Bounded theories for polyspace computability

Ricardo Bianconi, Gilda Ferreira, Emmanuel Silva∗

Abstract. We present theories of bounded arithmetic and weak analysis whose provably total functions (with appropriate graphs) are the polyspace computable functions. More precisely, inspired in Ferreira’s systems PTCA, $\Sigma^b_2$-NIA and BTFA in the polytime framework, we propose analogue theories concerning polyspace computability. Since the techniques we employ in the characterization of \textsc{PSPACE} via formal systems (e.g. Herbrand’s theorem, cut-elimination theorem and the expansion of models) are similar to the ones involved in the polytime setting, we focus on what is specific of polyspace and explains the lift from \textsc{PTIME} to \textsc{PSPACE}.

Mathematics Subject Classification (2010). Primary 68Q15, 03F30, 03F35, 03D15; Secondary 03F03, 03F05.


1. Introduction

Close connections between bounded theories of arithmetic and computational complexity classes have been established in the mid 1980’s by Samuel Buss. In [2], Buss introduces the weak formal systems $S^1_2$, $U^1_2$ and $V^1_2$, whose provably total functions (with appropriate graphs) are respectively the classes \textsc{PTIME}, \textsc{PSPACE} and \textsc{EXPTIME}. The idea is that weak (subexponential) theories can be used to analyze complexity-theoretic questions. For more on bounded arithmetic and related work see [21, 20, 22, 4, 18].

A few years later [8, 9], Fernando Ferreira presents alternative characterizations of polytime computability via formal systems. Among the systems proposed we highlight the theory $\Sigma^b_2$-NIA, which corresponds to Buss’s system $S^1_2$ in a binary notation framework (see [17]), and the system PTCA which allows induction for polynomial time decidable predicates.

∗The second author acknowledges support of FCT-Fundaçao para a Ciência e a Tecnologia (project PTDC/MAT/104716/2008 and grant SFRH/BPD/34527/2006), Centro de Matemática e Aplicações Fundamentais da Universidade de Lisboa (PEst-OE/MAT/UI0209/2011) and is also grateful to Núcleo de Investigação em Matemática da Universidade Lusófona.
Following this line of research (weak theories in binary notation characterizing poly-time computability) and responding to a challenge of Wilfried Sieg (find a subsystem for analysis whose provably recursive functions consists only of the computationally feasible ones), Ferreira introduces in [10] the theory $BTFA$. $BTFA$ is proposed as a base theory in the reverse mathematics’s style, playing, in the polytime framework, the role that $RCA_0$ plays in the original reverse mathematic’s setting (see [26]). While $PTCA$ and $\Sigma^b_1$-$NIA$ are first-order theories, since $BTFA$ is designed to the formalization of analysis, it is a second-order system, being able to deal not only with binary words but with sets of words. For the formalization of analysis in $BTFA$ see [6].

In the present paper we focus in polyspace computability. The goal is to present theories of bounded arithmetic and weak analysis whose provably total functions (with appropriate graphs) are the polyspace computable functions. Since the study in the polyspace framework is strongly inspired and guided by Ferreira’s work in the polytime setting, we give special emphasis to what is specific of $PSPACE$ and refer to Ferreira’s work in the common parts.

The paper is organized as follows.

In Section 2, we recall a recursion-theoretic characterization of $PSPACE$ and introduce the theory $PSCA$ proving, via Herbrand’s theorem, that its provably total functions are the polyspace computable functions. The idea is that $PSCA$ follows the informal correspondence:

$$\begin{align*}
PTCA & \sim PSPACE, \\
PTIME & \sim PSPACE.
\end{align*}$$

In the next section, we introduce a second system designed to capture polyspace computability, $\Sigma^b_1$-$NIA$. The idea is to avoid the need for function symbols for each polyspace computable function (as in $PSCA$), reproducing in the new setting the simplification of language achieved in the polytime framework with the move from $PTCA$ to $\Sigma^b_1$-$NIA$. Graphically, we have the correspondence:

$$\begin{align*}
\Sigma^b_1$-$NIA & \sim \Sigma^b_1$-$NIA, \\
PTCA & \sim PSCA.
\end{align*}$$

While $\Sigma^b_1$-$NIA$ is a first-order theory, the theory $\Sigma^b_1$-$NIA$ is a second-order bounded system. As discussed in the section, the latter theory has to be able to deal with (bounded) sets of words in order to capture functions described by bounded recursion. Note that the class $PSPACE$ is closed under the bounded recursion scheme. The strategy to prove that $\Sigma^b_1$-$NIA$ corresponds to polyspace computability is similar to the one used by Buss to prove an homologous result (in unary notation) relatively to $U_2$: an application of partial cut-elimination.

