Witnessing functions in bounded arithmetic and search problems

Mario Chiari[1*] Jan Krajíček[1,2†]

Mathematical Institute[1] and Institute of Computer Science[2]
Academy of Sciences, Prague

Abstract

We investigate the possibility to characterize (multi)functions that are Σ_i^b-definable with small i ($i = 1, 2, 3$) in fragments of bounded arithmetic T_2 in terms of natural search problems defined over polynomial-time structures. We obtain the following results:

1. A reformulation of known characterizations of (multi)functions that are Σ_1^b- and Σ_2^b-definable in the theories S_2^1 and T_2^1.
2. New characterizations of (multi)functions that are Σ_2^b- and Σ_3^b- definable in the theory T_2^2.
3. A new non-conservation result: the theory $T_2^2(\alpha)$ is not $\forall\Sigma_1^b(\alpha)$-conservative over the theory $S_2^2(\alpha)$.

To prove that the theory $T_2^2(\alpha)$ is not $\forall\Sigma_1^b(\alpha)$-conservative over the theory $S_2^2(\alpha)$, we present two examples of a $\Sigma_1^b(\alpha)$-principle separating the two theories:

(a) the weak pigeonhole principle $WPHP(a^2, f, g)$ formalizing that no function f is a bijection between a^2 and a with the inverse g,

(b) the iteration principle $Iter(a, R, f)$ formalizing that no function f defined on a strict partial order $(\{0, \dots, a\}, R)$ can have increasing iterates.

Introduction

Bounded arithmetic theories are subtheories of first order arithmetic. They attempt to formalize feasible reasoning about finite structures, i.e., relations, predicates, functions and algorithms referred to in arguments are of bounded

[*] Supported by a grant of the Education Ministry of the Czech Republic.

[†] Partially supported by *US - Czechoslovak Science and Technology Program* grant # 93025, and by grant #119107 of the *AVČR*.

computational complexity: depending on the particular bounded arithmetic theory, within the polynomial-time hierarchy or its level, or even within circuit classes AC^0 or NC^1. Bounded arithmetic was remarkably successful in this attempt. The most widely studied systems of bounded arithmetic, $I\Delta_0$ of [21], $I\Delta_0 + \Omega_1$ of [23] and S_2 of [2], and S^1_2 of [2] and PV of [9], can define by bounded formulas all rudimentary functions of [28], all functions computable by a polynomial-time Turing machine with an oracle from the polynomial-time hierarchy, and all polynomial-time functions respectively.

On the other hand, bounded arithmetic theories do not define functions computationally unfeasible. Already [21] showed that a function whose graph is defined by a bounded formula and that is definable in a bounded arithmetic theory is majorized by a term of the language. Considerably finer information was obtained in [2] where classes of functions (with the graph of a particular complexity) definable in a particular subtheory of bounded arithmetic S_2 were characterized. These characterizations are known as *witnessing theorems*. Further witnessing theorems were obtained in [17, 13, 26, 14, 6, 5, 15].

Feasibility of bounded arithmetic is shown by its relation to propositional logic too, as discovered in [9] and, in a different form, in [22]. A bounded formula can be translated into a propositional formula in various ways. To give an idea of a translation we consider a rather well-known example.

Let $PHP(a, R)$ be the bounded formula:

$$\forall x < a + 1 \exists y < a R(x, y) \rightarrow \exists y < a \exists x_1 < x_2 < a + 1, R(x_1, y) \wedge R(x_2, y) \ .$$

$PHP(a, R)$ formalizes the *pigeonhole principle*. For any fixed $a := n$, the formula $PHP(a, R)$ can be translated into the propositional formula PHP_n:

$$\bigwedge_{x<n+1} \bigvee_{y<n} p_{xy} \rightarrow \bigvee_{y<n} \bigvee_{x_1<x_2<n} (p_{x_1 y} \wedge p_{x_2 y}) \ ,$$

where the propositional atom p_{xy} is in place of $R(x, y)$. The formula PHP_n is a tautology (as the pigeonhole principle is valid in all finite structures), it has $n(n-1)$ atoms and has size (the number of occurrences of symbols) $O(n^3)$. As PHP_n is a tautology, it has a proof in any complete propositional proof system. In general, however, we cannot say more, that is, we cannot say whether there is a non-trivial upper bound on the size of (the smallest) proof of a tautology (other than the exponential bound obtained if you consider truth tables as proofs.)[1] The important feature of bounded arithmetic theories is that to a theory is associated a propositional proof system with the following property: whenever a finite combinatorial principle (expressed by a bounded formula of the language of bounded arithmetic) is provable in the theory, the tautologies expressing it in a propositional form do have proofs in the associated propositional proof system of size polynomial in the size of the formula. Moreover, the propositional

[1] For the particular formula PHP_n, good lower and upper bounds are known, see [18, 25, 3].

proof systems associated to the main bounded arithmetic theories are not some artificial systems but rather natural and well studied calculi like the extended resolution proof system and various forms of Gentzen-style and Hilbert-style systems.

Thus, the question whether a combinatorial principle is unprovable in a bounded arithmetic theory is related (in fact, more closely than it is sketched above, see [15]) to the problem whether there is a superpolynomial lower bound to the size of proofs in an associated propositional proof system. The latter problem is a fundamental one in the computational complexity theory: the famous P *vs* NP problem of [8] is formulated in terms of propositional logic and with a general notion of a propositional proof system the question whether there is a propositional proof system admitting polynomial size proofs is *equivalent* to the question whether the class NP is closed under complementation (the NP *vs* $coNP$ problem), cf. [10].

The discussion above should persuade the reader that to study the unprovability of finite combinatorial principles in bounded arithmetic theories is relevant to major open questions in propositional logic and complexity theory. Perhaps the most widely considered principle in mathematical logic is the *principle of induction* $IND(a, P)$ which is formulated as a finitary principle as follows:

$$\neg P(0) \vee (\exists x < a, P(x) \wedge \neg P(x+1)) \vee P(a) \ .$$

This principle is a basis of essentially all bounded arithmetic theories and they differ by posing various restrictions on the predicate P for which the induction is adopted. For example, the theory $T_2(\alpha)$ of [2] is based on $IND(a, P)$ for the predicates P defined by bounded formulas in the language of $T_2(\alpha)$, and the subtheories $T^i_2(\alpha)$ are defined by further restricting P to the predicates definable by $\Sigma^b_i(\alpha)$-formulas, bounded formulas of a particular quantifier complexity.

Thus, to prove that $IND(a, P)$ is not available in $T^i_2(\alpha)$ for some P defined by a bounded formula is the same as to show $T^i_2(\alpha) \neq T_2(\alpha)$; i.e., to show that $T_2(\alpha)$ is not finitely axiomatizable. This is indeed so but for the theory T_2 this problem is open and closely related to the question whether the polynomial-time hierarchy collapses, see [17].

Therefore, finer information about the difference between $T^i_2(\alpha)$ and $T_2(\alpha)$ or, in other words, good non-conservation results are desirable.

In this paper we obtain a new non-conservation result: $T^2_2(\alpha)$ is not $\forall\Sigma^b_1(\alpha)$-conservative over $S^2_2(\alpha)$. Previously only the non-$\forall\Sigma^b_2(\alpha)$-conservativity was known, cf. [13, 26] (see [7] for the question how to lift this non-conservativity to a non-$\forall\Sigma^b_i(\alpha)$-conservativity of $T^{i+1}_2(\alpha)$ over $S^{i+1}_2(\alpha)$, for all $i \geq 1$).

We believe that any further improvement of this non-conservation result (to the eventual result that $T^i_2(\alpha)$ is not $\forall\Sigma^b_1(\alpha)$-conservative over $S^i_2(\alpha)$, for all $i \geq 1$, which we expect to hold) will require to find characterizations of (multi)functions that are $\Sigma^b_j(\alpha)$-definable in $T^i_2(\alpha)$, for $j \leq i$, in terms of search problems defined over polynomial-time structures (instead of characterizations involving oracle computations). A simple example is this: the task to compute a value of a $\square_2^p$-function for an argument a is equivalent to the task to find a maximal value of a polynomial-time function at an interval $[0, t(a)]$ (see Theorem 3.4).

We prove several new characterizations in terms of search problems, and we state two specific questions about search problems definable in T_2^2 (see the end of Sections 4 and 6).

The paper is organized as follows. In the first section we overview basic notions and known results of bounded arithmetic. In the second and third sections, we study the search problems that are Σ_1^b-definable in T_2^1 and the search problems that are Σ_2^b-definable in S_2^1 and in T_2^1. The fourth section is devoted to the search problems that are Σ_2^b-definable and Σ_3^b-definable in T_2^2.

In the fifth and sixth sections we show that a form of the weak pigeonhole principle and a form of the iteration principle (both $\Sigma_1^b(\alpha)$- principles) are not provable in the theory $S_2^2(\alpha)$ while they are provable in the theory $T_2^2(\alpha)$.

