theorem for $WT$.

**Definition 6.14** The theory WT is axiomatized by the axioms of $\Sigma_{0}^{0}(PV)$ and the $\Sigma_{0}^{b}$-first-order comprehension scheme

$$
\exists y \leq t \forall z < |t| [\text{Bit}(y, z) = 1 \leftrightarrow \phi(\vec{x}, \vec{X}, z)],
$$

where $\phi \in \Sigma_{0}^{b}$ and $\text{Bit}(x, i)$ is a polynomial-time function which computes the $i$-th bit of the binary representation of $x$.

The sequent calculus $LK$-WT for WT consists of the structural rules, the propositional rules and the rules for bounded number quantifiers. Additionally, it has the $\Sigma_{0}^{b}$-LIND rule. Non-logical axioms in an $LK$-WT proof are sequents of the form $\rightarrow \phi$, where $\phi$ is either an instance of the $\Sigma_{0}^{b}$-first-order comprehension axiom or an instance of one of the BASIC axioms.
The sequent calculus LK-\textsc{WT} satisfies the following property, whose proof can be found in [Bus98]:

**Theorem 6.15** Suppose that \textsc{WT} proves a sequent $\Gamma \rightarrow \Delta$, which consists only of $\exists \Sigma^b_1$-formulae. Then there is an LK-\textsc{WT} proof $\pi$ of $\Gamma \rightarrow \Delta$ such that $\pi$ only contains $\exists \Sigma^b_1$-formulae.

Since \textsc{WT} is our base theory for the improved new-style witnessing theorem for $U^1_2$ and the weakest known base theory in the literature for $U^1_2$ is $S^1_2$ [BB14], it is therefore reasonable to compare \textsc{WT} with $S^1_2$. Clearly, \textsc{WT} is contained in $S^1_2$. Now, let us first consider the case when second-sort variables are stripped away from \textsc{WT} and $S^1_2$ and denote the resulting theories $\textsc{WT}^-$ and $S^1_2^-$, respectively. It is easy to see that $\textsc{WT}^-$ and $S^1_2^-$ have the same $\forall \Sigma^b_1$-theorems, since $\textsc{WT}^-$ includes all polynomial-time functions in its language. However, we expect that $\textsc{WT}^-(\alpha)$ is $\forall \Sigma^p_1(\alpha)$-separated from $S^1_2^-(\alpha)$. The intuitive reason is that $S^1_2^-(\alpha)$ proves the formula $\text{IITER}(\alpha)$, which says that $\alpha$ is an instance of the class $\text{IITER}$ (cf. Definition 4.4) and $\alpha$ has a solution, but we conjecture that $\textsc{WT}^-(\alpha)$ cannot prove $\text{IITER}(\alpha)$.

**Notation 6.16** In this chapter, for a polynomial-time Turing machine $M$, we write $M^\Omega$ to denote the machine $M$ which takes $\Omega$ as an oracle.

We assume that polynomial-time Turing machines are clocked (that is to say, a machine that counts its own steps of computation). Also, we assume that there is a suitable bound $T(n)$ on the runtime of $M^\Omega(x)$ such that $M^\Omega(x)$ will run for exactly $T(n)$-many steps. A configuration of $M$ is the complete description of $M$’s tape contents, head positions and current state at a given time, which can be coded by a number $c < 2^{p(n)}$, for some polynomial $p$. The full computation of $M$ is coded by a number $z = \langle c_0, c_1, \ldots, c_T \rangle$, where $c_i$ is the $i$-th configuration of $M^\Omega(x)$. The exact encoding is not important. However, it is important that information about tape contents, state and head position at a particular point in time can be extracted by polynomial-time functions. We also assume that when $M^\Omega(x)$ stops, then its output is stored on a special output tape which can easily be extracted from $c_T$. If $z$ encodes the full computation of $M^\Omega(x)$, then $M^\Omega(x)$’s output is denoted by $\text{out}(z)$.

**Definition 6.17** Let $\phi$ be a $\Sigma^1_b^b$-formula.

1. If $\phi$ is a quantifier-free formula, then a witness query $q$ to $\phi$ returns 1, if $\phi(q)$ is true, and returns 0, otherwise.

2. Suppose that $\phi$ is of the form $(\exists y \leq t) \psi(q, \bar{x}, \bar{X}, y)$. Then a witness query $q$ to $\phi$ returns a witness $y \leq t$ satisfying $\psi(q, \bar{x}, \bar{X}, y)$, if $\phi(q)$ is true, and returns $t + 1$, otherwise. If $\phi$ is of the form $(\forall y \leq t) \psi(q, \bar{x}, \bar{X}, y)$, then a witness query $q$ to $\phi$ is the same as a witness query $q$ to $(\exists y \leq t) \neg \psi(q, \bar{x}, \bar{X}, y)$.

**Definition 6.18** Let $\phi$ be a $\Sigma^1_b^b$-formula and $t$ be a term. Then a $\exists \Sigma^1_b$ comprehension oracle based on $\phi$ and $t$ is the $\exists \Sigma^1_b$-formula $\Omega(q, \bar{x}, \bar{X}, z)$ of the form

$$
\exists y \leq t \forall z < t [ \text{Bit}(y, z) = 1 \leftrightarrow \phi(q, \bar{x}, \bar{X}, z)].
$$
Notation 6.19 Let \( \phi(\vec{x}, \vec{X}) \in s\Sigma^b_1 \). If \( \phi \in \Sigma^b_0 \), then the formula “\( w \) witnesses \( \phi \)” is just \( \phi \). If \( \phi \in s\Sigma^b_{\geq 1} \) and \( \phi \) is of the form \( \exists y \leq t \phi_0(y) \). Then “\( w \) witnesses \( \phi \)” is the formula \( w \leq t \wedge \phi_0(w) \).

Lemma 6.20 (Witnessing Lemma for \( WT \)) Suppose that \( WT \) proves a sequent \( \Gamma(\vec{a}, \vec{A}) \rightarrow \Delta(\vec{a}, \vec{A}) \) which consists only of \( s\Sigma^b_1 \)-formulae. Let \( \Gamma = \phi_1, \ldots, \phi_l \) and \( \Delta = \psi_1, \ldots, \psi_m \), where each \( \phi_i \) and \( \psi_j \) is of the form \((\exists y \leq s_i) \phi_i(\vec{a}, \vec{A}, y_i)\) and \((\exists z_j \leq t_j) \psi_j(\vec{a}, \vec{A}, z_j)\), respectively (some quantifiers may be omitted). Then there exists a polynomial-time oracle Turing machine \( M \) and some \( s\Sigma^b_1 \) comprehension oracles \( \Omega_1(q, \vec{a}, \vec{A}) \), \( \ldots \), \( \Omega_k(q, \vec{a}, \vec{A}) \) such that \( M \) makes \( O(1) \)-many witness queries to each of \( \Omega_1, \ldots, \Omega_k \) and \( WT \) proves the following:

If “\( w_i \) witnesses \( \phi_i \)”, for \( i = 1, \ldots, l \), and \( z \) codes the complete computation of \( M^{\Omega}(\vec{a}, \vec{w}) \), then \( (j, w) = \text{out}(z) \) such that “\( w \) witnesses \( \psi_j \)”, for some \( j \in \{1, \ldots, m\} \).

Proof. By Theorem 6.15, let \( P \) be an LK-WT proof of \( \Gamma \rightarrow \Delta \). Hence, \( P \) only contains \( s\Sigma^b_1 \)-formulae. We show the conclusion of this lemma by induction on the depth of a sequent \( S \) in \( P \). Again, the inductive proof splits into cases, depending on whether \( S \) is an initial sequent or generated by the use of an inference rule. Here, we deal only with the case when the sequent \( S \) is obtained by the application of the \( \Sigma^b_0 \)-LIND rule. The other cases are simpler. Hence, assume that \( S \) is the bottom sequent of

\[
\psi(b), \Pi \rightarrow \Lambda, \psi(b + 1) \\
\psi(0), \Pi \rightarrow \Lambda, \psi(|t|)
\]

Assume that \( \Lambda \) contains \( m \)-many formulae in total. Let \( M_1 \) be the polynomial-time oracle Turing machine and \( \Omega_1(b), \ldots, \Omega_k(b) \) (with some other parameters omitted) be the \( s\Sigma^b_1 \) comprehension oracles for the upper sequent obtained by induction hypothesis. We now define the polynomial-time oracle Turing machine \( M \) and the \( s\Sigma^b_1 \) comprehension oracles for the bottom sequent. First, we define the \( s\Sigma^b_1 \) comprehension oracle \( \Omega(q, \vec{a}, \vec{A}) \) as follows:

\[
(\exists y \leq 2^{|t|+1} - 1)(\forall z < |t| + 1)[\text{Bit}(y, z) = 1 \leftrightarrow \phi(q, \vec{a}, \vec{A}, z)],
\]

where \( \phi(q, \vec{a}, \vec{A}, z) \) is the following:

\[
q = 1 \wedge \psi(z).
\]

Now, the machine \( M^{\Omega, \vec{a}, \vec{w}} \) is defined as follows. The machine \( M^{\Omega, \vec{a}, \vec{w}}(\vec{a}) \) starts by asking a witness query \( q = 1 \) to \( \Omega \). Then \( M \) proceeds as follows: if \( \psi(0) \) is false, then \( M \) aborts. Otherwise, it determines the largest \( i \leq |t| \) such that \( \psi(i) \) is true. If \( i < |t| \), then \( M \) runs \( M^{\Omega}(i, \vec{a}, \vec{w}) \) and uses the same output as \( M_1 \). If \( i = |t| \), then \( M \) outputs \( (m + 1, 0) \). We note here that \( M \) cannot make witness queries to \( \psi(i) \), but needs to establish the largest \( i \) via the value returned by the witness query to \( \Omega \).

It is easy to see that \( M \) is a polynomial-time oracle Turing machine, which makes \( O(1) \)-many witness queries to \( \Omega, \vec{a}, \vec{w} \). Furthermore, it is straightforward to check that the above arguments are formalizable in \( WT \).

\[\square\]
6.3 Improved New-style Witnessing Theorem for $U^1_2$

The following theorem is the witnessing theorem for $\text{WT}$ and is a direct application of Lemma 6.20:

**Theorem 6.21** (Witnessing Theorem for $\text{WT}$) Suppose that $\text{WT}$ proves $(\exists y) \varphi(\vec{a}, \vec{A}, y)$, where $\varphi \in s\Sigma^b_1$. Then there is a polynomial-time oracle Turing machine $M$ and a $s\Sigma^b_1$-comprehension oracle $\Omega(q, \vec{a}, \vec{A})$ such that $M$ makes $O(1)$ witness queries to $\Omega$ and $\text{WT}$ proves

“If $z$ codes the complete computation of $M^\Omega(\vec{a})$, then $\varphi(\vec{a}, \vec{A}, \text{out}(z))$ is true.”

6.3.1 Canonical Evaluation and Verification

In this subsection we define the notion of canonical evaluation and verification of a $\Sigma^b_1$ formula. For later developments, it is important that these notions make sense over the base theory $\text{WT}$.

Let $\varphi(\vec{x}, \vec{X})$ be a $\Sigma^b_1$-formula in prenex form with all free variables indicated. For the sake of simplicity and notational convenience, assume that the quantifiers all use the same bounding term $t(\vec{x})$. Hence, $\varphi$ is of the form

$$(Q_1 y_1 \leq t)(Q_2 y_2 \leq t)(Q_3 y_3 \leq t) \cdots (Q_k y_k \leq t) \psi(y, \vec{x}, \vec{X})$$

(6.5)

where $\psi$ is a quantifier-free formula.

Before we give formal definitions of the notions of canonical evaluation and verification, let us first explain the intent behind the definitions. Basically, we want to code the truth of $\phi$ into a second-order object. Also, we want a second-order object that carries the witnesses to the existential quantifiers in either $\phi$ or $\neg \phi$, depending on the truth of $\phi$. Hence, the definition of canonical evaluation is done in two stages: first, the construction of a second-order object $\alpha$ that stores the truth of $\phi$; second, the construction of a second-order object $\beta$ that carries the witnesses.

Intuitively, the object $\alpha$ can be viewed as a full binary tree $T_\alpha$ with depth $k \cdot (|t| - 1)$ such that the nodes at depth $d \cdot (|t| - 1)$ store the truth of

$$(Q_{d+1} y_{d+1} \leq t) \cdots (Q_k y_k \leq t) \psi(a_1, \ldots, a_d, y_{d+1}, \ldots, y_k).$$

As a result, the root node would store the truth of $\phi$ and the leaves would store the truth of $\psi(a_1, \ldots, a_k)$.

In the same way that $\alpha$ can be viewed as a full binary tree with depth $k \cdot (|t| - 1)$, so can $\beta$. However, in the case of $\beta$, the tree $T_\beta$ is used to carry the witnesses to the existential quantifiers in either $\phi$ or $\neg \phi$. So, for example, if $\phi$ is false, then available in the root node of $T_\beta$ is a path from the root to a leaf $v$, which shows that $\phi$ is false. Note that the length of such paths is equal to $k \cdot |t|$. 87
6. Extended Linear Local Improvement Principles and $U^1_2$

Let us explain how $\alpha$ and $\beta$ work together in more detail. Assume that $\phi$ is $(\exists y \leq 5) \psi(y)$. Then one can view the nodes in $T_\alpha$ and $T_\beta$ as being numbered from 1 to $2^3 + 7$ in the following way: If $x$ is a node in $T_\alpha$ (respectively, $T_\beta$), then the two children of $x$ are numbered $2x$ (left child) and $2x + 1$ (right child). Note that the root node is numbered $2^0$ and the leaves are numbered $2^3 + y$, for $y = 0, 1, \ldots, 7$. Now, assume that $\psi(y)$ is false for $y = 1, 5$ and true for the rest, that is to say, those values $y$ such that $y \leq 5$ and $y$ is different from 1 and 5.