The enrichment of $\Sigma^b_1$-$NIA$ with a bounded collection scheme keeping the connection with polyspace computability is done in Section 4 also via proof-theoretic means. The importance of the new scheme becomes visible in the subsequent section when joining recursive comprehension to the system: a fundamental scheme to the development of analysis.
In Section 5, we finally present a theory rich enough to develop analysis keeping the connection with polyspace computability. We named the theory BTPSA and were inspired in the following relation:

\[
\frac{BTFA}{\Sigma_1^1 \text{-NIA}} \sim \frac{BTPSA}{\Sigma_1^P \text{-NIA}}.
\]

The principal features of BTPSA (and of other weak theories for analysis) are a second-order language (with unbounded second-order variables) and a recursive comprehension scheme, central in expressing, for instance, the existence of particular real numbers and in the formalization of basic analytic concepts such as continuous real functions or sequences of real numbers.

In the recent paper [7], a general blueprint for the construction of theories for analysis connected with computational complexity classes is presented and is illustrated in the polytime framework. This work can be seen as the implementation of the general blueprint in the polyspace setting.

The results of this paper are part of the master thesis [15]. The material in Section 2 was, in addition to [15], also presented (in a slight different formulation) in the PhD dissertation [24]. Since both dissertations are written in Portuguese, we found it useful to make the work visible to the logic community through this article.

2. A first-order arithmetic theory for polyspace computability

In this section, we start by recalling an alternative characterization of \( \text{PSPACE} \), the well-known computational complexity class usually defined by limiting the amount of space available (polynomial space) in a deterministic Turing Machine. The inductive characterization of \( \text{PSPACE} \) we present, is essentially the one introduced in [23] and will be useful in what follows. Since it is written in binary notation, we start introducing some operations. Let \( 2^{<\omega} \) (also known as \( \{0, 1\}^* \)) be the set of all finite sequences of 0’s and 1’s. The empty sequence is denoted by \( \epsilon \). For \( x \) and \( y \) elements in \( 2^{<\omega} \), \( x \hat{y} \) represents the concatenation of \( x \) by \( y \) (we usually omit the symbol \( \hat{\} \) and just write \( xy \)); \( x \subseteq y \) means that \( x \) is an initial subword of \( y \) (string prefix); \(|x|\) denotes the length of \( x \), i.e. the number of 0’s and 1’s in the word \( x \); \( x \mid y \) is the truncation of \( x \) by \( y \) defined by \( x \mid y := \begin{cases} x, & \text{if } |x| \leq |y| \\ z, & \text{if } z \subseteq x \land |z| = |y| \end{cases} \); \( x \times y \) is the product of \( x \) by \( y \) defined as being the word \( x \) concatenated with itself length of \( y \) times; \( x \leq y \) (respectively \( x \equiv y \)) abbreviates \( 1 \times x \leq 1 \times y \) (respectively \( 1 \times x = 1 \times y \)) meaning that the length of \( x \) is less than or equal (respectively equal) to the length of \( y \); and \( \leq \) is the linear order defined by \( x \leq y :\Rightarrow (x \leq y \land \neg(x \equiv y)) \lor (x \equiv y \land \exists z (z \subseteq x \land z \subseteq y)) \lor (x = y) \), i.e. it is defined first according to length and then, within the same length, lexicographically.

**Definition 2.1.** \( \text{PSPACE} \) is the smallest class of functions that includes the initial functions:

\[(1) \quad C_0(x) = x0\]
(2) \( C_1(x) = x1 \)

(3) \( P^i_j(x_1, \ldots, x_n) = x_i, \text{ for } 1 \leq i \leq n \)

(4) \( Q(x, y) = \begin{cases} 1, & \text{if } x \subseteq y \\ 0, & \text{otherwise} \end{cases} \)

and is closed under the following schemes:

- composition
  \[ f(\bar{x}) = g(h_1(\bar{x}), \ldots, h_k(\bar{x})) \]

- bounded recursion on notation
  \[
  \begin{align*}
  f(\bar{x}, 0) &= h_0(\bar{x}, y, f(\bar{x}, y))_{h(x,y)} \\
  f(\bar{x}, 1) &= h_1(\bar{x}, y, f(\bar{x}, y))_{h(x,y)},
  \end{align*}
  \]
  where \( t \) is a bounding function\(^1\), i.e. \( t \) belongs to the smallest class of functions that includes \( \epsilon, 0, 1, ^\land, ^\times, P^n_j \) and is closed under composition

- bounded recursion
  \[
  \begin{align*}
  f(\bar{x}, \epsilon) &= g(\bar{x}) \\
  f(\bar{x}, S(y)) &= h(\bar{x}, y, f(\bar{x}, y))_{h(x,y)},
  \end{align*}
  \]
  where \( t \) is a bounding function and \( S \) is the successor function defined by \( S(\epsilon) = 0, S(x0) = x1, S(x1) = S(x)0. \)

Note that the last scheme is essential to capture polyspace computability. Removing the scheme of bounded recursion in the definition above we obtain exactly a characterization of \( \text{PTIME} \). See [8].