Although we recall notions and results from bounded arithmetic and propositional logic, we advice the reader to consult either the original papers or [15] for details. In [15] the relations of bounded arithmetic to computational complexity and to propositional proof systems are treated in depth.

1 Bounded arithmetic preliminaries

In this section we recall some definitions and results from bounded arithmetic relevant to our paper.

We study subtheories of the theory $T_2(\alpha)$. The language $L(\alpha)$ of the theory contains seven function symbols:

$$0, 1, x + y, x \cdot y, \lfloor \frac{x}{2} \rfloor, |x|, x\#y$$

with the first five having the usual meaning, $|x|$ is the length of the binary representation of x, and $x\#y$ is $2^{|x||y|}$, and three predicate symbols:

$$x = y, x \leq y, \alpha(x_1, \dots, x_k) \ ,$$

again the first two having the usual meaning and $\alpha(x_1, \dots, x_k)$ is a k-ary predicate symbol without any attached interpretation in the standard model. The arity k will vary. The language L is $L \setminus \{\alpha\}$.

The theory $T_2(\alpha)$ is axiomatized by a finite set $BASIC$ of bounded first order axioms codifying the recursive properties of function symbols and predicates $=, \leq$ ($BASIC$ contains no axioms about α) and by the induction scheme $IND(a, P)$, for all bounded formulas P of the language $L(\alpha)$.

The class $\Sigma^b_\infty(\alpha)$ of bounded formulas of $L(\alpha)$ is stratified into levels $\Sigma^b_0(\alpha) \subset \Sigma^b_1(\alpha) \subset \ldots$ similarly as the arithmetical formulas are stratified into levels $\Sigma^0_0 \subset \Sigma^0_1 \subset \ldots$. In particular, $\Sigma^b_0(\alpha)$ is the class of bounded formulas with all quantifiers bounded by a term of the form $|t|$ (the *sharply bounded* quantifiers), and $\Sigma^b_i(\alpha)$ is the class of bounded formulas with $(i-1)$ alternations of bounded quantifiers, starting with the existential one and without counting the sharply bounded ones. The important property of the $\Sigma^b_i(\alpha)$-formulas is that they define exactly the predicates in the i^{th} level $\Sigma^p_i(\alpha)$ of the polynomial time hierarchy PH^α relative to the oracle α. For example, the $\Sigma^b_1(\alpha)$-formulas define exactly the NP^α-predicates.

A subtheory $T^i_2(\alpha)$ is obtained from $T_2(\alpha)$ by restricting the $IND(a, P)$ scheme to $\Sigma^b_i(\alpha)$-formulas P only. Other important subtheories of $T_2(\alpha)$ are the theories $S^i_2(\alpha)$ that are based on a modified induction scheme for $\Sigma^b_i(\alpha)$-formulas.

It holds that: $S^i_2(\alpha) \subseteq T^i_2(\alpha) \subseteq S^{i+1}_2(\alpha)$ ([2]); $S^{i+1}_2(\alpha)$ is $\forall\Sigma^b_{i+1}(\alpha)$ - conservative over $T^i_2(\alpha)$ ([4]), although $S^{i+1}_2(\alpha) \neq T^i_2(\alpha)$ ([17]). Further it holds that $T^i_2(\alpha) \neq S^i_2(\alpha)$ ([14]) and, in fact, that $T^i_2(\alpha)$ is not $\forall\Sigma^b_i(\alpha)$ - conservative over $S^i_2(\alpha)$ ([5]).

Thus all possible conservation relations between $S^{i+1}_2(\alpha)$ and $T^i_2(\alpha)$ are decided, since $S^{i+1}_2(\alpha)$ is $\forall\Sigma^b_{i+2}(\alpha)$-axiomatizable. With respect to our study of conservation results between subtheories of $T_2(\alpha)$, the theories $S^{i+1}_2(\alpha)$ and $T^i_2(\alpha)$ are indistinguishable and we may therefore confine ourselves mostly to theories $T^i_2(\alpha)$.

To show that a formula $\Phi(\alpha)$ is not valid in all structures is equivalent to a computational complexity theory task to find an oracle α for which the principle $\Phi(\alpha)$ fails.

To establish the unprovability in $T_2(\alpha)$ of a bounded formula $\Phi(\alpha)$ which is, in fact, valid for all interpretations of α in all finite structures is related (though not equivalent) to the task to show that the principle $\Phi(\alpha)$ is not *witnessed* by a polynomial time machine with an oracle from $\Sigma^p_i(\alpha)$, fixed i for all α.

We shall not define the general concept of witnessing ([2]) but merely state one witnessing theorem for the theory T^1_2 proved in [5], and its straightforward generalization to the theories $T^i_2(\alpha)$ we shall need (Theorem 1.1).

A *search problem* (or a *multifunction*) is given by a binary predicate $R(x, y)$; if R is expressible by a Σ^b_i-formula we say that it is a Σ^b_i-search problem. Any y such that $R(x, y)$ holds is called a *solution* for *instance* x. The problem is *well-defined* if:

$$\forall x \exists y R(x, y)$$

holds, and it is Σ^b_i*-definable* in a theory T if there is a Σ^b_i-formula $\theta(x, y)$ expressing $R(x, y)$ such that:

$$\forall x \exists y \theta(x, y)$$

is provable in T.

The *search task* is: given an instance x find any solution y. Note that a solution is not necessarily unique.

This is a general concept and to fit in various examples of search problems the relation R must encode several functions and relations specifying a natural combinatorial setting for the search problem. Various classes of search problems were considered in [19, 20]. In this paper, search problems are defined, or specified, in terms of a list of data (functions, relations) that are required to satisfy a certain condition. A relation (thus, a search problem) is naturally associated to the given specifying data and well-definiteness is implied by the required condition.

A foremost example is the notion of a *polynomial local search* problem (PLS - problem) of [20]. An *instance* of a PLS-problem L is any finite string x of $0, 1$ (equivalently a number identified with its binary expansion). For any x there is a set $F_L(x)$ of *solutions*:

$$F_L(x) := \{y \in \{0,1\}^* \mid |y| \leq |x|^\ell\},$$

where ℓ is a constant depending on L. Any solution $s \in F_L(x)$ has its *cost* $C_L(x, s)$: a natural number. Moreover, the set $F_L(x)$ is augmented by a *neighbourhood function* $N_L(x, s)$ such that for $s \in F_L(x)$:

$$(*_L) \qquad [\ N_L(x,s) = s\ ] \vee [\ N_L(x,s) \in F_L(x) \wedge C_L(x, N_L(x,s)) < C_L(x,s)\ ]\ .$$

Hence $N_L(x, s)$, if different from s, provides a solution of a smaller cost than is that of s. A crucial requirement is that both functions $C_L(x, s)$ and $N_L(x, s)$ are *polynomial time functions* of x, s.

The search task of the PLS-problem L is, given x, find a solution $s \in F_L(x)$ for which $N_L(x, s) = s$. Such solution is called *locally optimal.* We refer the reader to [20] for examples of PLS-problems.

Clearly, the search problem determined by the relation $R(x, y)$:

$$R(x,y)\ \equiv_{df}\ (y \in F_L(x) \wedge N_L(x,y) = y)$$

is just the PLS-problem L and the condition $(*_L)$ implies that it is well-defined.

An *oracle PLS-problem* is defined identically, allowing functions $C_L(x, s)$ and $N_L(x, s)$ to be computed by polynomial-time oracle machines. For a given oracle PLS-problem L the machines computing these two functions are fixed.

An oracle PLS-problem L whose machines have access to a particular oracle α is called a PLS^α-problem and denoted $L(\alpha)$.