Let us first explain how the nodes in $T_\alpha$ are labeled. For $y \leq 5$ and for every leaf node $2^3 + y$, $2^3 + y$ is labeled with the truth value of $\psi(y)$ (this corresponds to condition (a) below). However, note that there are two extra leaves left in $T_\alpha$ that are not labeled; namely, $2^3 + 6$ and $2^3 + 7$. These leaves will be labeled with $\bot$ (this corresponds to condition (b) below). For a node $x$ in $T_\alpha$ that is not a leaf node, the label at $x$ is $\top$ if one of its children, $2x$ and $2x + 1$, is labeled with $\top$. Otherwise, $x$ is labeled with $\bot$ (this corresponds to condition (c) below). Note that, if $\phi$ were to start with a universal quantifier, then things would need to be adjusted accordingly. In particular, $2^3 + 6, 2^3 + 7$ are now labeled with $\top$ and a non-leaf node $x$ is $\top$ if, and only if, $2x$ and $2x + 1$ are labeled with $\top$. This finishes the case for $T_\alpha$.

Let us now explain how the nodes in $T_\beta$ are labeled. All leaf nodes $2^3 + y$ are labeled with $\beta(2^3 + y) = 0$ (this corresponds to condition (e) below). Let $x$ be a non-leaf node in $T_\beta$. Then $x$ is labeled with $(1^t)^c \beta(2x + 1)$, if $2x + 1$ is labeled with $\top$ in $T_\alpha$, and is labeled with $(0^t)^c \beta(2x)$ otherwise (this corresponds to condition (e) below). Here, the function $s^c s'$ is the sequence concatenation; that is to say,

$$\langle x_1, \ldots, x_k \rangle \langle y_1, \ldots, y_l \rangle = \langle x_1, \ldots, x_k, y_1, \ldots, y_l \rangle.$$

Note that, for our choice of $\phi$ above and that $\psi(y)$ is true for $y = 1, 5$ and false for all $y \leq 5$ that are not equal to 1 or 5, we have that $\alpha(2^1) = \top$ and $\beta(2^1) = \langle 1, 0, 0 \rangle$, where the entries in $\beta(2^1)$ correspond to the binary representation of the largest value $y \leq 5$ that witnesses $\phi$.

We now make these intuitions formal by setting the following conditions on $\alpha$ and $\beta$. Again, let $\phi$ be as in (6.5).

(a) For all $a_1, \ldots, a_k$, if $a_1, \ldots, a_k \leq t$, then

$$\alpha(2^{|t|} + a_1, \ldots, 2^{|t|} + a_k) \leftrightarrow \psi(a_1, \ldots, a_k).$$

(b) For $i = 1, \ldots, k; j = i, \ldots, k; a_1, \ldots, a_{i-1} \leq t; t < a_i < 2^{|t|}$ and $b_{i+1}, \ldots, b_j < 2^{|t|}$,

$$\alpha(2^{|t|} + a_1, \ldots, 2^{|t|} + a_i, 2^{|t|} + b_{i+1}, \ldots, 2^{|t|} + b_j) \leftrightarrow \top,$$

if $Q_i = \forall$, and $\top$ replaced by $\bot$, if $Q_i = \exists$.

(c) For $j = 1, \ldots, k; a_1, \ldots, a_{j-1} < 2^{|t|}; d < |t|$ and $c < 2^d$,

$$\alpha(\ldots, 2^{|t|} + c) \leftrightarrow \alpha(\ldots, 2 \cdot (2^d + c)) * \alpha(\ldots, 2 \cdot (2^d + c) + 1)$$

where “...” is $2^{|t|} + a_1, \ldots, 2^{|t|} + a_{j-1}$ and $* = \land$, if $Q_j = \forall$, and $* = \lor$, if $Q_j = \exists$. 

\[88\]
6.3. Improved New-style Witnessing Theorem for $U^1_2$

(d) For $j = 0, \ldots, k - 1; a_1, \ldots, a_j < 2^{|i|}$,
$$\alpha(2^{|i|} + a_1, \ldots, 2^{|i|} + a_j) \leftrightarrow \alpha(2^{|i|} + a_1, \ldots, 2^{|i|} + a_j, 2^0).$$

(e) For all $a_1, \ldots, a_k < 2^{|i|}$,
$$\beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_k) = \langle \rangle.$$

(f) For $j = 0, \ldots, k - 1$ and $a_1, \ldots, a_j < 2^{|i|}$,
$$\beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_j) = \beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_j, 2^0).$$

(g) For $j = 1, \ldots, k; a_1, \ldots, a_{j-1} < 2^{|i|}; d < |i|; c < 2^d$ and $Q_j = \forall$, we have that $\beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c)$ is equal to
$$\langle 1 \rangle \land \beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c + 1),$$
if $-\alpha(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c)$ is true, and,
$$\langle 0 \rangle \land \beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c),$$
otherwise.

(h) For $j = 1, \ldots, k; a_1, \ldots, a_{j-1} < 2^{|i|}; d < |i|; c < 2^d$ and $Q_j = \exists$, we have that $\beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c)$ is equal to
$$\langle 1 \rangle \land \beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c + 1),$$
if $\alpha(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c + 1)$ is true, and,
$$\langle 0 \rangle \land \beta(2^{|i|} + a_1, \ldots, 2^{|i|} + a_{j-1}, 2^d + c),$$
otherwise.

Note, from the above definition, that $\alpha(\langle \rangle)$ is equivalent to $\alpha(1)$ and $\beta(\langle \rangle)$ is equal to $\beta(1)$.

**Remark 6.22** Above, we considered $\beta$ as a string function. However, when formalizing the above in bounded arithmetic, we have to work with sets. Thus, we now consider sets representing bit-graphs of string functions. The set $\beta$ represents the bit-graph of string function $\overline{\beta}$ if, and only if, for all $x, y$,
$$\overline{\beta}(x) = y \leftrightarrow y \leq t(x) \land \forall i < |t(x)| t(x, i) \leftrightarrow \text{Bit}(y, i) = 1].$$

In the following, we will use $\beta$ for a string function and its bit-graph, it will be clear from the context what we mean.

**Definition 6.23** The second-order object $\langle \alpha, \beta \rangle$ canonically evaluates (or simply evaluates) $\phi(\overline{x}, \overline{X})$ provided that all the preceding conditions (a)-(h) hold. $\langle \alpha, \beta \rangle$ canonically verifies (or simply verifies) $\phi(\overline{x}, \overline{X})$ provided that $\langle \alpha, \beta \rangle$ evaluates $\phi$ and $\alpha(1)$ is true.
6. Extended Linear Local Improvement Principles and $U_2^1$

We next extend the definitions of canonical evaluation and verification to $\Sigma_1^{1,b}$-formulae as follows.

**Definition 6.24** Assume that $\phi(x, X) \equiv (\exists Y)\psi(x, X, Y)$, where $\phi \in \Sigma_0^{1,b}$, and let $\gamma$ be a second-order object. Then the second-order object $\langle \alpha, \beta \rangle$ canonically verifies (or simply verifies) that $\gamma$ witnesses $\phi$ if, and only if, $\langle \alpha, \beta \rangle$ verifies $\phi(x, X, \gamma)$.

**Lemma 6.25** Again, for a binary string $1i_1 \ldots i_j$, the expression $(1i_1 \ldots i_j)_2$ denotes the number represented by this string. Then the following facts are easily provable in WT:

1. If $\beta(\bar{a}, 1) = (i_1, \ldots, i_n)$, then
   $$\beta(\bar{a}, (1i_1 \ldots i_j)_2) = (i_{j+1}, \ldots, i_n)$$
   for $j = 0, 1, \ldots, n$.
2. If $Q_{d+1} = \forall, \bar{a} = (a_1, \ldots, a_d)$, $\neg \alpha(\bar{a}, 1)$ is true and $\beta(\bar{a}, 1) = (i_1, \ldots, i_n)$, then
   $$\neg \alpha(\bar{a}, (1i_1 \ldots i_j)_2)$$
   is true, for $j = 0, 1, \ldots, n$.
3. If $Q_{d+1} = \forall, \bar{a} = (a_1, \ldots, a_d)$ and $\neg \alpha(\bar{a}, (1i_1 \ldots i_n)_2)$ is true, then
   $$\neg \alpha(\bar{a}, (1i_1 \ldots i_j)_2)$$
   is true for $j = n, \ldots, 1, 0$.
4. If $Q_{d+1} = \exists, \bar{a} = (a_1, \ldots, a_d)$, $\alpha(\bar{a}, 1)$ is true and $\beta(\bar{a}, 1) = (i_1, \ldots, i_n)$, then
   $$\alpha(\bar{a}, (1i_1 \ldots i_j)_2)$$
   is true, for $j = 0, 1, \ldots, n$.
5. If $Q_{d+1} = \exists, \bar{a} = (a_1, \ldots, a_d)$ and $\alpha(\bar{a}, (1i_1 \ldots i_n)_2)$ is true, then
   $$\alpha(\bar{a}, (1i_1 \ldots i_j)_2)$$
   is true for $j = n, \ldots, 1, 0$.

**Theorem 6.26** Let $\phi(x, X) \in \Sigma_1^{1,b}$. Then WT proves

“If $\langle \alpha, \beta \rangle$ evaluates $\phi$, then $\alpha(1)$ is true if and only if $\phi$ is true.”.

Thus, WT proves

“Suppose that $\langle \alpha, \beta \rangle$ verifies $\phi$. Then $\phi$ is true.”.

If $\phi(x, X)$ is a formula of the form $(\exists Z)\psi(x, X, Z)$, where $\psi \in \Sigma_0^{1,b}$, then WT proves

“Suppose that $\langle \alpha, \beta \rangle$ verifies that $\gamma$ witnesses $\phi$. Then $\psi(x, X, \gamma)$ is true.”.
6.3. Improved New-style Witnessing Theorem for $U_1$

Proof. The proof is by structural induction on $\phi$ outside $WT$. If $\phi$ is a quantifier-free formula, then it trivially follows from condition (a) above. So, assume that $\phi$ is not a quantifier-free formula. Furthermore, for the sake of simplicity, assume that $\phi$ is of the form $(\forall x \leq t)(\exists y \leq t)\psi(x,y)$, where $\psi$ is quantifier-free (the case when $\phi$ is as in (6.5), is treated similarly) and $\phi_0(x) \equiv (\exists y \leq t)\psi(x,y)$.

(Reasoning in $WT$). Suppose that $\langle \alpha, \beta \rangle$ evaluates $\phi$. Define $\alpha_x, \beta_x$ from $\alpha$ and $\beta$ so that

(A) $\langle \alpha_x, \beta_x \rangle$ evaluates $\phi_0(x)$, and

(B) $\alpha_s(s)$ is true if, and only if, $\alpha(2^{|x|} + x, s)$ is true.

Then by (A) and the induction hypothesis,

$\alpha(1) \leftrightarrow \phi_0(x)$, \hspace{1cm} (6.6)

Let us first show the “if” direction. Suppose that $(\forall x \leq t)\phi_0(x)$ is true and, for the sake of contradiction, assume that $\neg \alpha(1)$ holds. Now $\beta(1) = \langle i_1, \ldots, i_n \rangle$. Let $a = (i_1 \ldots i_n)_2$. Note that $a \leq t$. By condition (2) in Lemma 6.25, $\neg \alpha(a)$ holds. By condition (d) and (B), $\neg \alpha(a)$ follows. By (6.6), $\neg \phi_0(a)$ is true, which is a contradiction to $(\forall x \leq t)\phi_0(x)$.

We next show the “only-if” direction. Suppose that $\neg \phi$ holds and let $a \leq t$ and $\neg \phi_0(a)$ is true. Therefore, by (6.6), (A) and (B), we get that $\neg \alpha(2^{|x|} + a)$ holds. By condition (3) in Lemma 6.25, $\neg \alpha(1)$ follows, which finishes the proof.

So far, we have talked about how to build canonical evaluations for $\Sigma^b_1$-formulae that are already in prenex form. However, we still need to build second-order objects that evaluate formulae that are not in prenex form. The next theorem, which is an adaptation of Theorem 4.4 of [BB14], formalizes that. In what follows, $\varphi^*$ denotes any prenex form of $\varphi$:

Theorem 6.27 ([BB14]) Let $\phi$ and $\psi$ be in prenex form with canonical evaluations given by $E_\phi$ and $E_\psi$.

- Suppose that $E$ is a canonical evaluation of $(\phi \land \psi)^*$. Then $E$ verifies $(\phi \land \psi)^*$ if, and only if, $E_\phi$ verifies $\phi$ and $E_\psi$ verifies $\psi$.

- Suppose that $E$ is a canonical evaluation of $(\phi \lor \psi)^*$. Then $E$ verifies $(\phi \lor \psi)^*$ if, and only if, $E_\phi$ verifies $\phi$ or $E_\psi$ verifies $\psi$.

- Suppose that $E$ is a canonical evaluation of $(\neg \phi)^*$. Then $E$ verifies $(\neg \phi)^*$ if, and only if, $E_\phi$ does not verify $\phi$.

Furthermore, the above statements are easily provable in $WT$.

6.3.2 Improved New-Style Witnessing Theorem for $U_1$

We are now going to establish conventions on how $PSPACE$ Turing machines run, how their computations are encoded and what it means for a $PSPACE$ Turing machine to either output a first- or a second-order object.
In what follows, for a second-order object \( X \), the notation \( X(i) \) gives the \( i \)-th bit of \( X \). Now, assume that \( M \) is a PSPACE Turing machine with space bound \( S(n) \) that is time-constructible. We also assume that, for an input \( x \) of length \( n \), the machine \( M \) will always take \( T(n) \)-many steps. A configuration \( C \) of \( M(x) \) is a complete description of \( M \)'s tape contents, head positions and current state at a given instant of time and is encoded by a first-order object \( C < 2^{s(n)} \), for some polynomial \( s \). The full computation of \( M(x) \) is encoded by a second-order object \( W \) by simply concatenating \( C_0, C_1, \ldots, C_{T(n)} \), where \( C_i \) codes the \( i \)-th configuration of \( M^X(x) \). The exact details of the encoding are not important; however, we must be able to extract information about the states, tape head positions and tape contents using polynomial-time functions that do not depend on oracles. Also, we must be able to express the statement that \( W \) codes the full computation of \( M^X(x) \) by a \( \Pi^1_1 \)-formula.

The notion of a PSPACE oracle machine \( M^X(x) \) outputting either a first- or a second-order object is from [BB14]. The second-order output of \( M^X(x) \) will be denoted by \( \text{Out}(W) \). We require that the encoding \( W \) of the full computation of \( M^X(x) \) allows the \( i \)-th bit of \( \text{Out}(W) \) to be computable by a polynomial-time function relative to \( W \): there is a polynomial-time function \( f(i) \) such that \( W(f(i)) \) gives you the \( i \)-th bit of \( \text{Out}(W) \). This can be done by requiring the machine \( M \) to write each \( \text{Out}(W)(i) \) at a special tape location at a prespecified time that is easily computed from \( i \). The case of the first-order output \( \text{out}(W) \) of \( M^X(x) \) is dealt similarly.

An important notion that will be used during the proof of the new-style witnessing theorem for \( U_2^1 \) is the notion of a PSPACE oracle machine with consistent restarts. Before we formally define this notion, first consider the following example of a PSPACE oracle machine \( M^X(x) \) that makes use of two PSPACE oracle submachines \( M_1^X(x) \) and \( M_2^{X,Y}(x) \) during its computation (the example will help in understanding the notion of PSPACE machines with restarts). Then we explain what we require of these kind of machines.

Assume that the machine \( M^X(x) \) starts by simulating the machine \( M_1^X(x) \). If the second-order output \( Y \) of \( M_1^X(x) \) satisfies certain condition(s), then \( M^X(x) \) stops and outputs the same as \( M_1^X(x) \). Otherwise, \( M^X(x) \) proceeds by simulating \( M_2^{X,Y}(x) \) and outputs whatever \( M_2^{X,Y}(x) \) outputs.

First, note that, if in case the machine \( M^X(x) \) needs to simulate the machine \( M_2^{X,Y}(x) \), then \( M \) does not have enough space in order to store the second-order output \( Y \) of \( M_1^X(x) \) as it is exponentially big. As a result, during the simulation of \( M_2^{X,Y}(x) \) by \( M^X(x) \), whenever a query to \( Y \) arises, the machine \( M^X(x) \) pauses the simulation of \( M_2^{X,Y}(x) \) and fully resimulates \( M_1^X(x) \). During this resimulation of \( M_1^X(x) \), \( M \) saves in its memory the information it needs in order to recontinue the simulation of \( M_2^{X,Y}(x) \). This process continues until \( M^X(x) \) fully simulates the computation of \( M_2^{X,Y}(x) \). Because the machine \( M^X(x) \) may resimulate the machine \( M_1^X(x) \) more than once and \( M_1 \) is a deterministic Turing machine, the \( i \)-th resimulation of \( M_1^X(x) \) by \( M \) is consistent with the first time \( M_1^X(x) \) has been simulated by \( M \) (in the sense that their respective \( j \)-th configurations are identical). Hence, we will call the machine \( M \) a PSPACE machine with consistent restarts. Second, note that the machines \( M_1 \) and \( M_2 \) are potentially PSPACE machines with consistent restarts as well.

In what follows, it is crucial that machines with consistent restarts, like \( M \) above, can always be defined so that their computations are regular: we know exactly when the \( i \)-th re-
6.3. Improved New-style Witnessing Theorem for \( U_2^1 \)

Computation of \( M^X(x) \) occurs within \( M \); when the wait phases happen; where the partial computations of \( M_2^{X,Y}(x) \) (which, patched together, form the full computation of \( M_2^{X,Y}(x) \)) occur; and where the second-order output(s) of \( M_2^{X,Y}(x) \) occur. To make the computation regular, one can require for \( M \) to only simulate one step of \( M_2^{X,Y}(x) \) at a time; then pauses \( M_2^{X,Y}(x) \)'s computation and fully resimulates \( M_1^X(x) \). Furthermore, a polynomial amount of steps are needed between each simulation step to make the overall computation regular.

In what follows, we write \([a]\) for \( \{0, 1, \ldots, a-1\} \). Also, when we write \( j \in n, n \) is a first-order object encoding a set.

**Definition 6.28** Let \( M^X(x) \) be a PSPACE oracle Turing machine with runtime \( T(x) \). Let \( B(x) \) be a bound on \( M \)'s configurations. Let \( C : [T(x)] \rightarrow P \{B(x)\} \) be a partial function. Then \( C \) is consistent with \( M^X(x) \) if, and only if, for all input \( t < T(x) \), if \( t, t+1 \in \text{dom}(C) \), then \( C(t+1) \) follows from \( C(t) \) by one step of \( M^X(x) \)'s transition function.

**Definition 6.29** Let \( M \) be a PSPACE oracle Turing machine. Let \( T(x) \) be the runtime of \( M^X(x) \). Let \( B(x) \) be a bound on \( M^X(x) \)'s configurations. Let \( f(x,i) \) be a polynomial-time function such that for all \( i < T(x) \) and \( j \in f(x,i) \), we require that \( j < i \). Let \( \text{cons}(x,j,i,v,u) \) be a polynomial-time predicate. Then we call \( (M,f,\text{cons}) \) a PSPACE oracle Turing machine with consistent restarts if and only if for all input \( x \) and oracle \( X \), all partial functions \( C : [T(x)] \rightarrow P \{B(x)\} \) and all \( 0 < t < T(x) \), if

1. \( \text{dom}(C) = \{t-1,t\} \cup f(x,t-1) \cup f(x,t) \),

2. \( C \) is consistent with \( M^X(x) \), and

3. for all \( j \in f(x,t-1) \), the predicate \( \text{cons}(j,t-1,C(j),C(t-1)) \) is true,

then for all \( j \in f(x,t) \), \( \text{cons}(j,t,C(j),C(t)) \) also holds.

We shall usually denote \((M,f,\text{cons})\) by just \( M \). It will be clear from the context that \( f \) and \( \text{cons} \) are present but omitted.

We should explain the above definition here. Suppose that \( M^X(x) \) is the previously discussed PSPACE oracle machine that makes use of the PSPACE submachines \( M_1^X(x) \) and \( M_2^{X,Y}(x) \). Let us first start with the function \( f(x,i) \). Suppose that time \( i \) corresponds to a recomputation of \( M_1^X(x) \) and time \( j \) corresponds to the time it has first been computed, then \( f(x,i) = \langle j \rangle \), otherwise, \( f(x,i) = \emptyset \). The function \( f(x,i) \) is able to compute such information without requiring a second-order oracle, because of the regularity of \( M^X(x) \)'s computation. Now, the predicate \( \text{cons}(x,j,i,v,u) \) is true if the parts corresponding to \( M_1 \) in \( u \) and \( v \) are the same.

**Definition 6.30** Let \( M \) be a PSPACE oracle Turing machine with consistent restarts. Then a second-order object \( W = \langle C_0, C_1, \ldots, C_T \rangle \) codes the full computation of \( M^X(x) \) locally if, and only if, \( C_0 \) codes the initial configuration of \( M^X(x) \) and for all \( t < T(x) \), \( C_{t+1} \) and \( C_t \) code two consecutive configurations of \( M^X(x) \), and \( C_T \) codes the final configuration of \( M^X(x) \).
6. Extended Linear Local Improvement Principles and $U_2^1$

**Definition 6.31** Let $(M, f, \text{cons})$ be a PSPACE oracle Turing machine with consistent restarts. Then a second-order object $W$ codes the full computation of $M^X(x)$ with consistent restarts if, and only if, $W$ codes the full computation of $M^X(x)$ locally and for all $i < T(x)$, $j \in f(x, i)$ and for all $u, v$ less than some polynomial bound $B(x)$ depending on $M$, if $u$ codes the $i$-th configuration of $M^X(x)$ in $W$ and $v$ codes the $j$-th configuration of $M^X(x)$ in $W$, then $\text{cons}(x, j, i, v, u)$ holds.

Note that the notion that a second-order object $W$ codes the full computation of a PSPACE oracle Turing machine $M^X(x)$ with consistent restarts is a $\Pi^b_1$-statement. This is because the part “$W$ codes the full computation of $M^X(x)$ locally” is a $\Pi^b_1$-statement and the other part “$u$ (respectively, $v$) codes the $i$-th (respectively, $j$-th) configuration of $M^X(x)$” is a $\Sigma^b_1$-statement.

**Lemma 6.32** Let $\varphi(\bar{a}, \bar{A}) \in \delta \Pi^b_1$.

1. There exists a PSPACE oracle Turing machine $M$ with consistent restarts such that WT proves:
   
   “If $W$ codes the full computation of $M^A(\bar{a})$ with consistent restarts, then $\text{Out}(W)$ canonically evaluates $\varphi(\bar{a}, \bar{A})$.”

2. There exists a PSPACE oracle Turing machine $M$ with consistent restarts such that WT proves:
   
   “If $W$ codes the full computation of $M^A(\bar{a})$ with consistent restarts, then $\text{Out}(W) = \langle U, V \rangle$ such that $V$ verifies that $U$ evaluates $\varphi(\bar{a}, \bar{A})$.”

**Proof Sketch.** The machine just follows the full binary evaluation tree in a depth-first search fashion starting from the highest value.

**Theorem 6.33** (New-style Witnessing Theorem for $U_2^1$)

(A) Suppose that $U^1_2$ proves $(\exists Z)\varphi(\bar{a}, \bar{A}, Z)$, where $\varphi$ is a $\Sigma^1_{b_0}$-formula. Then there is a PSPACE oracle Turing machine $M$ with consistent restarts such that WT proves the following statement: “If $W$ codes the full computation of $M^A(\bar{a})$ with consistent restarts, then $\langle V, Z \rangle = \text{Out}(W)$ such that $V$ verifies that $Z$ witnesses $(\exists Z)\varphi(\bar{a}, \bar{A}, Z)$.”.

(B) Suppose $U^1_2$ proves $(\exists y)\varphi(\bar{a}, \bar{A}, y)$, where $\varphi$ is a $\Sigma^1_{b_0}$-formula. Then there is a PSPACE oracle Turing machine $M$ with consistent restarts such that WT proves the following statement: “If $W$ codes the full computation of $M^A(\bar{a})$ with consistent restarts, then $\varphi(\bar{a}, \bar{A}, \text{out}(W))$ is true.”.

The new-style witnessing theorem for $U^1_2$ above (Theorem 6.33) is an improvement over Beckmann and Buss’s new-style witnessing theorem for $U^1_2$ (cf Theorem 4.5 of [BB14]) in the sense that our base theory WT is weaker than $S^1_2$. The proof technique we use here is similar to that of Theorem 4.5 of [BB14]. However, since WT cannot perform length induction on $\Sigma^b_1$-formulae, the cases of the cut and the $\delta \Sigma^1_{b_1}$-LIND rules are more involved in the proof of
the following witnessing lemma for $U^1_2$.

Theorem 6.33 follows from the following witnessing lemma:

**Lemma 6.34** (New-style Witnessing Lemma for $U^1_2$) Suppose that $U^1_2$ proves a sequent $\Gamma \rightarrow \Delta$ consisting only of $\Sigma^1_{b^*}$-formulae, with free variables $\vec{a}, \vec{A}$. Let $\Gamma = \phi_1, \ldots, \phi_k$ and $\Delta = \psi_1, \ldots, \psi_l$, where each $\phi_i$ is of the form $(\exists \psi_i)\phi'_i(\vec{a}, \vec{A}, \vec{Y}, \vec{Z})$ and each $\psi_i$ is of the form $(\exists Z_i)\psi'_i(\vec{a}, \vec{A}, Z_i)$ (some of the quantifiers may be omitted). Then there is a PSPACE oracle Turing machine $M$ with consistent restarts such that $WT$ proves the following statement:

“If $V_i$ verifies that $Y_i$ witnesses $\phi_i$, for $i = 1, \ldots, k$, and $W$ codes the full computation of $M^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$, then $j = out(W) \in \{1, \ldots, l\} \text{ and } \langle V, Z_j \rangle = Out(W)$ such that $V$ verifies that $Z_j$ witnesses $\psi_j$.”