Theorem 1.1 ([5]) *Let $\Theta(x, y, \alpha)$ be a $\Sigma^b_i(\alpha)$-formula, $i \geq 1$. Assume that:*

$$T^i_2(\alpha) \vdash \forall x \exists y \Theta(x,y,\alpha)\ .$$

Then there is an oracle PLS-problem L and a $\Sigma^b_{i-1}(\alpha)$-formula $\beta(x, \alpha)$ defining for all $\alpha \subseteq \omega$ the set $\beta^\alpha = \{x \in \omega \mid \beta(x, \alpha)\}$ such that for all α:

1. *the condition* $(*_L)$ *from the definition of* PLS*-problems holds for the problem* $L = L(\beta^\alpha)$
2. *whenever* $s \in F_{L(\beta^\alpha)}(x, s)$ *is a locally optimal solution, then*
   (a) s *has the form* $(y, z_1, \ldots, z_k)$
   (b) $\Theta(x, y, \alpha)$ *is valid.*

In particular, the $\forall\exists\Sigma^b_1$*-consequences of* T^1_2 *are witnessed by projections of* PLS*-problems and the* $\forall\exists\Sigma^b_2$*-consequences of* T^2_2 *are witnessed by projections of* PLS^{NP}*-problems.*

We remark only that the opposite statement is also valid: every $PLS^{\Sigma^p_{i-1}(\alpha)}$-problem is $\forall\exists\Sigma^b_i(\alpha)$-definable in $T^i_2(\alpha)$, see [5]. However, in this opposite statement the formula $R(x, y)$ defining the search problem need not to be the natural one as above but rather a Δ^b_1-formula $R'(x, y)$ such that the $\forall\Sigma^b_1$-sentence $(\forall x, s, (*_L)) \rightarrow \forall x \exists y, R(x, y)$ is expressible as $\forall x \exists y R'(x, y)$. For example, for $R'(x, y)$ defined by:

$$[N_L(x, y) \neq y \wedge (N_L(x, y) \notin F_L(x) \vee C_L(x, N_L(x, y) \geq C_L(x, y))] \vee$$
$$\vee\ [y \in F_L(x) \wedge N_L(x, y) = y]\ ,$$

the formula $\forall x \exists y R'(x, y)$ is provable in T^1_2 and $\forall x \exists y R'(x, y) \equiv \forall x \exists y R(x, y)$ holds, as the first disjunct can be never satisfied (by $(*_L)$).

2 Search problems that are Σ^b_1-definable in T^1_2

In [13], a herbrandization of the induction axiom for a particular $\Sigma^b_1(\alpha)$-formula was studied. Such herbrandization was reformulated in [5] as a finite Σ^b_1-combinatorial principle: the *iteration principle.*

We shall reformulate it as a search problem.

Definition 2.1 *An* iteration problem *(I-problem) is given by a polynomial-time function* $f(x, y)$ *satisfying, for all* x*, the following conditions:*

1. $0 < f(x, 0)$
2. $\forall y < x, y < f(x, y) \rightarrow f(x, y) < f(x, f(x, y))$

Numbers $x > 0$ *are the instances of the problem and, for any* x*,* $y < x$ *is a solution for* x*, if*

$$f(x, y) \geq x\ .$$

The iteration principle considered in [5] says that any I-problem is well-defined. Let $Iter(a, f)$ be the $\Sigma^b_1(f)$-formula expressing that the problem has a solution for instance a:

$$0 = f(a, 0) \vee \exists y < a, (y < f(a, y) \wedge f(a, y) \geq f(a, f(a, y))) \vee f(a, y) \geq a\ .$$

Theorem 2.2 ([5]) *The $\Sigma_1^b(f)$-formula $Iter(a,f)$ is provable in $T_2^1(f)$ but not in $S_2^1(f)$.*

We note as a simple observation that the I-problems actually characterize the Σ_1^b-consequences of T_2^1.

Theorem 2.3 *Let $\theta(x,y)$ be a Σ_1^b-formula and assume that*

$$T_2^1 \vdash \forall x \exists y \theta(x,y) \ .$$

Then there is an I-problem and a term $t(x)$ such that any solution u for an instance $a := t(x)$ has the form $u = (y,z)$ and $\theta(x,y)$ holds.

Proof :

We use Theorem 1.1. Let L be the PLS-problem whose projections witness the formula $\forall x \exists y \theta(x,y)$ in the sense of Theorem 1.1. Let $t(x)$ be a term such that $t(x)$ is larger than every $s \in F_L(x)$ and, for $s \in F_L(x)$ larger than all $c = C_L(x,s)$. Code pairs (u,v) of numbers $u,v < t(x)$ by $(t(x)-v)\cdot t(x) + u$. W.l.o.g., we may assume that the pair $(0, C_L(x,0))$ is coded by 0.

Define a polynomial-time function f by:

$$f(x,(s,c)) := 0 \ \text{ if } \ s \notin F_L(x) \vee c \neq C_L(x,s)$$

and otherwise define:

$$f(x,(s,c)) := \begin{cases} (N_L(x,s), C_L(x, N_L(x,s))) & \text{if } s \neq N_L(x,s) \\ t(x)^3 & \text{otherwise} \end{cases}$$

Then the condition $(*_L)$ from the definition of PLS-problems implies that the function f satisfies the requirements of Definition 2.1, and that any solution u of the iteration problem must satisfy $f(x,u) = t(x)^3$. Hence, $u = (s, C_L(x,s))$ for some locally optimal solution s of the PLS-problem L.

q.e.d.

After the experience offered by Theorems 1.1 and 2.3 we define the notion that a class of search problems characterizes search problems that are Σ_i^b-definable in a theory T. Recall the definition of search problems from Section 1.

Definition 2.4 *Let $\mathcal{L}$ be a class of search problems and let T be a theory in a language extending the language of the theory T_2.*

The class $\mathcal{L}$ characterizes *search problems that are Σ_i^b-definable in T iff the following two conditions hold:*

1. *For every search problem $R(x,y) \in \mathcal{L}$ there is a Σ_i^b-formula $\theta(x,y)$ such that:*

(a) $T \vdash \forall x \exists y \theta(x, y)$

(b) For every $x, y : \theta(x, y) \rightarrow R(x, y)$.

2. *For every* Σ_i^b*-formula* $\theta(x, y)$ *for which* T *proves* $\forall x \exists y \theta(x, y)$ *there exists a search problem* $R(x, y) \in \mathcal{L}$ *which is well-defined and such that any solution* z *for an instance* x *has the form:*

$$z = (y, z_1, \ldots, z_k)$$

and $\theta(x, y)$ *holds.*

Thus the classes of PLS-problems and of I-problems both characterize the search problems that are Σ_1^b-definable in the theory T_2^1.

3 Search problems that are Σ_2^b-definable in S_2^1 and in T_2^1

The search problems that are Σ_2^b-definable in S_2^1 and in T_2^1 were characterized in computational terms in [14, 4]. In this section we reformulate these characterizations in terms of natural search problems.

Definition 3.1 *A* function-maximization *problem (FM-problem) is given by two polynomial-time functions* $g(x)$ *and* $f(x, y)$. *A* solution *for an instance* x *is any* $y < g(x)$ *such that:*

$$f(x, y) = max_{z<g(x)} f(x, z) \ .$$

The FM*-problem is* sharply bounded *iff*

$$\forall y < g(x), f(x, y) < |t(x)|$$

for some term $t(x)$.

Lemma 3.2 *Every* FM*-problem is* Σ_2^b*-definable in* T_2^1.

Proof :

The formula $\theta(x, y)$:

$$y < g(x) \wedge \forall z < g(x), f(x, z) \leq f(x, y)$$

is $\Pi_1^b \subseteq \Sigma_2^b$ and defines the FM-problem. To prove that $\forall x \exists y \theta(x, y)$, consider the following Σ_2^b-formula $\phi(u)$:

$$\exists v < h(x) \exists y < g(x) \forall z < g(x), v \leq f(x, y) \wedge f(x, z) \leq v + u \ ,$$

where $h(x)$ is an apriori polynomial-time bound to $max_{z<g(x)} f(x,z)$. Clearly, $\phi(u)$, for $u := h(x)$, and:

$$\phi(u) \to \phi(\lfloor \frac{u}{2} \rfloor) \ .$$

Hence, S^2_2 proves $\phi(0)$ and the witnesses v, y to the validity of $\phi(0)$ yield a solution y for the instance x.

This shows that FM-problems are Σ^b_2-definable in S^2_2 and hence (by the $\forall\Sigma^b_2$-conservativity of S^2_2 over T^1_2, see [4]) they are also definable in T^1_2.

q.e.d.

The theory T_3 is defined exactly as T_2 except that its language contains one more function symbol $\#_2$ where:

$$x \#_2 y := 2^{|x| \# |y|} \ ,$$

see [2]. The theory R_3 was defined in [29] and it is based on the following induction axiom:

$$(\psi(0) \wedge \forall x(\psi(\lfloor \frac{x}{2} \rfloor) \to \psi(x))) \ \to \ \forall x \psi(|x|) \ .$$

Lemma 3.3 *Every sharply bounded FM-problem is Σ^b_2-definable in S^1_3.*

Proof :

The proof goes analogously with the proof of Lemma 3.2 with one change. As the FM-problem is sharply bounded, the formula $\phi(u)$ holds for $u := |t(x)|$ and hence the induction available in R^2_3 proves that $\phi(0)$ is valid.

By [6] the theory R^2_3 is $\forall\Sigma^b_2$-conservative over S^1_3, which yields the lemma.

q.e.d.

It is an open problem whether the theory R^2_2 is also $\forall\Sigma^b_2$-conservative over S^1_2. If so, the previous lemma would hold with S^1_2 in place of S^1_3.

The next theorem is a converse to Lemma 3.2.

Theorem 3.4 *The class of FM-problems characterizes the search problems that are Σ^b_2-definable in T^1_2.*

Proof :

Lemma 3.2 shows that every FM-problem is Σ^b_2-definable in T^1_2.