**Proof.** By Theorem 6.13, let $P$ be an LK-$U^1_2$ proof of $\Gamma \rightarrow \Delta$ such that $P$ only contains $\Sigma^1_{b^*}$-formulae. The proof is by induction on the depth of a sequent $S$ in $P$. The inductive proof splits into cases, depending on whether $S$ is an initial sequent or generated by the use of an inference rule. Here, we will consider only the case of the cut rule and the $\Sigma^1_{b^*}$-LIND rule. We omit the other cases as they are treated in a similar way to the ones in the proof of Theorem 4.6 of [BB14].

Suppose that $S$ is obtained by the application of the cut rule. Hence, $S$ is the bottom sequent of

\[
S_1 = \Gamma \rightarrow \Delta, \chi \quad S_2 = \chi, \Gamma \rightarrow \Delta
\]

For $i = 1, 2$, let $M_i$ be the PSPACE oracle Turing machine with consistent restarts obtained by the induction hypothesis for the sequent $S_i$. Furthermore, let $l$ be the number of formulae in $\Delta$. First, let us describe the PSPACE oracle Turing machine $M$ with consistent restarts for the lower sequent $S_i$. The machine $M$, on inputs $\vec{a}$ and oracles $\vec{A}, \vec{V}$ and $\vec{Y}$, simulates $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$. If $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$ finishes with a second-order output $\langle U_i, Z_i \rangle$, for some $i \in \{1, \ldots, l\}$, such that $U_i$ verifies that $Z_i$ witnesses $\psi_i$, where $\psi_i$ is a formula in $\Delta$, then $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$ stops and its outputs are defined to be the same as $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$’s. Otherwise, $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$’s second order output is $\langle V_0, Y_0 \rangle$.

In that case, the machine $M^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$ continues its computation by simulating $M_2^{\vec{A}, V_0, Y_0, \vec{V}, \vec{Y}}(\vec{a})$ and outputs whatever $M_2^{\vec{A}, V_0, Y_0, \vec{V}, \vec{Y}}(\vec{a})$ outputs. Now, remember that the machine $M$ cannot store $V_0$ and $Y_0$, as $M$’s memory is too small to store exponentially big objects. Therefore, to work around this problem, we will use recomputations: during the simulation of $M_2$ by $M$, whenever $M_2$ asks a query to either $V$ or $Y$, $M$ will need to fully resimulate $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$. Since $M_1$ is deterministic, recomputed values of $V_0$ and $Y_0$ are always consistent. Also, remember that for regularity, we require for $M$ to fully simulate $M_i^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$’s computation after simulating each step of $M_2^{\vec{A}, V_0, Y_0, \vec{V}, \vec{Y}}(\vec{a})$.

We will now argue that the above arguments are formalizable in the theory $WT$. Let $m$ be the number of formulae in $\Gamma$. Suppose that the computation of $M^{\vec{A}, \vec{V}, \vec{Y}}(\vec{a})$ includes
Then, structuring a canonical evaluation of the computation of \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) with consistent restarts. Our tasks are to extract out of \( W \) an object \( W_1 \) that codes the computation of the oracle PSPACE machine \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) with consistent restarts, to extract second-order objects \( V_0 \) and \( V_1 \) such that \( V_0 \) verifies that \( V_1 \) witnesses \( \chi \) and to extract an object \( W_2 \) such that \( W_2 \) codes the computation of the oracle PSPACE machine \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) with consistent restarts. Then, applying the induction hypothesis, we will achieve our goal; that is to say, we can extract out of \( W \) a pair \( U_j \) and \( Z_j \) such that \( U_j \) verifies that \( Z_j \) witnesses a formula \( \psi \), in \( \Delta \). Again, the point here is that the computation of \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) is regular. Therefore, we know exactly where to look for \( W_1 \) in \( W \). Then, applying the induction hypothesis, we obtain a pair \( \langle V_0, Y_0 \rangle = \text{Out}(W_i) \) such that \( V_0 \) verifies that \( Y_0 \) witnesses \( \chi \). Similarly to \( W_1 \), we know exactly where all partial computations of \( M_2 \) occur within \( W \). Then, concatenating them all together, in the right order, we get \( W_2 \). However, in order to make sure that \( W_2 \) codes the correct computation of the oracle PSPACE machine \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) with consistent restarts, it is important that every time a recomputation of \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) occurs within \( W_i \), its second-order output is consistent with \( \langle V_0, Y_0 \rangle \). But this is not a problem, since \( W \) codes the correct computation of the PSPACE machine \( M^\vec{A},\vec{V},\vec{Y}(\vec{a}) \) with consistent restarts, by assumption.

In what follows, let \( \text{Eval}(V, \varphi) \) be a \( \Pi^1_1 \)-formula that states that \( V \) is a canonical evaluation of \( \varphi \). Also, remember that a canonical evaluation of a \( \Sigma^1_{0,b} \)-formula is a pair \( E = \langle \alpha, \beta \rangle \). To check if \( \alpha(i) \) is true, we check \( E(0, i) \).

Consider the case of the \( s\Sigma^1_{1,b} \)-LIND rule. This case is a generalization of the cut rule. Suppose that \( S \) is the bottom sequent of

\[
\frac{\chi_i(a_0), \Gamma \rightarrow \Delta, \chi_i(a_0 + 1)}{\chi_i(0), \Gamma \rightarrow \Delta, \chi_i(|\Gamma|)}
\]

Without loss of generality, assume that \( \chi_i(i) \) is of the form \( (\exists Z)\chi'_i(i, Z) \). Let \( M_1 \) be the PSPACE machine with consistent restarts given by the induction hypothesis for the upper sequent and let \( l \) be the number of formulae in \( \Delta \). We now show how to define a PSPACE machine \( M \) with consistent restarts for the lower sequent. The machine \( M \) takes as inputs \( \vec{a} \) and as oracles \( V_0 \) and \( V_1 \) (\( M \) also has additional oracles, namely the witnesses and verifiers for \( \Gamma \), but also \( \vec{A} \) as oracles; however, we omit them in the following arguments, for notational convenience). The machine \( M \) starts by constructing a canonical evaluation \( U_0 \) of the formula \( \text{Eval}(V_0, \chi'_i(0, Y_0)) \). Then, \( M \) checks whether \( U_0(0, 1) \) and \( V_0(0, 1) \) are true or not. If not, then \( M \) outputs an evaluation of \( \perp \). Otherwise, we have that \( V_0 \) is a verifier of \( \chi'_i(0, Y_0) \) and \( M \) proceeds by simulating \( M_1 \) with \( a_0 \) set to 0, which we denote \( M^\vec{A},\vec{V}_i,\vec{Y}_i[a_0 := 0] \) (again, we omit extra parameters), and let \( j_1 \) denote the first-order output of \( M^\vec{A},\vec{V}_i,\vec{Y}_i[a_0 := 0] \) and \( \langle V_1, Y_1 \rangle \) denote its second-order output. If \( j_1 < l \), then \( M \) outputs \( j_1 \) and \( \langle V_1, Y_1 \rangle \). Otherwise, \( M \) carries on its computation by constructing a canonical evaluation \( U_1 \) of \( \text{Eval}(V_1, \chi'_i(1, Y_1)) \). Next, \( M \) checks whether \( U_1(0, 1) \)
and \( V_1(0, 1) \) are true or not. If not, then \( M \) outputs an evaluation of \( \bot \). Otherwise, \( V_1 \) is a verifier of \( \chi_1'(1, Y_1) \) and \( M \) repeats the same procedure, but now with \( M_{V_0,Y_0}^{V_1,Y_1}[a_0 := 1] \) and potentially continues for \( a_0 = 2, 3, \ldots, t_{\text{end}} < |t| \). At \( t_{\text{end}} \), the outputs \( j_{\text{end}}+1 \) and \( (V_{\text{end}}+1, Y_{\text{end}}+1) \) of \( M_{V_1,Y_1}^{V_0,Y_0}[a_0 := t_{\text{end}}] \) are the desired outputs for \( M \). Again, note that the second-order output \( \langle V_j, Y_j \rangle \) of \( M_{V_j,Y_j}^{V_{j-1},Y_{j-1}}[a_0 := j - 1] \) is exponentially large and cannot be written out in polynomial space. Therefore, as in the case of the cut rule, whenever \( M_{V_j,Y_j}^{V_{j-1},Y_{j-1}}[a_0 := j] \) queries either \( V_j \) or \( Y_j \), the machine \( M \) pauses the computation of \( M_{V_j,Y_j}^{V_{j-1},Y_{j-1}}[a_0 := j] \) and resimulates the entire computation of \( M_{V_j,Y_j}^{V_{j-1},Y_{j-1}}[a_0 := j - 1] \). These recomputations are carried out recursively, but only to a depth of \(|t| \). Similarly, when computing \( U_j \)'s, the machine \( M \) needs to resimulate the entire computation of \( M_{V_j,Y_j}^{V_{j-1},Y_{j-1}}[a_0 := j - 1] \). Again, we require that \( M \)'s computation is regular; therefore, a full resimulation is needed between each step. Because of regularity, we know where the computations of \( M_{V_0,Y_0}^{V_1,Y_1}[a_0 := 0], \ldots, M_{V_0,Y_0}^{V_{t_{\text{end}}},Y_{t_{\text{end}}}}[a_0 := t_{\text{end}}] \) occur in \( M_{V_0,Y_0}^{V_{t_{\text{end}}},Y_{t_{\text{end}}}}(\bar{a}) \)'s computation and, also, where to find \( V_0, V_1, \ldots, V_{t_{\text{end}}}, Y_{t_{\text{end}}} \) and \( U_0, U_1, \ldots, U_{t_{\text{end}}} \) in \( M_{V_0,Y_0}^{V_{t_{\text{end}}},Y_{t_{\text{end}}}}(\bar{a}) \)'s computation.

Arguing within \( W \), we want to show that if \( V_0 \) verifies that \( Y_0 \) witnesses \( \chi_0'(0, Y_0) \) and \( W \) codes the computation of \( M_{V_0,Y_0}^{V_{t_{\text{end}}},Y_{t_{\text{end}}}}(\bar{a}) \) with consistent restarts (some extra parameters are omitted; in particular, the verifiers and witnesses for the formulae in \( T \)), then \( (V, Y) = \text{Out}(W) \) such that \( V \) verifies that \( Y \) witnesses a formula in \( \Delta, \chi_0(t_{\text{end}} + 1) \). Note that the reasoning behind the case of the induction rule is a generalization of the cut rule. Hence, let \( W_0, \ldots, W_{t_{\text{end}}} \) be from \( W \), which code the computations of \( M_{V_0,Y_0}[a_0 := 0], \ldots, M_{V_{t_{\text{end}}},Y_{t_{\text{end}}}}[a_0 := t_{\text{end}}] \) with consistent restarts, respectively, and where \( V_1, V_{t_{\text{end}}}, Y_{t_{\text{end}}} \) are from \( W_1, \ldots, W_{t_{\text{end}}} \), respectively. Similarly, let \( U_0, U_1, \ldots, U_{t_{\text{end}}} \) be a sequence from \( W \), which by construction has the property that

\[
U_j \text{ evaluates } \text{Eval}(V_j, \chi_0'(j, Y_j)), \text{ for } j = 0, 1, \ldots, t_{\text{end}}.
\]

Now, to prove that \( M_{V_0,Y_0}^{V_{t_{\text{end}}},Y_{t_{\text{end}}}}(\bar{a}) \) stops with a verifier and a witness for one of the formulae in \( \Delta, \chi_0(t_{\text{end}} + 1) \), it suffices to show, using open-length induction, that \( \varphi(V_j, U_j) \) holds, for all \( j = 0, 1, \ldots, t_{\text{end}} < |t| \), where \( \varphi(V_j, U_j) \) is the formula

\[
V_j(0, 1) \land U_j(0, 1).
\]

This is because \( U_{t_{\text{end}}} \) evaluates \( \text{Eval}(V_{t_{\text{end}}}, \chi_0'(t_{\text{end}}, Y_{t_{\text{end}}})) \). Hence, by Theorem 6.26,

\[
U_{t_{\text{end}}}(0, 1) \leftrightarrow \text{Eval}(V_{t_{\text{end}}}, \chi_0'(t_{\text{end}}, Y_{t_{\text{end}}}))\end{equation}

Since \( \varphi(V_{t_{\text{end}}}, U_{t_{\text{end}}}) \) holds, we have that \( V_{t_{\text{end}}} \) evaluates \( \chi_0'(t_{\text{end}}, Y_{t_{\text{end}}}) \) and \( V_{t_{\text{end}}}(0, 1) \) is true. Hence, by the definition of the notion of verifier, \( V_{t_{\text{end}}} \) verifies \( \chi_0'(t_{\text{end}}, Y_{t_{\text{end}}}) \). As \( W_{t_{\text{end}}} \) codes the computation of \( M_{V_{t_{\text{end}}},Y_{t_{\text{end}}}}^{V_0,Y_0}[a_0 := t_{\text{end}}] \) with consistent restarts, by induction hypothesis, the machine \( M_{V_0,Y_0}(\bar{a}) \) must stop with a verifier and a witness for one of the formulae in \( \Delta, \chi_0(t_{\text{end}} + 1) \).