By [2] the $\forall\Sigma^b_2$-consequences of T^1_2 are witnessed by $\square^p_2$-functions. Hence it is sufficient to prove the following claim.

Claim : *Let $F(x)$ be a $\square^p_2$-function. Then there exists a term $t(x)$ and a polynomial-time function $f(x,y)$ such that: for all x,* *if $u < t(x)$ is a solution to the FM-problem given by $t(x)$ and $f(x,y)$, then:*

$$u = (y, z_1, \ldots, z_k)$$

and $y = F(x)$ holds.

Let N be the oracle polynomial-time machine computing F with the NP-oracle $\exists w \beta(u,w)$. We assume that w is implicitly bounded in the polynomial-time relation β, and that the time bound of N is n^ℓ.

Let D_x be the set of all 5-tuples of the form:

$$(y, v, w, \epsilon, x)$$

where:

1. ϵ is a tuple $(\epsilon_1, \ldots, \epsilon_\ell) \in \{0,1\}^*$.
2. v is a computation of machine N on the input x in which the i^{th} oracle query $[\exists w \beta(u_i, w)?]$ is answered YES and NO according to whether $\epsilon_i = 1$ or $\epsilon_i = 0$.
3. w is a tuple $(w_1, \ldots, w_\ell)$.
4. For all $i \leq \ell$: if $\epsilon_i = 1$ then $\beta(u_i, w_i)$ holds.
5. y is the output of the computation v.

Note that every tuples in D_x has size bounded by some term $t(x)$ (hence $D_x \subseteq \{0, \ldots, t(x) - 1\}$) and $D_x \neq \emptyset$ (take $\epsilon = w = (0, \ldots, 0)$ and the computation v of N whose queries are all answered negatively).

Define the function $f(x,u)$ for $u < t(x)$:

$$f(x,u) := \begin{cases} 1 + \sum_{i=1}^{\ell} \epsilon_i \cdot 2^{\ell - i} & \text{if } u = (y, v, w, \epsilon, x) \in D_x \\ 0 & \text{otherwise} \end{cases}$$

Assume that $u = (y, v, w, \epsilon, x)$ yields the maximal value of $f(x,u)$, for $u < t(x)$. We claim that all oracle queries in v are answered correctly by ϵ. The affirmative answers are correct, as they are witnessed by w. Assume that a negative answer $\epsilon_i = 0$ is incorrect and let z be such that $\beta(u_i, z)$. Consider a new ϵ':

$$\epsilon'_j := \begin{cases} \epsilon_j & \text{if } j < i \\ 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

and a new w':

$$w'_j := \begin{cases} w_j & \text{if } j < i \\ z & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Let v' be the unique computation of N on x with the oracle answers ϵ' and the output y'. Then, however, $f(x, u') > f(x, u)$, for $u' = (y', v', w', \epsilon', x) \in D_x$.

q.e.d.

Lemma 3.2 and Theorem 3.4 generalize to $T^1_2(\alpha)$ and oracle FM-problems. Hence the following corollary follows from the separation $S^1_2(\alpha) \neq T^1_2(\alpha)$, cf. [13, 26, 14].

Corollary 3.5 *The oracle FM-problem with oracle functions f and g is not $\Sigma^b_2(f, g)$-definable in $S^1_2(f, g)$.*

We consider one more reformulation of the characterization of $\forall\Sigma^b_2$ - consequences of T^1_2.

Theorem 3.6 *The search problems that are Σ^b_2-definable in T^1_2 are characterized by the class of search problems given by a polynomial-time function $g(x)$ and a polynomial-time predicate $H(x, y)$, in which a solution to an instance x is the minimal element of the set $\{y < g(x) \mid H(x, y)\} \cup \{g(x)\}$.*

Proof :
That every such problem is Σ^b_2-definable in T^1_2 follows from Lemma 3.2, when f is defined by:

$$f(x, y) := \begin{cases} g(x) - y & \text{if } H(x, y) \\ g(x) & \text{otherwise} \end{cases}$$

For the other direction, note that in the proof of Theorem 3.4 we may code tuples $u \in D_x$ in such a way that if $f(x, u) > f(x, u')$ then the code of u is bigger than the code of u'. Take H to be the definition of such codes.

q.e.d.

Since we proved Lemma 3.3 for S^1_3 instead for S^1_2, we cannot prove a characterization in terms of sharply bounded FM-problems of the search problems that are Σ^b_2-definable in S^1_2 (analogous to Theorem 3.4). However, one direction still works.

Theorem 3.7 *Every search problem that is Σ^b_2-definable in S^1_2 is represented by a sharply bounded FM-problem, i.e., it is a projection of a sharply bounded FM-problem.*

Proof :

The search problems that are Σ_2^b-definable in S_2^1 are characterized as those problems which can be witnessed by a polynomial-time machine that may ask $O(\log\log x)$ queries to an NP-oracle, and modified in such a way that the oracle supplements its affirmative answers with witnesses to those answers (cf. [14], in [6] this class of search problems is called $FP^{NP}[wit, O(\log n)]$).

This implies that an FM-problem defined as in the proof of Theorem 3.4 is, in the case of S_2^1, actually sharply bounded, as $\ell \leq O(\log\log x)$.

q.e.d.

4 Search problems that are Σ_2^b- and Σ_3^b- definable in T_2^2

Theorem 1.1 gives a characterization of the search problems that are Σ_2^b - definable in T_2^2 in terms of oracle PLS - problems. An axiomatization of the $\forall\Sigma_2^b$ - consequences of T_2^2 by the reflection principles of the proof system G_2 is given in [16]. Neither characterization is entirely satisfactory. The latter one is not satisfactory because the reflection principles lack a direct combinatorial meaning. In the former characterization the definition of the search problems contains unfeasible functions ($\Box_2^p$), while we would like a characterization involving only polynomial-time objects and whose search task complexity stems from the task itself rather than from the underlying structure.

In this section we give such characterizations of search problems that are Σ_2^b- and Σ_3^b-definable in T_2^2. We start with the latter, as it is simpler.

Definition 4.1 *A* minimal minimum *problem (MM-problem) is given by a polynomial-time relation $R(x, y, z)$ and polynomial-time functions $c(x, y)$ and $g(x)$ such that the relation*

$$\{(y, z) \mid R(x, y, z)\}$$

is a strict partial ordering of the set $\{u \mid u < g(x)\}$ (henceforth, we write $\prec_x$).

A solution y for an instance x is any $y < g(x)$ satisfying the following two conditions:

1. *y is a minimal element of the ordering $\prec_x$.*

2. *For any $\prec_x$-minimum z, $c(x, y) \leq c(x, z)$.*

Theorem 4.2 *Search problems that are Σ_3^b-definable in T_2^2 are characterized by the class of MM-problems. In particular, the theory T_2^2 is axiomatizable over S_2^1 by the statements formalizing that all MM-problems are well-defined.*

Proof :

The property that y is $\prec_x$-minimal with cost u is Π^b_1. Hence the set X of pairs (u, y) such that $y < g(x)$ is a $\prec_x$-minimum and $u = c(x, y)$ is Π^b_1-definable. Moreover, the property that $\prec_x$ is a strict partial ordering on $\{0, \dots, v\}$ is also Π^b_1. Thus the formula $\phi(v)$ expressing that if $\prec_x$ is a partial ordering of $\{0, \dots, v\}$ then there is $y \leq v$ $\prec_x$-minimal on $\{0, \dots, v\}$ is Σ^b_2, and so the theory T^2_2 suffices to prove by induction on v that there is at least one $\prec_x$-minimum on the whole set $\{0, \dots, g(x) - 1\}$. Hence, the set X is non-empty. Since T^2_2 proves Π^b_1-MIN axiom, X has a minimal element (w.r.t. to the standard ordering). W.l.o.g. we may assume that the minimality of the pair (u, y) implies the minimality of u.

This shows that T^2_2 proves that all MM-problems are well-defined.

To verify that every search problem which is Σ^b_3-definable in T^2_2 is represented by an MM-problem, it is sufficient to show that the assumption that all MM-problems are well-defined implies (over S^1_2) that all non-empty Π^b_1-definable sets have a minimum, as the latter statement (the Π^b_1-MIN axioms) $\forall\Sigma^b_3$-axiomatizes T^2_2 (cf. [2]).

Let $B(x, u) = \forall v < x, A(x, u, v)$ be a Π^b_1-formula with $A \in \Delta^b_1$. For the pairs $(u, v), (r, s) \in (x + 1) \times (x + 1)$ define

$$(u, v) \prec_x (r, s)$$

iff $u = r$ and one of the following three conditions holds:

1. $(v = x \lor \neg A(x, u, v)) \land A(x, r, s)$
2. $A(x, u, v) \land A(x, r, s) \land v < s$
3. $\neg A(x, u, v) \land v < x \land (\neg A(x, r, s) \lor s = x) \land v < s$.

Note that if (u, v) is $\prec_x$-minimal then

$$B(x, u) \text{ iff } v = x \ .$$

Define the function:

$$c(x, (u, v)) := \begin{cases} u & \text{if } v = x \\ x & \text{otherwise.} \end{cases}$$

Hence for a $\prec_x$-minimum (u, v):

$$c(x, (u, v)) := \begin{cases} u & \text{if } B(x, u) \text{ holds} \\ x & \text{otherwise.} \end{cases}$$

Clearly, assuming that $B(x, u)$ is satisfied by at least one element u, the $\prec_x$-minimum of the smallest possible cost (it is unique) is a pair (u, x), where u is the smallest number satisfying $B(x, u)$.

q.e.d.

Now we move on to characterizations of search problems that are Σ_2^b - definable in T_2^2. We generalize the definition of local search problems.

Definition 4.3 *A* generalized local search *problem (GLS-problem) is given by the following data:*

1. *a polynomial-time function $g(x)$ (specifying the domain as the set of $y < g(x)$)*
2. *a polynomial-time function $c(x, y)$ (the cost function)*
3. *a polynomial-time relation $N(x, y, z)$ (the neighbourhood of y, we write N_x^y)*
4. *a polynomial-time relation $R(x, y, u, v)$ (the ordering on N_x^y, we write $\prec_x^y$)*

satisfying the following conditions:

(i) $N_x^y \neq \emptyset$, all $y < g(x)$

(ii) $\prec_x^y$ is a strict linear ordering of N_x^y, all $y < g(x)$

(iii) if $u \in N_x^y$ is $\prec_x^y$-minimal and $v \in N_x^u$ is $\prec_x^u$-minimal then:

$$c(x, v) \leq c(x, u) \ .$$

The search task is : given an instance x find $u, v, w < g(x)$ such that

(a) $v \in N_x^u$ is $\prec_x^u$-minimal

(b) $w \in N_x^v$ is $\prec_x^v$-minimal

(c) $c(x, v) = c(x, w)$.

Since the definition is not terribly illuminating, let us consider an example. Let $f(x, y)$ be a cost function defined on the elements of the d-dimensional cube $x \times x \times \ldots \times x$, and define the neighbours to be the elements which differ in every coordinate by at most 1. The task is to find a local optimum; i.e., an y such that no neighbour of y has a smaller cost.

If d is a constant (or $O(\log n)$, where $n = |x|$), then a neighbourhood has at most polynomially many elements and a polynomial-time algorithm may search throughout it to find an element of minimal cost. This is just the usual PLS-problem.

If $d = n$, the domain is still bounded by some $g(x)$. However, a neighbourhood may have exponential size and no polynomial-time algorithm can search through it. In this case, however, we still have a linear ordering in every neighbourhood (given by the cost) and the minimal neighbours are those where the cost does not increase. Hence, we have a GLS-problem.

Theorem 4.4 *The search problems that are Σ_2^b-definable in T_2^2 are characterized by the class of GLS-problems.*

Proof :
Consider the formula $\phi(c)$ expressing that c is the cost of a $\prec_x^y$-minimum for some $y < g(x)$:

$$\exists y, u < g(x) \forall v < g(x), \ \ u \in N_x^y \wedge c = c(x,u) \wedge \ \ (v \in N_x^y \rightarrow (u = v \vee u \prec_x^y v)) \ .$$

Let $Y = \{c \mid \phi(c)\}$. Then Y is Σ_2^b-definable and non-empty (as every neighbourhood N_x^y is non-empty and contains $\prec_x^y$-minimal elements).

By the Σ_2^b-MIN axioms available in T_2^2 there exists a minimal element $c \in Y$. Witnesses y, u to the validity of $\phi(c)$ provide a solution of the GLS-problem.

To prove that every search problem which is Σ_2^b-definable in T_2^2 can be represented by a GLS-problem, we employ Theorem 1.1. Thus, it is sufficient to show that every PLS^{NP}-problem can be represented by a GLS-problem.

Let L be an PLS-problem with access to an NP-oracle $\exists w \beta(u,w)$, where β is a polynomial-time predicate and w is implicitly bounded in β. First we define the data specifying the GLS-problem and then we shall verify that it has the required properties.

Elements of N_x^y, $y = (z_0, c_0, v_0, w_0, \epsilon_0, y_0)$, are 6-tuples of the form:

$$(z, c, v, w, \epsilon, y_0)$$

where:

1. ϵ is a tuple $(\epsilon_1, \ldots, \epsilon_\ell) \in \{0,1\}^*$, where n^ℓ is a fixed time bound to a machine computing simultaneously $N_L(x, y_0)$ and $C_L(x, N_L(x, y_0))$: Call such machine $M_L(x, y_0)$.

2. v is a computation of the machine $M_L(x, y_0)$ on the input x, y_0 in which the i^{th} oracle query $[\exists w \beta(u_i, w)?]$ is answered YES or NO according to whether $\epsilon_i = 1$ or $\epsilon_i = 0$.

3. w is a tuple $(w_1, \ldots, w_\ell)$.

4. For all $i \leq \ell$: if $\epsilon_i = 1$ then $\beta(u_i, w_i)$ holds.

5. (z, c) is the output of the computation v.

The ordering $\prec_x^y$ on N_x^y is defined by the anti-lexicographic order of the ϵ's, and the cost of such y is defined to be its second component:

$$c(x,y) := c \ .$$

Condition $(*_L)$ from the definition of PLS-problems in Section 1 implies that the GLS-problem defined in this way satisfies the conditions $(i)-(iii)$ of Definition 4.3.

On the other hand, a solution u, v, w:

$$u = (z^u, c^u, v^u, w^u, \epsilon^u, y^u)$$

$$v = (z^v, c^v, v^v, w^v, \epsilon^v, y^v)$$

$$w = (z^w, c^w, v^w, w^w, \epsilon^w, y^w)$$

to the GLS-problem satisfies:

$$z^v = N_L(x, y^u) \wedge c^v = C_L(x, z^v)$$

and

$$z^w = z^v \ .$$

Hence z^v is a locally optimal solution of the PLS^{NP}-problem.

q.e.d.

We conclude this section by defining a simple Σ_2^b-search problem which is definable in T_2^2.

Definition 4.5 *A* minimization *problem (MIN - problem) is given by a polynomial - time function $g(x)$ and a polynomial - time relation $y \prec_x z$ such that for all x the relation $\prec_x$ is a strict linear ordering of the set $\{0, \ldots, g(x)-1\}$.*

The search task is : given an instance x find $y < g(x)$ which is $\prec_x$-minimal.

Every MIN-problem is Σ_2^b-definable in T_2^2 (for fixed x show by induction on u that there is the $\prec_x$-minimal element among those smaller than u). However, the oracle version with $\prec_x$ being an oracle relation $\prec$ (same for all x) is not $\Sigma_2^b(\prec)$-definable in $S_2^2(\prec)$. This follows, for example, from [27], where a non-standard model M of $S_2^2(\prec)$ is constructed in which, for some $a \in M$, the relation $\prec$ is a strict linear ordering of $\{0, \ldots, a\}$ without the minimal element.

We do not know whether the class of MIN-problems characterizes the search problems that are Σ_2^b-definable in T_2^2.

5 A version of the pigeonhole principle

In this section we present the first proof that $T_2^2(\alpha)$ is not $\forall\Sigma_1^b(\alpha)$-conservative over $T_2^1(\alpha)$. The $\forall\Sigma_1^b(\alpha)$-formula that separates the two theories formalizes a version of the *pigeonhole principle.*

The pigeonhole principle (or *Dirichlet's Schubfachprinzip*) is the basic principle saying that there is no bijection f defined on a finite set D and with values in a finite set R, if $|D| > |R|$. A strong version of the principle is for $|D| = 1+|R|$ but for bounded arithmetic weaker versions are needed, where $|D| = 2 \cdot |R|$ or $|D| = |R|^2$. To explain the difference between these versions, we sketch a proof of the following theorem.

Theorem 5.1 ([24]) *Let $WPHP(a^2, f)$ be the following formula*

$$(\exists x < a^2, f(x) \geq a) \vee (\exists x < y < a^2, f(x) = f(y)) \vee$$

$$(\exists y < a \forall x < a^2, f(x) \neq y) \ .$$

Then the theory $T^2_2(f)$ proves the $\forall\Sigma^b_2(f)$-formula

$$\forall a \ WPHP(a^2, f).$$

Proof-sketch:

Assume that $f : a^2 \to a$ violates $WPHP(a^2, f)$. Then the function $f_1 : a^4 \to a$:

$$f_1(u \cdot a^2 + u) = f(u) \cdot a + f(v), \ \text{for } u, v < a^2$$

is a bijection between a^4 and a. Iterating this procedure $t = \lceil log_2(a) \rceil$ -times we get a bijection $f_t : a^a \to a$.