Let us now show that \( \varphi(V_j, U_j) \) holds, for all \( j = 0, 1, \ldots, t_{\text{end}} \). The base case follows from the fact that \( U_0 \) evaluates \( \text{Eval}(V_0, \chi_0'(0, Y_0)) \), from Theorem 6.26 and from the assumption that \( V_0 \) verifies that \( Y_0 \) witnesses \( \chi_0'(0) \). Now, let \( j < t_{\text{end}} \) and assume that \( \varphi(V_j, U_j) \) is true. We want to show that \( \varphi(V_{j+1}, U_{j+1}) \) follows. Since \( U_j \) evaluates \( \text{Eval}(V_j, \chi_0'(j, Y_j)) \), by Theorem 6.26,

\[
U_j(0, 1) \leftrightarrow \text{Eval}(V_j, \chi_0'(j, Y_j)).
\]
6. Extended Linear Local Improvement Principles and $U_2^1$

Since $U_j(0,1)$ is true by assumption, it follows that $Eval(V_j, \chi'_j(j,Y_j))$ is true. Because $V_j(0,1)$ also holds by assumption, we have that $V_j$ verifies that $Y_j$ witnesses $\chi_i(j)$, by definition. As $W_j$ codes the computation of $M^1_{V_j,Y_j}(a)$ with consistent restarts, by induction hypothesis, $V_{j+1}$ verifies $\chi'_j(j+1,Y_{j+1})$. That is to say, $V_{j+1}(0,1)$ and $Eval(V_{j+1}, \chi'_j(j+1,Y_{j+1}))$ are true. Because $U_{j+1}$ evaluates $Eval(V_{j+1}, \chi'_j(j+1,Y_{j+1}))$, by Theorem 6.26, it follows that $U_{j+1}(0,1)$ holds. Therefore, we conclude that $\varphi(V_{j+1},U_{j+1})$ is true.

\[\square\]

6.4 Extended Linear Local Improvement Principles

We write $[x,y]$ for $\{x,x+1,\ldots,y-1\}$; we write $[x]$ for $\{x,x+1,\ldots,y\}$ and, as a reminder, we write $[y]$ for $\{0,1,\ldots,y-1\}$.

**Definition 6.35** An instance of the extended linear local improvement principle consists of a specification of a directed acyclic graph $G$ on domain $\{0,1,\ldots,a-1\}$; parameters $b,c > 0$; a polynomial-time neighborhood function $f$ such that if $y \in f(x)$, then $y < x$; a polynomial-time initial labeling function $E$; a polynomial-time improvement function $I$; a polynomial-time scoring function $S$ and a polynomial-time wellformedness predicate $wf$. These satisfy the following conditions:

(i) The directed graph $G$ has vertices $[a] = \{0,1,\ldots,a-1\}$. The set of all edges in $G$ is defined as follows:

$$\{(x,x+1) : x \in [0,a-1]\} \cup \{(y,x) : y \in [a] \text{ and } y \in f(x)\}$$

The local neighborhood of a vertex $x$ in $G$ consists of $x-1, x, x+1$ (if they exist). The neighborhood of $x$ consists of the local neighbors of $x$ and the vertices in

$$\{y : y \in f(x)\} \cup f^-(x)$$

where $f^-(x) = \{y-1 : y \in f(x)\}$. The extended neighborhood of $x$ is the union of the neighborhoods of the neighbors of $x$.

(ii) Vertices in $G$ will be assigned a series of labels in $[0,b]$. For a label, we associate a score in the range $[0,c]$. The score associated with a label for a vertex $x$ is polynomial-time computable as a function of the label on $x$. The polynomial-time predicate $wf$ takes as inputs the vertices in the neighborhood of a vertex $x$ and a labeling of these vertices and decides whether that labeling is wellformed. A labeling of vertices is extended wellformed around a vertex $x$ if, for each vertex $y$ in the neighborhood of $x$, it is wellformed around $y$.

(iii) The initial labeling function $E$ is a polynomial-time function which assigns a label to every node in $G$. The improvement function $I$ is a polynomial-time computable function which takes as inputs a vertex $x$ and a wellformed labeling of the local neighborhood of $x$ and returns a new label for $x$ with a higher score.
Assuming that (i), (ii) and (iii) are satisfied, then the following cannot all be true:

(a) The function \( E \) assigns a label to every vertex \( x \) in \( G \) with score 0 so that all neighborhoods are wellformed.

(b) Let \( x \) be a vertex in \( G \) and \( w \) be a labeling of the extended neighborhood of \( x \) such that \( w \) is extended wellformed around \( x \). Suppose that \( x - 1 \) has a label (with respect to \( w \)) with score \( 2s + 1 \) and \( x \) and its successor \( x + 1 \) have labels (with respect to \( w \)) with scores \( 2s \). Then the improvement function \( I \) (whose inputs consist of \( x \) and the labeling of the local neighborhood of \( x \) extracted from \( w \)) provides a new label for \( x \) with score \( 2s + 1 \) and the labeling \( w' \) obtained by replacing the labeling of \( x \) in \( w \) with the new label given by \( I \) remains extended wellformed around \( x \).

(c) Let \( x \) be a vertex in \( G \) and \( w \) be a labeling of the extended neighborhood of \( x \) such that \( w \) is extended wellformed around \( x \). Suppose that \( x \)'s successor \( x + 1 \) has a label (with respect to \( w \)) with score \( 2s + 2 \) and \( x \) and \( x - 1 \) have labels (with respect to \( w \)) with scores \( 2s + 1 \). Then the improvement function \( I \) (whose inputs consist of \( x \) and the labeling of the local neighborhood of \( x \) extracted from \( w \)) provides a new label for \( x \) with score \( 2s + 2 \) and the labeling \( w' \) obtained by replacing the labeling of \( x \) in \( w \) with the new label given by \( I \) remains extended wellformed around \( x \).

In all other cases, the function \( I \) is undefined.

Remark 6.36 The polynomial-time functions \( f, E, I \) and polynomial-time predicate \( wf \) in the definition of an instance of the extended linear local improvement principle take some hidden argument; namely, the size \( a \) of the underlying graph.

The intuition behind the definition of the extended linear local improvement function is as follows. The initial labeling function \( E \) assigns a label to every vertex in \( G \) with score 0 such that all neighborhoods are wellformed. At the end of the first forward pass (forward pass is the update of scores from even to odd), every node in \( G \) will have a label with score 1 such that all neighborhoods are wellformed. Similarly, at the end of the first backward pass (backward pass is the update of scores from odd to even), every node in \( G \) will have a label with score 2 such that all neighborhoods are wellformed. Because extended wellformedness is always preserved, scores can always be improved without a bound. Hence, contradicting the property that score values are less than or equal to \( c \).

Definition 6.37 A solution to an instance of the extended linear local improvement principle consists of either (a) a neighborhood of a vertex \( x \) where the initialization function \( E \) fails to provide an extended wellformed labeling with scores all equal to zero, or (b) an extended wellformed labeling of a vertex \( x \) and its extended neighborhood where the local improvement function is defined but fails to provide a new label for \( x \) with the correct score value that preserves the extended wellformedness property.

Definition 6.38 An instance of the extended linear local improvement principle is formalized in bounded arithmetic by first-order values \( a, b \) and \( c \); by a polynomial-time function \( f \) such
that if \( y \in f(a, x) \), then \( y < x \); by polynomial-time functions \( E, I \) and \( S \) and polynomial-time predicate \( w f \); it also consists of the \( \Sigma^b_1 \)-formula (with free variables \( a, b \) and \( c \)) which states that a solution exists.

\( \text{ELLI} \) is the set of \( \Sigma^b_1 \)-formulae obtained from all instances of the extended linear local improvement principle. \( \text{ELLI}_{\log k} \) is \( \text{ELLI} \) where \( c = |c'|_k \) (that is to say, \( k \) applications of the binary length function to \( c' \)), for some term \( c' \). \( \text{ELLI}_k \) is defined similarly, but now with \( c = k \).

Kolodziejczyk, Nguyen and Thapen [KNT11] introduced the linear local improvement principle, which is essentially the extended linear local improvement principle, except that now the following set is empty:

\[
\{ (y, x) : y, x \in [a] \text{ and } y \in f(x) \}.
\]

The linear local improvement principle \( \text{LLI} \) is then defined to be the set of \( \Sigma^b_1 \)-formulae obtained from all instances of the linear local improvement principle; \( \text{LLI}_{\log} \) is \( \text{LLI} \) but with the bound \( c \) on scores equals \( |c'| \), for some term \( c' \).

Definition 6.39 ([Bus86]) A formula \( \phi \) is \( \Delta^b_1 \), if there is a \( \phi \in \Sigma^1_1 \) and a \( \psi \in \Pi^1_1 \) such that the following conditions hold: \( U_{1^2} \vdash \phi \leftrightarrow \phi \) and \( U_{1^2} \vdash \phi \leftrightarrow \psi \).

Theorem 6.40 \( U_{1^2} \) proves \( \text{ELLI}_{\log} \).

Proof Sketch. \( U_{1^2} \) can compute the set of all labels of scores 0, 1, \ldots, \( c \) using the improvement function as in [KNT11, Lemma 14] because, except for \( w f \), \( \text{ELLI}_{\log} \) and \( \text{LLI}_{\log} \) are equivalent. Once these are defined, then \( \Delta^1_{1^b} \)-IND can be used to show that all extended neighborhoods are wellformed.

Definition 6.41 A function \( f \) dominates another function \( g \) if \( g(x) \leq f(x) \), for all \( x \).

Corollary 6.42 For any function \( f \) and \( g \), if \( f \) dominates \( g \), then \( \text{ELLI}_k \) is many-one reducible to \( \text{ELLI}_f \).

By Theorem 6.40 and Corollary 6.42, we have the following corollary:

Corollary 6.43 \( U_{1^2} \) proves \( \text{ELLI}_{\log^d} \), for all \( d \geq 1 \). Furthermore, \( U_{2^1} \) proves \( \text{ELLI}_k \), for all \( k > 0 \).

6.4.1 Proof of the Main Theorem

Theorem 6.44 (Main Theorem) Suppose that \( \phi \) is a \( \Delta^b_0 \)-formula and \( (\forall x)(\forall X)(\exists y) \phi(x, X, y) \) is provable in \( U_{1^2} \). Then there is a many-one reduction from \( (\exists y) \phi(x, X, y) \) to an instance of \( \text{ELLI}_m \), for some \( m \in \mathbb{N} \), provable in \( S_{2^1} \).

The constant \( m \) in the statement of Theorem 6.44 will be defined during the course of the proof of the theorem.

Kołodziejczyk, Nguyen and Thapen [KNT11] proved a version of Theorem 6.44: they showed that the \( \forall \Sigma^b_1 \)-theorems of \( U_{2^1} \) are many-one reducible to \( \text{LLI}_{\log} \), provable in \( S_{2^1} \).
From Theorem 6.44, Lemma 6.42 and Corollary 6.43, we obtain the following characterizations of the $\forall \Sigma_1^b$-theorems of $U_2^1$.

**Theorem 6.45** For all $d \geq 1$, $\text{ELLI}_{\log, d}$ is many-one complete for the class of all $\forall \Sigma_1^b$-theorems of $U_2^1$, where the reduction is provable in $S_2^1$. Furthermore, for some $m \in \mathbb{N}$ and all $k \geq m$, $\text{ELLI}_k$ is $\Pi$-equivalent to $\text{ELLI}_{\log, k}$, provable in $S_2^1$.

Our above characterizations of the $\forall \Sigma_1^b$-theorems of $U_2^1$ extend the results of [KNT11], who showed that $\text{LLI}_{\log}$ is many-one complete for the $\forall \Sigma_1^b$-theorems of $U_2^1$, where the reduction is provable over $S_2^1$.

**Proof of Theorem 6.44.** The proof technique that we use for this proof is an amalgamation of the proof technique used by Kołodziejczyk, Nguyen and Thapen for Theorem 16 of [KNT11], and Beckmann and Buss, for Theorem 5.14 of [BB14].

By the witnessing theorem for $U_2^1$ (Theorem 6.33), let $M = (M, f, \text{cons})$ be a PSPACE oracle Turing machine with consistent restarts such that $WT$ proves:

(A) “Suppose that $Y$ codes a complete computation of $M^X(x)$ with consistent restarts. Then $\phi(x, X, \text{out}(Y))$ is true.”.

Assume that the input $x$ is on a read-only tape of $M$. Let $Y = (C_0, C_1, \ldots, C_{T(n)})$, where $T(n)$ is the runtime of $M^X(x)$; here $n$ is the length of $x$ and $C_t$ codes the $t$-th configuration of $M^X(x)$. Then, writing out (A) in more detail gives that the theory $WT$ proves (B), which is:

Either there is a $y$ such that $y = \text{out}(Y) \land \phi(x, X, y)$ is true, or there is a $t < T$ such that either:

(B.1) $t = 0$ and $C_0$ is not consistent with $M$’s initial configuration on input $x$ and oracle $X$, or
(B.2) $t > 0$ and $C_t, C_{t+1}$ violate $M$’s transition function, or
(B.3) there is a $t' \in f(x, t)$ such that $\text{cons}(x, t', t, C_r, C_t)$ is false.”.

Without loss of generality, assume that (B) is given by a $s\Sigma_2^b$-formula. By the witnessing theorem for $WT$ (Theorem 6.21), let $g$ be a polynomial-time function and $\phi(q, x, X, Y)$ be a $s\Sigma_2^b$-comprehension oracle based on a $\Sigma_0^b$-formula $\psi$ and a term $s$ such that $g$ makes $O(1)$-many queries to $\phi$ and $WT$ proves:

(C) “If $z$ codes the complete computation of $g^{\phi(x, X, Y)}(x)$, then $\text{out}(z)$ witnesses one of the disjuncts in (B).”.