In particular, f_t defines an injection of the set of subsets of a (identified with their characteristic functions) into a and the usual *Cantor's* diagonal argument applies.

In bounded arithmetic, however, we cannot formalize this argument directly, as we cannot prove that the number a^a exists (exponentiation is not provably total, cf. [21]). However, thinking about elements $< a^a$ as coding sequences of a numbers $< a$, given $w < a$ and $i < a$ we can still meaningfully define the function:

the i^{th}-element of the sequence mapped by f_t to w,

(use the inverse function $f_t^{(-1)}$ to f_t defined using the iterations of the inverse function to f). That is enough to carry out the diagonal argument, as the diagonal set can be defined as:

$$\{i < a | i \notin f_t^{(-1)}(i)\} \ .$$

Writing down the definitions of f_t and $f_t^{(-1)}$ shows that the argument is formalized in $S^3_2(f)$. By $\forall\Sigma^b_3(f)$-conservativity of $S^3_2(f)$ over $T^2_2(f)$ (cf. [4]), the theorem is proved.

q.e.d.

The reader may wonder why we simply do not prove that any bijection must preserve the cardinality of the domain. This is because there is no bounded $L(\alpha)$-formula $\varphi(x, y)$ such that for all α and n, m:

$$\varphi(n, m) \text{ holds } \quad \textit{iff} \quad |\{i < n \mid \alpha(i)\}| = m \ .$$

In other words, we cannot define the cardinality of finite sets by a bounded formula, [1, 11, 12, 30].

We also cannot simulate the proof with a function $f : a + 1 \to a$ in place of a function from a^2 to a; the construction of $f_t : a^a \to a$ would require $\Omega(a)$ iterations and the definition of $i \in (f_t)^{(-1)}(w)$ would need to code sequences of length $\Omega(a)$, which is impossible in bounded arithmetic. Indeed, the theory $T_2(f)$ does not prove the pigeonhole principle for $f : a + 1 \to a$, cf. [18, 25].

The formula $WPHP(a^2, f)$ is only $\Sigma_2^b(f)$ while we want a $\Sigma_1^b(f)$-formula. To express with a smaller quantifier complexity the condition that f is onto we introduce a function symbol g for the inverse function of f.

Let $WPHP(a^2, f, g)$ be the following $\Sigma_1^b(f, g)$-formula:

$$(\exists x < a^2, f(x) \geq a) \vee (\exists x < y < a^2, f(x) = f(y)) \vee$$
$$\vee\ (\exists x < a, g(x) \geq a^2) \vee (\exists y < a, f(g(y)) \neq y) \ .$$

Lemma 5.2 *Let R be either the theory S_2^i or the theory T_2^i for some i, and let $R(f)$ and $R(f, g)$ denote the same theory in the language expanded by f or by f, g respectively.*

Then the theory $R(f)$ proves the $\forall\Sigma_2^b(f)$-formula

$$\forall a \ WPHP(a^2, f)$$

iff the theory $R(f, g)$ proves the $\forall\Sigma_1^b(f, g)$-formula

$$\forall a \ WPHP(a^2, f, g) \ .$$

Proof:

The *only if* part is obvious as $\neg WPHP(a^2, f, g)$ implies that f is onto. For the *if* part let (M, f) be a model of $R(f)$ in which $WPHP(a^2, f)$ fails for $a := m \in M$. Define g in (M, f) to be the inverse function to f. Clearly, the expanded structure (M, f, g) satisfies $\neg WPHP(m^2, f, g)$ and it also satisfies all induction axioms of $R(f, g)$, as $g(y) = x$ is defined by an open formula $f(x) = y$ and any other formula involving terms with g can be rewritten using the graph $g(y) = x$ instead, without increase of quantifier complexity. The open formula $A(z/g(y))$ is equivalent to

$$\exists x < a^2, g(y) = x \wedge A(z/x)$$

and also to

$$\forall x < a^2, g(y) = x \to A(z/x)$$

and so its is a $\Delta_1^b(f)$-formula.

q.e.d.

The following result improves upon [13] where it was proved that the formula $WPHP(a^2, \alpha)$:

$$(\exists x < a^2 \forall y < a \neg\alpha(x, y)) \vee (\exists x < y < a^2 \exists z < a, \alpha(x, z) \wedge \alpha(y, z)) \vee$$

$$(\exists y < a \forall x < a^2, \neg\alpha(x, y)) \vee (\exists y < z < a \exists x < a^2, \alpha(x, y) \wedge \alpha(x, z))$$

formalizing $WPHP$ with a predicate symbol α for the graph of f rather than with a symbol for f itself, is not provable in $S^2_2(\alpha)$.

Theorem 5.3 *The formula $WPHP(a^2, f)$ is not provable in $T^1_2(f)$ and hence it is not provable in $S^2_2(f)$ either.*

Proof:

For the sake of contradiction assume that $WPHP(a^2, f)$ is provable in $T^1_2(f)$. By Lemma 5.2 $T^1_2(f, g)$ proves $WPHP(a^2, f, g)$. By Theorem 1.1 there is an oracle PLS-problem L such that for every f, g the problem $L(f, g)$ witnesses the formula. That is: whenever $s = (y, z_1, \ldots, z_k)$ is a locally optimal solution for an instance $x := a$ of the problem $L(f, g)$, one of the following conditions holds:

1. $y < a^2 \wedge f(y) \geq a$
2. $y = (u, v) \wedge \; u < v < a^2 \wedge f(u) = f(v)$
3. $y < a \wedge g(y) \geq a^2$
4. $f(g(y)) \neq y$

We show that no such oracle PLS-problem L. For any fixed L, we find a number a and functions f, g for which all these four conditions fail.

Let C_L and N_L be the cost function and the neighborhood function associated to L. We identify these functions with the oracle polynomial-time machines computing them. Fix $x := a$ for a large enough (we shall specify this later). We want to find f, g for which there are locally optimal solutions $s \in F_{L(f,g)}(a)$ for which 1. - 4. fail.

We say that a computation of a machine with oracles f, g *respects* a partial function $F :\subseteq a^2 \rightarrow a$ iff, whenever $[f(u) =?]$ is queried in the computation, $u \in dom(F)$ and the oracle answer is $F(u)$, and whenever $[g(v) =?]$ is queried then $v \in rng(F)$ and the answer is $F^{(-1)}(v)$. A computation respecting a partial injective function is called *good*.

Let c_0 be the minimal possible cost $C_{L(f,g)}(a, s)$ computed by the machine C_L for any f, g and some solution $s \in F_{L(f,g)}(a)$ in a good computation. Let F_0 be a partial injective function $\subseteq a^2 \rightarrow a$ respected by such computation w

and let s_0 be the solution for which w is the computation of its cost. Clearly we may assume

$$|dom(F_0)| \leq |a|^k,$$

where n^k majorizes the time bounds of C_L and N_L. Note that there is at least one good computation, and so c_0, F_0, and s_0 are well defined.

The cost c_0 is the minimal possible hence s_0 is locally optimal in all $PLS^{f,g}$-problems $L(f,g)$, for any f,g containing F_0 and $F_0^{(-1)}$ respectively.

Let $s_0 = (y, z_1, \ldots, z_k)$ and assume first that condition 1. holds for F_0:

$$y < a^2 \wedge F_0(y) \geq a \ .$$

That is, however, impossible as F_0 has no value greater than $a-1$. So assume that condition 2. holds:

$$y = \langle u,v \rangle \wedge \ u < v < a^2 \wedge F_0(u) = F_0(v) \ .$$

The last conjunct is not forced by F_0, as F_0 is injective, and in fact, we may always extend F_0 to a partial injective $F_1 :\subseteq a^2 \rightarrow a$ defined on u, v; this is because

$$|dom(F_0)| + 2 \leq |a|^{O(1)} << a.$$

For a similar reason conditions 3. and 4. cannot be forced by F_0.

Hence, the proof is concluded, since we may had chosen a sufficiently large as to satisfy the last inequality.

q.e.d.

The following criterion implying the unprovability of a principle in $S^2_2(\alpha)$ was proved in [27]. If a sentence of a relational language L' disjoint with L admits an infinite model then it is consistent with $S^2_2(L')$ that the sentence has a model with the universe $[0,a]$. This was strengthened in [15] to languages with function symbols. As there is an infinite structure M with a binary function $f(x,y)$ which is a bijection between $M \times M$ and M, the strengthened criterion offers another proof of Theorem 5.3. However, we feel that the first proof is needed should the independence result be lifted to higher fragments of $S_2(\alpha)$, see [7].

Since $T_2(\alpha)$ is defined in terms of a predicate symbol α we need to formalize a principle $\Phi(a,f)$ in terms of α. Of course, we can think of $\alpha(x,y)$ as the graph of f. For example, $WPHP(a^2,f)$ could be translated into the formula $WPHP(a^2,\alpha)$ above which is, however, only $\forall\Sigma^b_2(\alpha)$. This is a genuine difference between the formulas $WPHP(a^2,f)$ and $WPHP(a^2,\alpha)$: the former can be witnessed by a polynomial-time machine with a $\Sigma^b_1(f)$-oracle while the latter cannot be witnessed by a polynomial-time machine with a $\Sigma^b_1(\alpha)$-oracle, cf. [13].

We follow [5] in using a different translation[2] in the language $L(\alpha)$ a principle $\Phi(a, f)$. Interpret $\alpha(x, j)$ as the *bit-graph* of f:

$$\alpha(x,j) \text{ holds } \textit{ iff } \ (f(x))_j = 1 \ ,$$

where $(f(x))_j$ is the j^{th} bit of $f(x)$. Then $f(x) = y$ can be written as the $\Delta_1^b(\alpha)$-formula (since the function $(y)_j$, the j^{th} bit of y, is Δ_1^b-definable):

$$\forall j < |a|; \alpha(x,j) \equiv ((y)_j = 1) \ .$$

Thus, any $\Sigma_i^b(f)$-formula $\Phi(f)$ translates into a $\Sigma_i^b(\alpha)$-formula $\Phi^*(\alpha)$. In particular, $WPHP^*(a^2, \alpha, \beta)$ is the $\forall\Sigma_1^b(\alpha, \beta)$-formula:

$$\exists x < a^2; \alpha(x, |a|) \vee \exists x < y < a^2 \forall j < |a|, \alpha(x,j) \equiv \alpha(y,j) \ \vee$$

$$\exists y < a, \beta(y, |a|) \ \vee$$

$$\exists y < a \exists z < a^2 [\forall j \leq |a|, \beta(y,j) \equiv ((z)_j = 1) \wedge \neg \forall j < |a|, \alpha(z,j) \equiv ((y)_j = 1)] \ .$$

The following lemma is immediate.

Lemma 5.4 *For any $i \geq 1$: the formula $\Phi(a, f)$ is $\Sigma_i^b(f)$ iff the formula $\Phi^*(a, \alpha)$ is $\Sigma_i^b(\alpha)$.*

If R is one of the theories S_2^i or T_2^i, and if $R(f)$ and $R(\alpha)$ denote the same theory in the language expanded by f or by α respectively, then it holds:

$$R(f) \vdash \forall a \Phi(a, f) \ \textit{ iff } \ R(\alpha) \vdash \forall a \Phi^*(a, \alpha) \ .$$

The following corollary follows from Theorems 5.1 and 5.3 using Lemmas 5.2 and 5.4, and from the possibility of coding predicates β_0 (translating f) and β_1 (translating g) in only one predicate α (though the arity increases from 2 to 3):

$$\alpha(x, j, t) \equiv \ [\ (\beta_0(x,j) \wedge t = 0) \vee (\beta_1(x,j) \wedge t = 1) \ ] \ .$$

Corollary 5.5 *The theory $T_2^2(\alpha)$ is not $\forall\Sigma_1^b(\alpha)$-conservative over the theory $T_2^1(\alpha)$.*

Previously only the non-$\forall\Sigma_2^b(\alpha)$-conservativity was known, cf. [13, 26].

To improve Corollary 5.5 we should find a Σ_1^b-search problem that is definable in $T_2^3(\alpha)$ but not in $T_2^2(\alpha)$. Several Σ_1^b-search problems were considered in [19, 20]; in particular the classes PPP, PPA and $PPAD$. Theories of bounded arithmetic corresponding to these classes were identified in [15, Chapter 7]. However, they are not subtheories of $T_2(\alpha)$; i.e., not even the whole theory $T_2(\alpha)$ defines the relativized versions of these classes.

[2] Interpreting $\alpha(x, y)$ as $f(x) \geq y$ and $\beta(x, y)$ as $g(x) \geq y$ yields yet another translation of $WPHP(a^2, f, g)$ into a $\Sigma_1^b(\alpha, \beta)$-formula.

6 A generalized iteration principle

In this section we present another finite Σ_1-principle which also gives a $\Sigma_1^b(\alpha)$-formula separating the theories $T_2^2(\alpha)$ and $T_2^1(\alpha)$. It is a generalization of the *iteration principle* considered in Section 2.

Definition 6.1 *The formula $Iter(a, R, f)$ is the disjunction of the negations of the following eight conditions (we write $x \prec y$ in place of $R(x, y)$):*

1. $0 \prec a$
2. $\forall x \leq a, \neg x \prec x$
3. $\forall x, y, z \leq a,\ x \prec y \wedge y \prec z \rightarrow x \prec z$
4. $\forall x < a, \neg a \prec x$
5. $\forall x \leq a, f(x) \leq a$
6. $0 \prec f(0)$
7. $\forall x < a,\ x \prec f(x) \rightarrow f(x) \prec f(f(x))$
8. $\forall x < a,\ f(x) \prec a$

The principle is valid in every finite structure: whenever $\prec$ is a strict partial order on the set $\{0, \ldots, a\}$ (conditions 2. and 3.) and $0 \prec a$ (condition 1.), the iterations of a function $f : \{0, \ldots, a\} \rightarrow \{0, \ldots, a\}$ satisfying conditions 5. - 7. produces an $\prec$-increasing sequence $0 \prec f(0) \prec \ldots \prec f^{(k)}(0) \prec \ldots$; hence, some $f^{(k)}(0)$ is not $\prec$-smaller than a (which violates conditions 4. and 8.).

In [5] it was proved that the $\Sigma_1^b(f)$-formula $Iter(a, <, f)$, $<$ the standard ordering, separates the theories $S_2^1(f)$ and $T_2^1(f)$ (see Theorem 2.2). We prove now that the formula $Iter(a, R, f)$ separates the theories $S_2^2(R, f)$ and $T_2^2(R, f)$.

Theorem 6.2 *The $\Sigma_1^b(R, f)$-formula $Iter(a, R, f)$ is not provable in the theory $T_2^1(R, f)$ but it is provable in the theory $T_2^2(R, f)$.*

Proof:

The proof of the unprovability of the formula $Iter(a, R, f)$ in $T_2^1(R, f)$ parallels the proof of Theorem 5.3. Assume for the sake of contradiction that $T_2^1(R, f)$ does prove the formula. By Theorem 1.1 there is an oracle PLS-problem L such that, for every R, f, the problem $L(R, f)$ witnesses the formula: whenever $(y, z_1, \ldots, z_k)$ is a local minimum of $L(R, f)$ for the instance $x := a$, one of the following seven conditions holds:

1. $\neg 0 \prec a \vee \neg 0 \prec f(0)$
2. $y \leq a \wedge y \prec y$

3. $y = \langle y_1, y_2, y_3 \rangle \wedge y_1 \leq a \wedge y_2 \leq a \wedge y_3 \leq a \wedge$
   $\wedge y_1 \prec y_2 \wedge y_2 \prec y_3 \wedge \neg y_1 \prec y_3$
4. $y < a \wedge a \prec y$
5. $y \leq a \wedge f(y) > a$
6. $y < a \wedge y \prec f(y) \wedge f(y) \prec a \wedge \neg f(y) \prec f(f(y))$
7. $y < a \wedge \neg f(y) \prec a$

We show how to find, for any fixed L, a number a, an R, and an f, such that none of these conditions hold. That proves the first part of the theorem.

We modify the diagonalization of polynomial-time machines (attempting to witness the formula $Iter(a, R, f)$) from [5].

Pick a sufficiently large a. Call a computation of a machine with oracles R, f *good* iff it satisfies the following properties. After the i^{th} oracle query there are two subsets $\{r_0, \ldots, r_t\}$ and $\{w_1, \ldots, w_m\}$ of $\{0, \ldots, a\}$ such that:

1. $t + m \leq i$
2. $r_0 = 0$

and such that for any of the first i queries it holds:

3. if the query had the form $[f(u) =?]$ then
   either $u = r_j$ some $j < t$ and the oracle answer was r_{j+1}
   or $u \in \{w_1, \ldots, w_m\}$ and the oracle answer was 0
4. if the query had the form $[u \prec v?]$ then $u, v \in \{r_0, \ldots, r_t\} \cup \{w_1, \ldots, w_m\}$ and the oracle answer was according to the ordering:

$$r_0 \prec w_1 \prec \ldots \prec w_m \prec r_1 \prec r_2 \prec \ldots \prec r_t \prec a \ .$$

Let c_0 be the minimal value of $C_L(s, x)$ for $s \in F_L(x)$, and such that the computation of C_L on inputs x, s is a *good* computation, and let s_0 be the corresponding solution, $\{r_0, \ldots, r_t\} \cup \{w_1, \ldots, w_m\}$ the two associated sets, and R_0, f_0 the corresponding ordering and partial function. Clearly,

$$t + m \leq |a|^{O(1)} << a \ ,$$

where $|a|^{O(1)}$ majorizes the run-time of the machine C_L.