Note that with a single reply $r$ to a query $q$ to $\phi$, polynomially-many queries to $X$ and $Y$ are answered all at once during $\phi$’s evaluation. Hence, if we were to simulate a computation $g^{\phi(x, X, Y)}(x)$ by a function which only takes as oracles $X$ and $Y$, then it would require polynomially-many queries to $X$ and $Y$ in order to obtain $r$. But then these queries to $X$ and $Y$ are predetermined by the form of $\phi$ and the query $q$. 

101
6. Extended Linear Local Improvement Principles and $U^1_2$

From now on, we reason within $S^1_2$, since we require that the many-one reduction from $(\exists y)\phi(x,X,y)$ to an instance of ELLI$_m$ is provable in $S^1_2$, for some constant $m > 0$. Since WT proves (C) and WT is contained within $S^1_2$, it follows that $S^1_2$ proves (C). Without loss of generality, assume that $g$ is computed by a clocked Turing machine, so $S^1_2$ proves that its runtime is polynomially bounded. Also, without loss of generality, $S^1_2$ proves that if $g$ outputs a value $t$ satisfying one of (B.1), (B.2) or (B.3), then $g$ must have queried all the necessary information. This can be done by defining a new polynomial-time function $h$, which will take $\phi$ as its oracle but also another $s\Sigma^b_1$-comprehension oracle $\phi'(q,t,x,Y)$ will make sure that all those necessary information are asked. So, $h^{\phi,\phi'}(x)$ first runs $g^\phi(x)$. Now, for instance assume that $g^\phi(x)$ outputs a $t$ such that there is a $t' \in f(x,t)$ and $\text{cons}(x,t',C_{t'},C_t)$ is false. Then the function $h^{\phi,\phi'}(x)$ will continue its computation by querying $\phi'$. This single query to $\phi'$ would query $C_{t'}$ and $C_t$, for all $t' \in f(x,t)$. Note that $\phi(q,x,X,Y)$ and $\phi'(q,t,x,Y,Y)$ can be easily combined into one single $s\Sigma^b_1$-comprehension oracle $\Omega(q,t,x,Y,Y)$.

Because we are mostly interested in those queries to $Y$ in the following argument, for the sake of simplicity, assume that $\phi$ is of the form

$$(\forall y \leq |u(x)|)\phi_0(q,x,X,Y,y),$$

where $\phi_0$ is a quantifier-free formula. Also, without loss of generality, assume that $l$ is the total number of queries asked by $g$ to $\phi$ at the end of $g^\phi(x)$’s computation. Note that, because $\phi_0$ is a quantifier-free formula, there can only be $k$-many queries $q^\phi_1(x,y), \ldots, q^\phi_l(x,y)$ to $Y$ in $\phi_0$, for some constant $k$ depending on $\phi_0$. Therefore, a query $q$ to $\phi$ corresponds to polynomially-many queries to $Y$, because $\phi$ is sharply bounded.

Let us now give a brief overview of the rest of the proof before we proceed with the details. The idea is to simulate a run of $g^\phi(x)$ using an instance of the ELLI$_m$ problem, where $m$ will be defined later. The intent is that from a solution to the ELLI$_m$ problem, we can extract a sequence of queries and replies $\tilde{q}, \tilde{r}$ such that if we use $\tilde{q}, \tilde{r}$ to define an oracle $A_{\tilde{q},\tilde{r}}$ (which we define later as well), then the output of $g$ on input $x$ and oracle $\phi'$, where $\phi'$ is $\phi$ with $Y$ replaced by $A_{\tilde{q},\tilde{r}}$, does not witness the second disjunct in (B), provably in $S^1_2$. Hence, the only possibility for a solution is a value $y$ such that $\phi(x,X,y)$ is true. This will suffice to prove this theorem.

We are now ready to define the instance of the ELLI$_m$ problem.

**The graph.** The underlying graph $G$ has domain $[T] = \{0,1,\ldots,T-1\}$ and the edges are specified by $f(x,t) \cup f^{-}(x,t)$ in addition to $\{(t,t+1) : t \in [T-1]\}$, where $f^{-}(x,t)$ is again $\{y-1 : y \in f(x,t)\}$.

**The scores and the labels.** Set the bound on the score to be $c = 2l+1$, that is to say, one more than twice the number of queries that $g$ makes to $\phi$. Hence, $m = 2l+1$. The vertices of $G$ will be labeled with sequences. Initially, the initial labeling function $E$ labels every node $t$ in $G$ with $\langle \rangle$. The empty sequence is the only label with score 0.
The first forward pass corresponds to the simulation of the full computation of the PSPACE machine $M^X(x)$ with consistent restarts. During this pass, every vertex $t$ in $G$ with score 0 is relabeled with a new label $\langle \beta_t \rangle$ with score 1, where $\beta_t$ is what the graph locally believes the value of $C_t$ to be.

For the first backward pass, we calculate the first query $q_1$ to $\phi$ that arises during the computation of $g^\phi(x)$. Let $q_1^\gamma = \langle q_{1,1}^\gamma, q_{1,2}^\gamma, \ldots, q_{1,p}^\gamma \rangle$ be the sequence of polynomially-many queries to $Y$ that corresponds to $q_1$. Then we load $q_1$ and $q_1^\gamma$ into the label $\langle \beta_{T-1} \rangle$ of $T - 1$; that is, we update the label of $T - 1$ from $\langle \beta_{T-1} \rangle$ to $\langle \beta_{T-1}, \langle q_1, q_1^\gamma \rangle \rangle$ and pass through the graph backwards, replacing each label $\langle \beta_r \rangle$ with the new label $\langle \beta_r, \langle q_1, q_1^\gamma \rangle \rangle$. Therefore, after the first backward pass, every vertex $t$ in $G$ will be labeled with $\langle \beta_t, \langle q_1, q_1^\gamma \rangle \rangle$ with score 2.

For the next forward pass, we want to calculate the reply $r_{1,i}$ to $q_{1,i}$, for all $i$ from 1 to $l'$. Suppose that $q_{1,i}^Y = \langle t, p \rangle$. First, we define

$$P_t = \{ t, t+1, \ldots, T-1 \}.$$

For a vertex $t'$ in $G$, if $t' = t$, then $\beta_t$ is already available in the label of $t'$. Therefore, we define $r_{1,i}^Y = \beta_i(p)$. If $t' \in P_t \setminus \{ t \}$, then $r_{1,i}^Y = \beta_i(p)$ is already available. $t'$ is the $t$-1's label. Therefore, we merely copy $r_{1,i}^Y = \beta_i(p)$ into the label of $t'$. Otherwise, $r_{1,i}^Y = \infty$. Let $r_t^Y = \langle r_{1,1}^Y, \ldots, r_{1,p}^Y \rangle$. By the end of this forward pass, every vertex in $G$ will have a label of the form

$$\langle \beta_t, \langle q_1, q_1^\gamma \rangle, r_t^Y \rangle$$

with score 3. Note that, at vertex $T - 1$, every $q_{1,i}^Y$ is answered; that is to say, $r_{1,i}^Y$ is different from $\infty$. Also note that from $r_t^Y$ we can compute in polynomial-time the answer $r_t$ to $q_1$.

For the next backward pass, we calculate the second query $q_2$ to $\phi$, which arises during the computation of $g^\phi(x)$, and the $q_{1,i}^Y$ values that correspond to $q_2$ and add these to the labels in the same way as in the first backward pass. Additionally, we carry out an extra task. For a reply $r_{1,i}^Y = \infty$ in the label of a vertex $t$, we replace it with the corresponding reply at vertex $t + 1$. Note that this corresponding reply at vertex $t + 1$ is different from $\infty$.

This process carries on for another $2(l - 1)$-many passes. Hence, in total, we have $(2l + 1)$-many passes. By the end of the $(2l + 1)$-many passes, every node in $G$ will have a label of the form

$$\langle \beta_t, \langle q_1, q_1^\gamma \rangle, r_t^Y, \ldots, \langle q_i, q_i^\gamma \rangle, r_i^Y \rangle$$

with score $2l + 1$ and every query $q_i$ to $\phi$ that arises during the full computation of $g^\phi(x)$ is loaded. Thus, at this point, there is no more query to load. As a result, the label at vertex $T - 1$ will be replaced by $b$ and we stop. Here, $b$ is a bound on the labels; that is to say, all the labels that are generated (excluding $b$) during this back and forth process are $< b$.

In what follows, we omit the parameter $x$.

The wellformedness property. Let us first fix some notations. For a vertex $t$ in $G$, let $\text{nbh}(t)$ denote the neighborhood of $t$, that is to say,

$$\text{nbh}(t) = \{ t-1, t, t+1 \} \cup f(t) \cup f^-(t).$$

103
Let $enbh(t)$ denote the extended neighborhood of $t$, that is to say,

$$enbh(t) = \bigcup_{j \in nbh(t)} nbh(j).$$

The wellformedness predicate takes as inputs the vertices in $nbh(t)$ and their labels. Let the labels of the vertices in $nbh(t)$ be given by $L : nbh(t) \rightarrow [b]$ and let $C$ be a partial function from $[T]$ to $[b]$ such that

$$C(j) = \begin{cases} \beta_j & \text{if } L(j) = (\beta_j, \ldots) \\ \uparrow & \text{if } L(j) = () \text{ or } L(j) = \uparrow \end{cases} \quad (6.7)$$

where $\uparrow$ is the symbol for undefined. Then $L$ is wellformed around $t$ if, and only if, the following sets of conditions are satisfied:

C1. The first set of these conditions is about the $\beta_j$ values. For these, we require that $C$ is consistent with $M$, that is to say,

$$\langle \forall i, i+1 \in dom(C) \rangle [C(i) \rightarrow_M C(i+1)]$$

and $\langle \forall j \in f(t) \rangle [cons(j, t, C(j), C(t))]$ is true, where the notation $C(i) \rightarrow_M C(i+1)$ means that $C(i)$ and $C(i+1)$ code two consecutive configurations of $M^X(x)$.

C2. We require that the labels in the neighborhood of $t$ agree on the $q_i, q_j^Y$ and $r_i^Y$ values. This means that the values of $q_i, q_j^Y$ must be the same in all labels in the neighborhood of $t$. As for the $r_i^Y = (r_i^Y, \ldots, r_i^Y, \ldots)$ values, if a non-? value for $r_i^Y$ appears in a label, then there cannot be a non-different non-? value for $r_i^Y$ in any label in the neighborhood of $t$. Furthermore, $q_i^Y$ and $r_i^Y$ (if both present) must have the same number of entries.

C3. For the labels on vertices in the neighborhood of $t$, we require that each $q_j$ must be the $j$-th query made during the computation of $g^\phi(x)$ when the entries of $r_i^Y$ are used to come up with an answer. Additionally, the entries in $q_j$ must be the queries to $Y$ when $q_j$ is the query to $\phi$. This last check is done in polynomial-time given $q_j, x$ (remember that $\phi$ is of the form $\langle \forall y \leq |x| \rangle \phi_0(q, x, X, Y)$, where $\phi_0$ is a quantifier-free formula).

C4. Let $t', t'' \in nbh(t)$. Suppose that $C(t') = \beta_{t'}$ and $(q_i, q_i^Y)$ is an entry in $L(t'')$ such that there is an entry $q_{i,j}^Y = (t', p')$ in $q_i^Y$. Additionally, suppose that $r_i^Y$ is present in $L(t'')$. Then $r_i^Y$ must satisfy the following: if $r_i^Y$ is the last entry in $L(t'')$ and $t''$ is not reachable from $t'$ (that is to say, $t'' < t'$), then $r_{i,j}^Y = ?$; otherwise, $r_{i,j}^Y = \beta_{t'}(p')$.

C5. Let $t' \in nbh(t)$. Then there should not be any sequence of queries and replies in $L(t')$ that would witness “there is a $t_0 < T$ such that (B.1), (B.2) or (B.3) in (B) holds”.

Note that a labeling $L : enbh(t) \rightarrow [b]$ is extended wellformed around $t$ if, and only if, $\forall j \in nbh(t), L|_{nbh(j)}$ is wellformed, where $L|_{nbh(j)}$ is $L$ restricted to $nbh(j)$. 

104
6.4. Extended Linear Local Improvement Principles

The improvement function

**Increasing scores from 0 to 1.** This case corresponds to updating the label at vertex $t$ from $\langle \rangle$ to $\langle \beta_t \rangle$. If $t = 0$, then $\beta_t$ is just the initial configuration of $M^X(x)$. Suppose that $t > 0$, then $\beta_t$ is computed from $\beta_{t-1}$ using $M$'s transition function.

**Claim 6.46** Suppose that $L : \text{enbh}(t) \to [b]$ is extended wellformed around $t$ and

$$
(\forall j \in \text{enbh}(t) \cap [t]) [L(j) = \langle \beta_j \rangle] \text{ and } (\forall j \in \text{enbh}(t) \cap [t]^c) [L(j) = \langle \rangle] \text{ are true, where } [t]^c \text{ is the complement of } [t].
$$

Let $\beta_t$ be such that $\beta_t \rightarrow_M \beta_t$ and define

$$L'(j) = \begin{cases} L(j) & \text{if } j \neq t \\ \langle \beta_t \rangle & \text{if } j = t \end{cases}$$

Then $L'$ is extended wellformed around $t$.

**Proof of Claim 6.46.** Let $C$ correspond to $L$ (in the sense of (6.7)) and $C'$ to $L'$. Then $C' = C \cup \{(t, \beta_t)\}$. Now, to show the claim, it is enough to show the following:

(i) $C'_{\text{nbh}(t)}$ is consistent with $M^X(x)$, and

(ii) $\langle \forall j \in f(t) \rangle [\text{cons}(j, t, C'(j), C'(t))]$ hold.