Assume that $s_0 = (y, z_1, \ldots, z_k)$. The element y should witness the validity of one of the conditions 2.-7. above. However, y cannot witness the validity of any of the conditions 2.-4. as the ordering $\prec$ can be extended to a strict linear ordering of $\{0, \ldots, a\}$ that violates these conditions. The element y can satisfy neither 5. nor 7., for all R and f extending R_0 and f_0 respectively, since either $f_0(y)$ is defined and the two conditions fail or we may force failure by putting $f(y) = 0$. To check that y cannot satisfy 6. either, three cases have to be considered: either both $f_0(y)$ and $f_0(f_0(y))$ are defined, or $f_0(y)$ is undefined, or $f_0(y)$ is defined and $f_0(f_0(y))$ is not. In the first case 6. fails. In the second case we may force failure by putting $f(y) = 0$. In the last case $y = r_{t-1}$, $f_0(y) = r_t$ and we may define $f(r_t) = r_{t+1}$, where $r_{t+1} \in \{0, \ldots, a\} \setminus \{r_0, \ldots, r_t, w_1, \ldots, w_m, a\}$ is arbitrary. This is possible as the set is, by the inequality above, non-empty.

To prove the second part of the theorem consider the formula $\Phi(u)$:

$$\exists x \leq u \forall y \leq u,\ x \prec f(x) \wedge f(x) > u \wedge\ \neg f(x) \prec f(y)$$

and assume that the conjunction of the first seven conditions of the formula $Iter(a, R, f)$ is valid. Then $\Phi(0)$, by conditions 2. and 6., $x := 0$ being the witness. Also:

$$\Phi(u) \to \Phi(u+1)$$

since either there is an x that witnesses $\Phi(u)$ and so $\Phi(u+1)$ too, or $u+1$ does witness $\Phi(u+1)$.

Since the formula Φ is $\Sigma_2^b(R, f)$, $T_2^2(R, f)$ proves $\Phi(a)$. The second part of the theorem follows, since the witness to $\Phi(a)$ violates condition 8.

q.e.d.

Definition 6.3 *A* generalized iteration *problem (GI - problem) is given by polynomial - time functions $g(x)$ and $f(x, y)$ and a polynomial - time relation $y \prec_x z$ satisfying:*

1. *$\prec_x$ is a strict linear ordering of $\{0, \ldots, g(x) - 1\}$*
2. *0 is the $\prec_x$-minimum and $0 \prec_x f(x, 0)$*
3. *for all $y < g(x)$:*

$$y \prec_x f(x, y) \to f(x, y) \prec_x f(x, f(x, y))\ .$$

The search task is : given an instance x find $y < g(x)$ such that $f(x, y) \geq g(x)$.

The second part of Theorem 6.2 implies that every GI - problem is Σ_1^b - definable in T_2^2. We do not know whether (a variant of) the class of GI - problems characterizes the search problems that are Σ_1^b - definable in T_2^2.

Acknowledgement:

The first author wish to thank the Czech Academy of Sciences for the generous hospitality and the Logic Group therein for making him feel very welcome in the beautiful city of Prague.

References

[1] Ajtai, M. (1983) Σ^1_1 - formulae on finite structures, *Annals of Pure and Applied Logic*, **24** : 1-48.

[2] Buss, S. R. (1986) *Bounded Arithmetic.* Naples, Bibliopolis.

[3] —— (1987) The propositional pigeonhole principle has polynomial size Frege proofs, *J. Symbolic Logic*, **52**: 916-927.

[4] ——, (1990) Axiomatizations and conservation results for fragments of bounded arithmetic, in: *Logic and Computation*, Contemporary Mathematics, **106**:57-84. Providence, American Mathematical Society.

[5] Buss, S. R., and Krajíček, J. (1994) An application of boolean complexity to separation problems in bounded arithmetic, *Proceedings of the London Mathematical Society*, **69(3)**:1-27.

[6] Buss, S. R., Krajíček, J., and Takeuti, G. (1993) On provably total functions in bounded arithmetic theories R^i_3, U^i_2 and V^i_2, in: *Arithmetic, Proof Theory and Computational Complexity*, eds. P. Clote and J. Krajíček, pp.116-161, Oxford. Oxford University Press.

[7] Chiari, M., and Krajíček, J. (1995) Lifting independence results in bounded arithmetic, submitted.

[8] Cook, S A. (1971) The complexity of theorem proving procedures, in: *Proc. 3rd Annual ACM Symp. on Theory of Computing*, pp. 151-158. ACM Press.

[9] —— (1975) Feasibly constructive proofs and the propositional calculus, in: *Proc. 7th Annual ACM Symp. on Theory of Computing*, pp. 83-97. ACM Press.

[10] Cook, S. A., and Reckhow, A. R. (1979) The relative efficiency of propositional proof systems, *J. Symbolic Logic*, **44(1)**:36-50.

[11] Furst, M., Saxe, J., B., and Sipser, M. (1984) Parity, circuits and the polynomial-time hierarchy, *Math. Systems Theory*, **17**: 13–27.

[12] Hastad, J. (1989) Almost optimal lower bounds for small depth circuits. in: *Randomness and Computation*, ed. S.Micali, Ser.Adv.Comp.Res. **5**: 143-170. JAI Press.

[13] Krajíček, J. (1992) No counter-example interpretation and interactive computation, in: *Logic From Computer Science*, Proceedings of a Workshop held November 13-17, 1989 in Berkeley, ed. Y.N.Moschovakis, *Mathematical Sciences Research Institute Publication*, **21**: 287-293. New York. Springer-Verlag.

[14] ——— (1993) Fragments of bounded arithmetic and bounded query classes, *Transactions of the A.M.S.*, **338(2)** : 587-598.

[15] ——— (1995) *Bounded arithmetic, propositional logic, and complexity theory*, Encyclopedia of Mathematics and Its Applications, Vol. **60**, *Cambridge University Press*, Cambridge - New York - Melbourne, 343 p.

[16] Krajíček, J., and Pudlák, P. (1990) Quantified propositional calculi and fragments of bounded arithmetic, *Zeitschrift f. Mathematikal Logik u. Grundlagen d. Mathematik*, **36**: 29-46.

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

[18] Krajíček, J.,Pudlák, P. and Woods, A. (1995) Exponential lower bound to the size of bounded depth Frege proofs of the pigeonhole principle, *Random Structures and Algorithms*, **7(1)**: 15-39.

[19] Papadimitriou, C. H. (1990) On graph-theoretic lemmata and complexity classes (extended abstract), in: *Proceedings of the 31st IEEE Symposium on Foundations of Computer Science (Volume II)*, IEEE Computer Society, pp. 794–801.

[20] Papadimitriou, C. H., and Yannakakis, M. (1988) Optimization, approximation and complexity classes, in: *20th Annual ACM Symp. on Th. of Computing*, pp.229-234. ACM Press.

[21] Parikh, R. (1971) Existence and feasibility in arithmetic, *Journal of Symbolic Logic*, **36**:494-508.

[22] Paris, J., and Wilkie, A. J. (1985) Counting problems in bounded arithmetic, in: *Methods in Mathematical Logic*, LNM 1130, pp.317-340. Springer-Verlag.

[23] ——— (1987) On the scheme of induction for bounded arithmetic formulas, *Annals of Pure and Applied Logic*, **35**: 261-302.

[24] Paris, J. B., Wilkie, A. J., and Woods, A. R. (1988) Provability of the pigeonhole principle and the existence of infinitely many primes, *Journal of Symbolic Logic*, **53**: 1235–1244.

[25] Pitassi, T., Beame, P., and Impagliazzo, R. (1992) Exponential lower bounds for the pigeonhole principle, *Computational Complexity*, **3**: 97-208.

[26] Pudlák, P. (1992) Some relations between subsystems of arithmetic and the complexity of computations, in :*Logic From Computer Science*, Proceedings of a Workshop held November 13-17, 1989 in Berkeley, ed.Y.N.Moschovakis, *Mathematical Sciences Research Institute Publication*, **21**: 499-519. Springer-Verlag.

[27] Riis, S. (1993) Making infinite structures finite in models of second order bounded arithmetic, in: *Arithmetic, Proof Theory and Computational Complexity*, eds. P. Clote and J. Krajíček, pp.289-319. Oxford. Oxford University Press.

[28] Smullyan, R. (1961) *Theory of Formal Systems*, Annals of Mathematical Studies, **47**. Princeton. Princeton University Press.

[29] Takeuti, G. (1993) RSUV isomorphism, in: *Arithmetic, Proof Theory and Computational Complexity*, eds. P. Clote and J. Krajíček, pp.364-386. Oxford. Oxford University Press.

[30] Yao, Y. (1985) Separating the polynomial-time hierarchy by oracles, in: *Proc. 26th Ann. IEEE Symp. on Found. of Comp. Sci.*, pp. 1-10.