The first part (i) is straightforward. Therefore, we focus on showing the second part (ii). Because $L$ is extended wellformed around $t$, therefore,

$$L'' = L_{\text{nbh}(t-1)}$$

is wellformed around $t - 1$. Note that $C'_{\text{nbh}(t-1)}$ corresponds to $L''$. Thus, by the definition of wellformedness,

$$\langle \forall j \in f(t-1) \rangle [\text{cons}(j, t - 1, C(j), C(t - 1))]$$

holds. Now, consider $\bar{C}$, which is defined to be

$$\bar{C} = C'_{\text{nbh}(t-1)} \cup \text{f}(t)$$

To see that $\bar{C}$ is consistent with $M$, let $i, i + 1 \in \text{dom}(\bar{C})$. We need to show that $\bar{C}(i) \rightarrow_M \bar{C}(i + 1)$. There are three cases to consider:

1. If $i = t - 1$, then $\bar{C}(i) = \beta_{t-1} \rightarrow_M \beta_t = \bar{C}(i + 1)$.

2. If $i + 1 \in f(t - 1)$, then $i \in f^-(t - 1)$. Thus, $i, i + 1 \in \text{nbh}(t - 1)$, and $\bar{C}(i) \rightarrow_M \bar{C}(i + 1)$ follows from the fact that $L''$ is wellformed around $t - 1$.

3. The case when $i + 1 \in f(t)$ is similar to the previous case.
For $j \in f(t - 1)$, we compute
\[ \bar{C}(j) = C'(j) = C(j). \]

Also,
\[ \bar{C}(t - 1) = C'(t - 1) = C(t - 1). \]

Therefore,
\[ (\forall j \in f(t - 1))[\text{cons}(j, t - 1, \bar{C}(j), \bar{C}(t - 1))] \]
holds. By the definition of PSPACE oracle Turing machine with consistent restarts, we have that the following holds:
\[ (\forall j \in f(t))[\text{cons}(j, t, \bar{C}(j), \bar{C}(t))]. \]

For $j \in f(t)$, we have that $\bar{C}(j) = C'(j)$ and $\bar{C}(t) = C'(t)$. Thus, we have that
\[ (\forall j \in f(t))[\text{cons}(j, t, C'(j), C'(t))] \]
holds, which finishes the proof of (ii) and, thus, the Claim. 

Increasing scores from $2s + 1$ to $2s + 2$. This corresponds to loading queries $\langle q_{s+1}, q_{s+1}^Y \rangle$ into the labels (if there are). There are two cases to consider. First consider the case when we update the label
\[ \langle \beta_{T-1}, \langle q_1, q_1^Y \rangle, r_1^Y, \ldots, \langle q_s, q_s^Y \rangle, r_s \rangle \]
at vertex $T - 1$. For this case, the improvement function simulates $g^\phi(x)$ in the following way. Whenever $g$ makes its $i$-th query to $\phi$, with $i < s + 1$, this query equals $q_i$ and the improvement function $I$ uses the $r^Y_i$ values during the evaluation of $\phi$. More precisely, the value of $Y(q^Y_{i,j})$ in $\phi$ is set to be $r^Y_{i,j}$ during $\phi$’s evaluation. If there is a $q_{s+1}$ query, then $I$ merely loads $\langle q_{s+1}, q_{s+1}^Y \rangle$ into $(T - 1)$’s label so that $(T - 1)$’s new label is
\[ \langle \beta_{T-1}, \langle q_1, q_1^Y \rangle, r_1^Y, \ldots, \langle q_s, q_s^Y \rangle, r_s, \langle q_{s+1}, q_{s+1}^Y \rangle \rangle. \]

Otherwise, if $g$ halts without making any new query to $\phi$, then the improvement function changes the label at $T - 1$ to $b$. Now, note that evaluating $\phi$ is done in polynomial-time, since $\phi$ is a sharply-bounded formula. For any other vertex $t < T - 1$, the improvement function just simply propagates $\langle q_{s+1}, q_{s+1}^Y \rangle$ across the graph. This is done by simply copying $\langle q_{s+1}, q_{s+1}^Y \rangle$ from the label at vertex $t + 1$. Additionally, for a vertex $t < T - 1$, if an entry $r^Y_{s,j}$ in $r^Y_s$ is equal to $\text{?}$, then the improvement function replaces it with the corresponding entry at vertex $t + 1$. Note that by the end of this backward pass, all entries in $r_s$ are now answered and, therefore, different from $\text{?}$.

Increasing scores from $2s$ to $2s + 1$. This pass corresponds to calculating the replies $r^Y_{s+1}$ to the queries $q^Y_{s+1}$ and the way the improvement function works is explained in the paragraph “The scores and labels”.

This finishes the definition of the instance of $\text{ELLI}_m$.

Now, arguing in $S^1_1$, suppose that we have a solution to this instance. There are three possible ways for a solution to the above instance of $\text{ELLI}_m$ to be recorded:
6.4. Extended Linear Local Improvement Principles

S1. There is a vertex \( t < T \) such that either there is a vertex \( t' \in \text{nbh}(t) \) such that \( S(E(t')) \neq 0 \) or the labeling of the neighborhood of \( t \) (according to \( E \)) is not wellformed.

S2. There is a vertex \( t < T \) and a labeling \( L \) of the extended neighborhood of \( t \) such that \( L \) is extended wellformed around \( t \) and \( S(L(t-1)) = 2s + 1 \) and \( S(L(t)) = S(L(t+1)) = 2s \), but either \( S(I(t, L(t-1), L(t), L(t+1))) \neq 2s + 1 \) or \( L' \), which is

\[
L'(j) = \begin{cases} L(j) & \text{if } j \neq t \\ I(t, L(t-1), L(t), L(t+1)) & \text{otherwise}, \end{cases}
\]

is not extended wellformed.

S3. There is a vertex \( t < T \) and a labeling \( L \) of the extended neighborhood of \( t \) such that \( L \) is extended wellformed around \( t \) and \( S(L(t-1)) = S(L(t)) = 2s + 1 \) and \( S(L(t+1)) = 2s + 2 \), but either \( S(I(t, L(t-1), L(t), L(t+1))) \neq 2s + 2 \) or \( L' \), which is defined as above, is not extended wellformed.

First, S1 cannot happen, since for every \( t \in G \), we have that \( S(E(t)) = 0 \), and the labeling of all neighborhoods according to \( E \) satisfy C1-C5 trivially.

Also, S2 cannot happen. In order to see it, first consider the case when the improvement function updates the label at vertex 0. If we have originally started with a labeling \( L \) of the extended neighborhood of 0, which is extended wellformed around 0, then new labeling \( L' \) of the extended neighborhood of 0 obtained from \( L \) by replacing 0’s old labeling (with score 2s) by the one given by the improvement function trivially satisfy C1-C4. The only way where things might go wrong is for \( L' \) to not satisfy C5. However, \( L' \) must satisfy C5. This is because we are at vertex 0, where the first entry of \( L'(0) \) is \( \beta_0 \). By wellformedness, \( \beta_0 \) correctly encodes the initial configuration of \( M^X(x) \). We use Claim 6.46, when the improvement function updates the label at a vertex \( t > 0 \).

Having considered S1 and S2, the only possibility is S3. This corresponds to either updating the label at vertex \( T - 1 \) to \( b \) (invalid label) or loading a new query \( q_{s+1} \) into the labeling of a vertex \( t \) and, also, tidying up the entry \( r^t_j \) at vertex \( t \) (that is to say, replacing the entries \( r^t_j \) in \( r^t_s \), that are equal to ?, by the corresponding entries in the labeling of \( t + 1 \)). If \( t < T - 1 \), then it is easy to see that nothing can go wrong. Hence, the only place where something can go wrong is at vertex \( T - 1 \). The first case is that the improvement function loads a new query \( q_{s+1} \) (note that the improvement function does not need to tidy up \( r_s \), because, at this point all entries in \( r_s \) are guaranteed to have been answered). As before, nothing can go wrong for this case. The only way for things to go wrong is when the improvement function updates the label at vertex \( T - 1 \) to \( b \) (invalid label). Hence, the labeling \( L \) of the extended neighborhood of \( T - 1 \) (that is extended wellformed around \( T - 1 \) before the update of \( T - 1 \)’s label to \( b \)) and \( T - 1 \) constitute a solution. Note that

\[
L(T - 1) = \langle \beta_{T-1}, (q_1, q_1^T), (q_2, q_2^T), \ldots, (q_i, q_i^T), r^T_{T-1} \rangle,
\]

where all entries in \( r^T_{T-1} \) are all answered (hence, different from ?) and all the queries to \( \phi \) by \( g \) are loaded. By the extended wellformedness of \( L \), there is no sequence of queries and replies
6. Extended Linear Local Improvement Principles and $U^1_2$

in $L(t')$, for all $t' \in \text{enbh}(T - 1)$, that witness the second disjunct in (B). Because $S^1_2$ proves that $g^{(q \mathbin{X} \bar{A} \mathbf{q}, \mathbf{r})}(x)$ witnesses one of the disjuncts in (B) (see (C)), where $A_{\mathbf{q}, \mathbf{r}}$ is defined as follows:

$$A_{\mathbf{q}, \mathbf{r}}(q^Y_{i,j}) = \begin{cases} r^Y_{i,j} & \text{if } q^Y_{i,j} \in \bar{q} = q^Y_1 \mathbin{\&} q^Y_2 \mathbin{\&} \ldots \mathbin{\&} q^Y_l \\ 0 & \text{otherwise,} \end{cases}$$

it follows that the only possibility is for $g^{(q \mathbin{X} \bar{A} \mathbf{q}, \mathbf{r})}(x)$ to output a value $y$ that witnesses $(\exists y)\varphi(x, X, y)$. Thus, this completes the proof. \qed
Conclusion

In this thesis, we provided characterizations of the provably total search problems in the theories corresponding to complexity classes from \( \text{AC}^0(m) \) to \( \text{PSPACE} \) in terms of subclasses of TFNP. Furthermore, we formulated improved new-style witnessing theorems for these theories and introduced novel classes of total search problems that lie within TFNP: the class KPTC, the class IITER of inflationary iteration problems and the extended linear local improvement principle.

For the theories corresponding to complexity classes from \( \text{AC}^0(m) \) to \( \text{PH} \), our characterizations of their provably total search problems are in terms of subclasses of \( \forall \exists \text{AC}^0 \). Additionally, the base theory over which we formulate the new-style witnessing theorems for these theories is always \( \text{V}^0 \). However, the choice of the base theory over which one can formulate a new-style witnessing theorem for \( U_2 \) becomes more delicate. For instance, in this thesis, we prove a new-style witnessing theorem for \( U_2 \) over the base theory \( \text{WT} \), by using PSPACE Turing machines with consistent restarts (Theorem 6.33). But we know that \( \Sigma^b_0 - \text{LIND} \) is too weak to use as a base theory for a new-style witnessing theorem for \( U_2 \) that is formulated in terms of standard PSPACE Turing machines. The intuitive reason for this is as follows. Suppose that we have a new-style witnessing theorem for \( U_2 \) over \( \Sigma^b_0 - \text{LIND} \), and formulated in terms of PSPACE Turing machines (we note that, for \( d \geq 1 \), \( \Sigma^b_0 - \text{LIND} = \Sigma^b_0 - \text{LIND} \) [Bec96], where the constant \( k \) means that we can only do induction on \( \Sigma^b_0 \)-formulae up to \( |x|_k \) and \( |x|_k \) denotes \( k \) applications of the length operator to \( x \)). Then a version Theorem 6.44 with \( \text{LLI}_{\log 3} \) instead of \( \text{ELLI}_m \) would follow. By the result of Buss [Bus15], it follows that the theory \( U_2 \) proves the pigeonhole principle \( \text{PHP}^{\text{poly}} \) (expressed as a \( \forall \Sigma^b_1 \)-formula). By our assumption, the principle \( \text{PHP}^{\text{poly}} \) is many-one reducible to \( \text{LLI}_{\log 3} \). From [Tha11], it follows that \( \text{PHP}^{\text{poly}} \) has subexponential-size proof in bounded-depth Frege, which is a contradiction to the result of [PBI93]. This brings us to our first open problem:

**Problem 7.1** What is the weakest theory over which we can prove a new-style witnessing theorem for \( U_2 \)?

Problem 7.1 is stated in terms of \( U_2 \). However, one can also ask the same question for theories beyond \( U_2 \) (such as the theory \( V_2 \) for EXPTIME).
The characterizations of the provably total search problems in TV\textsuperscript{i} in the literature [ST11, Pud06, Pud08, KNT11, Tha11, PT12], among others, use \(P\)-many reduction, which implies that they are all \(P\)-equivalent. Our characterizations use \(AC^0\)-many-one reduction. Thus, it is tempting to ask the following question:

**Problem 7.2** Are the characterizations of the provably total search problems in TV\textsuperscript{i} proposed in [ST11, Pud06, Pud08, KNT11, Tha11, PT12] – including ours – all \(AC^0\)-equivalent?

Other characterizations of the provably total search problems in \(U^1_2\) (and \(V^1_2\)) appeared in [KNT11, BB14, BB, Kra15] and they all use \(P\)-many-one reduction. Now, by analogy to Problem 7.1, it would be interesting to know what is the lowest complexity class \(C\) over which these characterizations [KNT11, BB14, BB, Kra15] (including ours) are \(C\)-equivalent.

In Chapter 6, we introduced the extended linear local improvement principle and showed that, for all \(d \geq 1\) and all \(k \geq m\), where \(m\) is some constant in \(\mathbb{N}\), ELL\textsubscript{log}\textsuperscript{d} and ELL\textsubscript{k} are \(P\)-many-one complete for the \(\forall\Sigma_1^0\)-theorems of \(U^1_2\), where the reduction is provable in \(S^1_2\). However, the strength of ELL\textsubscript{1} remains open. It could be that ELL\textsubscript{1} is complete for the \(\forall\Sigma_1^0\)-theorems of \(V^1_2\) or already provable in \(U^1_2\).

Finally, the class KPTC characterizing the provably total search problems in V\textsubscript{C} is a generic \(\forall\exists AC^0\) class. It would be interesting to characterize the provably total search problems in V\textsubscript{C} in terms of a concrete class of \(\forall\exists AC^0\) total search problems. Examples of total search problems of this kind (that is to say, \(\forall\exists AC\) search problem) are as follows:

1. The stable marriage problem (SM) was first introduced by Gale and Shapley in 1962 [GS13] and have applications in a variety of real-world situations, perhaps the best known of these being in the pairing of medical interns with hospital residencies jobs in the USA. An instance of size \(n\) of the SM involves two sets of \(n\) men and \(n\) women. Associated with each person is a strictly ordered preference list containing all the members of the opposite sex: Person \(p\) prefers person \(q\) to \(r\) if and only if \(q\) precedes \(r\) on \(p\)’s preference list.

Given an instance of SM, a matching \(M\) is a bijection between the sets of men and women. A man \(m\) and a woman \(w\) are called partners in \(M\) if and only if they are matched in \(M\); we write \(p_M(m)\) to denote the partner of \(m\) in \(M\) (similarly for \(p_M(w)\)). A matching \(M\) is called unstable if there is a man \(m\) and a woman \(w\) such that \(m\) and \(w\) are not partners in \(M\) and \(m\) prefers \(w\) to \(p_M(m)\) and \(w\) prefers \(m\) to \(p_M(w)\). Otherwise, \(M\) is called stable.

The search task associated with SM is, given an instance of SM, find a matching that is stable. Gale and Shapley showed that such a stable matching always exists. Hence, SM is a total search problem. Moreover, the graph of SM can be expressed by a \(\Sigma_0^0\)-formula.

In order to see this, the following statements are all expressible by \(\Sigma_0^0\)-formulae:

(a) “\(P\) is a valid preference list for \(n\) men and \(n\) women”;

110
(b) “$M$ is a valid matching between $n$ men and $n$ women”;
(c) “$M$ is a stable matching for an instance $(n, P)$ of SM”.

Let $ValidList(a, P), ValidMatching(a, M)$ and $Stable(a, P, M)$ be some $\Sigma^B_0$-formulae which express (a), (b) and (c), respectively. Then the totality of SM is expressed as follows:

$$\forall a \forall P \exists M [ ValidList(a, P) \supset ValidMatching(a, M) \land Stable(a, P, M) ].$$

(7.1)

2. Let $G = (V, E)$ be a graph. An independent set in $G$ is a set of vertices, no two of which are adjacent. A maximal independent set $S$ in $G$ is an independent set such that for every vertex $v$ in $V$, either $v$ belongs to $S$ or has at least one neighbor vertex that belongs to $S$. Note that any neighbor to a vertex in $S$ cannot be in $S$.

The maximal independent set problem (MIS) is the following computational problem: Given a graph $G = (V, E)$, find a maximal independent set in $G$. MIS is a total search problem, since for a given graph $G = (V, E)$, a maximal independent set $S$ is always guaranteed to exist. Note that if $V = \emptyset$, then $S = \emptyset$ is a maximal independent set. Furthermore, the graph of MIS can be expressed by a $\Sigma^B_0$-formula. This is because the statement “$U$ is a maximal independent set in a graph $G = (V, E)$” is a $\Sigma^B_0$-statement.

Let us argue why SM is an interesting example. As previously explained, SM is a total search problem. Therefore, a stable matching is always guaranteed to exist, but it may not be unique. However, there is always a unique man-optimal and woman-optimal stable matching. In the man-optimal stable matching, each man is matched with a woman whom he likes as much as any woman that he is matched with in any stable matching. The woman-optimal stable matching is defined dually to man-optimal stable matching. The man-optimal stable marriage decision problem (MOSM) is then defined as follows: given an instance of SM and a designated man-woman pair, determine whether that pair is married in the man-optimal stable marriage. The woman-optimal stable marriage decision problem (WOSM) is defined dually to MOSM. It is shown in [LCY11] that MOSM and WOSM are complete for the complexity class CC (see [CFL14] for Definition) and that they are computationally equivalent to SM. Furthermore, it is known that CC $\subseteq$ P [MS92, CFL14]. Now, the reason why SM is interesting is because NC also satisfies NC $\subseteq$ P and that the complexity classes CC and NC are conjectured to be incomparable ([CFL14] gives a strong evidence that CC and NC are incomparable). It is shown in [LCY11] that SM is provable in the theory VCC corresponding to CC. Therefore, the theory VP proves SM, since VCC $\subseteq$ VP [LCY11].

**Conjecture 7.3** VNC does not prove SM.

The maximal independent set problem is included within this short list of examples because there are NC$^2$ algorithms that solve MIS [Lub85, ABI86]. However, it remains an open question whether it can be solved by an NC$^1$ algorithm. Thus, it would be interesting to know whether VNC$^1$ proves MIS or not.
Index

(X ∗ Y), 44
(Z)∗, see seq(x, Z)
(x, X, Y)j, 45
(w(S))2, 16
Bit(x, i), 84
POW2(x), 24
Row(x, Z), 24
S(X), 26
S2, 81
SiIter(k), 75
U2, 82
X + Y, 26
X < Y, 27
X ≤ Y, 44
X ≤ Y, 27
Z(k), see Row(x, Z)
[a], 93
[x, y], 98
AC0-closure, 25
AC0-reducibility, 25
AC0(m), 17
ACk, 16
BASIC, 80
C-equivalent, 29
FC, 18
FAC0-closure, 25
IITER, 42

IPLS, 41
KPTC, 34
L, 17
LK, 14
LK-Φ proof, 14
LK-TV1, 70
LK-∀1, 53
structural rules, 15
LLI, 100
LLI log, 100
NC, 16
NCk, 16
NL, 17
NP, 17
NPΣp, see Σp
OPEN(L), 13
P, 17
PH, 17
PLS, 28
PSPACE, 79
Φ |= φ, see logical consequence
Φ-COMP, see comprehension axiom
Φ-IND, see number induction scheme
Φ-MAX, see number maximization scheme
Φ-MIN, see number minimization scheme
Φ-bit definable, see Φ-bit definable from L
Φ-bit definable from L, 21
INDEX

\( \Phi \)-definability in \( \mathcal{J} \), 20
\( \Phi \)-definable search problem in a theory, 29
\( \Pi^B_0 \), see \( \Pi^B_0(\mathcal{L}) \)
\( \Pi^B_k(\mathcal{L}) \), see \( \Sigma^B_k(\mathcal{L}) \)
\( \Pi^B_k(\mathcal{L}) \), 13
\( \Pi^B_{\mathcal{L}} \), 79
\( \Pi^B_{\mathcal{L}} \), 82
2BASIC, 19
SPC, 30
\( \Sigma^B_0 \), 17
\( \Sigma^B_1 \), see \( \Sigma^B_1(\mathcal{L}) \)
\( \Sigma^B_1(\mathcal{L}) \), 13
\( \Sigma^B_2 \), see \( \Sigma^B_2(\mathcal{L}) \)
\( \Sigma^B_2(\mathcal{L}) \)-definable from \( \mathcal{L} \), 21
\( \Sigma^B_2^\mathcal{L} \), 13
\( \Sigma^B_{\mathcal{L}} \), see \( \Sigma^B_{\mathcal{L}}(\mathcal{L}) \)
\( \Sigma^B_{\mathcal{L}} \)-SMAX, see string maximization axiom
\( \Sigma^B_{\mathcal{L}} \)-SIND, see string induction axiom
\( \Sigma^B_j(\mathcal{L}) \), 13
\( \Sigma^B_{k+1} \)-ITER
formalizable, 62
\( \Sigma^B_{k+1} \)-ITER, 61
Skolemizable, 73
with goals, 62
\( \Sigma^B_{k+1} \)-PLS, 61
with goals, 61
\( \Sigma^B_k \), 79
\( \Sigma^B_j \), 18
\( \Sigma^B_j \), 82
\( \Sigma^B_j \), 82
\( \Sigma^B_k \), 17
TFNP, 28
\( \Sigma^B_k \), 64
TV, 26
TV\(^{\infty} \), see V\(^{\infty} \)
VC, 26
\( V_0 \), 19
\( V_1 \), 26
\( V^{\infty} \), 27
ELLI, 100
ELLI\(_k \), 100
ELLI\(_{\log} \), 100
\( \emptyset \), 26
\( \forall \Phi \), see universal closure
\( \forall \exists \AC^0 \), 28
\( \forall \varphi \), see universal closure
\( \mathcal{L}_S \), 78
\( \mathcal{L}_\varphi \), 22
\( \forall^1 \), 53
\( \forall^0 \), 23
\( \tilde{\Pi}^B_k \), see \( \tilde{\Pi}^B_k(\mathcal{L}) \)
\( \tilde{\Pi}^B_k(\mathcal{L}) \), 63
\( \tilde{\Pi}^B_k \), see \( \tilde{\Pi}^B_k(\mathcal{L}) \)
\( \tilde{\Pi}^B_k(\mathcal{L}) \), 64
\( \tilde{\Sigma}^B_k \), see \( \tilde{\Sigma}^B_k(\mathcal{L}) \)
\( \tilde{\Sigma}^B_k(\mathcal{L}) \), 63
\( \tilde{\Sigma}^B_k \), see \( \tilde{\Sigma}^B_k(\mathcal{L}) \)
\( \tilde{\Sigma}^B_k(\mathcal{L}) \), 64
WT, 84
\( j \in n \), 93
\( s \prec s' \), 88
\( \text{seq}(x, Z) \), 25
\( w(S) \), 16
\( x \vdash y \), 44
comprehension oracle, 85
aggregate function, 26
antecedent of a sequent, 14
bit definition, see bit defining axiom
bit defining axiom, 21
bit-graph, 18
bounded formula, 13
bounded number quantifier rules, 15
bounded quantifier, 12
bounded number quantifier, 12
bounded string quantifier, 13
canonical evaluation, 89
canonical verification, 89
comprehension axiom, 19
conservative extension, 23
cut formula, 15
INDEX

cut rule, 15

deduction theorem of first-order logic, 32
defining axiom, 20
equality axioms, 14
extended linear local improvement principle, 98
graph, 18
function, 18
search problem, 28

Herbrand theorem, 23

inflationary function, 40
initial sequent, 14

KPT witnessing theorem, 27

linear local improvement principle, 100
logical axioms, 14

many-one completeness, 29
many-one reduction, 29

non-logical axiom, 14
number induction scheme, 20
number maximization scheme, 20
number minimization scheme, 20

p-bounded, 18
function, 18
in \( \mathcal{T} \), 21
theory, 21

pairing functions, 24
\( \langle X, Y \rangle \), 24
\( \langle X_1, \ldots, X_{k+1} \rangle \), 24

polynomially-balanced, 28

propositional rules, 15

provably total in \( \mathcal{T} \), 20
function, 20
search problem, 29

representation theorems
\( \Sigma^0_0 \), 17
\( \Sigma^1_0 \), 17

search problem, 28

sequent, 14

sharply bounded quantifiers, 79

single-\( \Sigma^1_1 \), 53
single-\( \Sigma^0_1 \), 53

Stable Marriage Problem (SM), 110

string maximization axiom, 27

string induction axiom, 26

string quantifier rules, 16

succeedent of a sequent, 14

total search problem, 28

Turing machines with consistent restarts, 93

universal, 22
formula, 22
theory, 22

universal closure, 18

weak structural rules, 15

witness query, 85
Bibliography


[Coo75] Stephen A. Cook. Feasibly constructive proofs and the propositional calculus (pre-
liminary version). In Seventh Annual ACM Symposium on Theory of Computing

[Coo04] Stephen Cook. Theories for complexity classes and their propositional translations.

[CR79] Stephen A. Cook and Robert A. Reckhow. The relative efficiency of propositional


[Fag73] Ronald Fagin. CONTRIBUTIONS TO THE MODEL-THEORY OF FINITE-
STRUCTURES. ProQuest LLC, Ann Arbor, MI, 1973. Thesis (Ph.D.)–University
of California, Berkeley.

[Fer95] Fernando Ferreira. What are the ∃Σ1b-consequences of T1 2 and T2 2? Ann. Pure Appl.
Logic, 75(1-2):79–88, 1995. Proof theory, provability logic, and computation (Berne,
1994).


[KNT11] Leszek Aleksander Kołodziejczyk, Phuong Nguyen, and Neil Thapen. The provably

[KP90] Jan Krajiček and Pavel Pudlák. Quantified propositional calculi and fragments of

[KPT91] Jan Krajiček, Pavel Pudlák, and Gaisi Takeuti. Bounded arithmetic and the polyno-
posium on Mathematical Logic and its Applications (Nagoya, 1988).


Bibliography