Let \( \mathcal{L} \) be the first-order language which has three constant symbols \( \epsilon, 0 \) and \( 1 \), two binary function symbols \( ^\land \) and \( ^\times \) (intended to be interpreted respectively as \textit{concatenation} and \textit{product} in the standard model) and two binary relation symbols \( = \) and \( \subseteq \) (for \textit{equality} and \textit{initial subwordness} respectively). The domain of the intended standard model of the language is \( 2^{\omega_1}. \) Let \( \mathcal{L}_{PS} \) be an extension of the former language by adding a function symbol for each description of a polyspace computable function according to Definition 2.1.

**Definition 2.2.** The class of \textit{polyspace decidable matrices} is the smallest class of formulas of \( \mathcal{L}_{PS} \) containing the atomic formulas and closed under the Boolean operations and quantifications of the form \( \forall x (x \leq t \rightarrow \ldots) \) or \( \exists x (x \leq t \land \ldots) \), where \( t \) is a term of \( \mathcal{L}_{PS} \) where \( x \) does not occur.

**Definition 2.3.** \( \text{PSCA} \) (acronym for \textit{Polynomial Space Computable Arithmetic}) is the first-order theory, in the language \( \mathcal{L}_{PS} \), which has the following axioms:

- **Basic axioms**

\(^1\)The bounding functions ensure that the recursion scheme does not produce functions with exponential growth.
Bounded theories for polysize computability

\[ x \varepsilon = x, \; x(0) = (xy)0 \quad \text{and} \quad x(1) = (xy)1; \]
\[ x \times \varepsilon = \varepsilon, \; x \times y0 = (x \times y)x \quad \text{and} \quad x \times y1 = (x \times y)x; \]
\[ x \subseteq \varepsilon \leftrightarrow x = \varepsilon, \; x \subseteq y0 \leftrightarrow x \subseteq y \lor x = y0 \quad \text{and} \quad x \subseteq y1 \leftrightarrow x \subseteq y \lor x = y1; \]
\[ x0 = y0 \rightarrow x = y \quad \text{and} \quad x1 = y1 \rightarrow x = y; \]
\[ x0 \neq y1, \; x0 \neq \varepsilon \quad \text{and} \quad x1 \neq \varepsilon; \]

- **Defining axioms**
  
  a. Initial functions
  
  1. \( C_0(x) = x0 \)
  
  2. \( C_1(x) = x1 \)
  
  3. \( P^n(x_1, \ldots, x_n) = x, \) for \( 1 \leq i \leq n \)
  
  4. \( Q(x, y) = 1 \leftrightarrow x \subseteq y; \; Q(x, y) = 0 \lor Q(x, y) = 1 \)

  b. Derived functions
  
  1. \( f(\bar{x}) = g(h_1(\bar{x}), \ldots, h_k(\bar{x})), \)
    if \( f \) is the description of the composition from \( g, h_1, \ldots, h_k \)
  
  2. \( f(\bar{x}, \varepsilon) = g(\bar{x}) \)
    \( f(\bar{x}, y0) = h_0(\bar{x}, y, f(\bar{x}, y))_{\bar{h}t1}, \)
    \( f(\bar{x}, y1) = h_1(\bar{x}, y, f(\bar{x}, y))_{\bar{h}t1}, \)
    where \( t \) is a term of the language \( \mathcal{L} \) and \( f \) is the description of the bounded recursion on notation defined from \( g, h_0, h_1 \) and \( t \)
  
  3. \( f(\bar{x}, \varepsilon) = g(\bar{x}) \)
    \( f(\bar{x}, S(y)) = h(\bar{x}, y, f(\bar{x}, y))_{\bar{h}t1}, \)
    where \( t \) is a term of \( \mathcal{L} \), \( S \) is the successor function and \( f \) is the description of the bounded recursion defined from \( g, h \) and \( t \)

- **Scheme of induction on notation**

  \[ A(\varepsilon) \land \forall x (A(x) \rightarrow A(x0) \land A(x1)) \rightarrow \forall x A(x), \]

  where \( A \) is a polysize decidable matrix, possibly with other free variables besides \( x \).

  The theory \( \text{PSCA} \) described above was inspired in \( \text{PTCA} \) (Polynomial Time Computable Arithmetic), a theory introduced by Ferreira in [8] designed to correspond to polytime computability. The novelty in \( \text{PSCA} \) is that the scope of the induction on notation scheme extends from polytime decidable matrices to \( \text{polysize} \) decidable matrices and in the derived functions we add an extra scheme for **bounded recursion**.

  Note that the scheme of slow induction

  \[ A(\varepsilon) \land \forall x (A(x) \rightarrow A(S(x))) \rightarrow \forall x A(x) \]

  (which in unary notation corresponds to the usual \(+1\)-induction) is valid in \( \text{PSCA} \) for \( A \) a polysize decidable matrix possibly with other free variables besides \( x \). The proof is entirely similar to the one presented in [8], page 53.
Having in view to show that PSCA corresponds to the computational complexity class PSPACE, i.e. to polysize computability, we start by arguing that PSCA is a universal theory. Since it is not visible from the above formulation because of the induction scheme, our strategy is to use a well-known result of Łoś and Tarski that ensures that if a theory is preserved by substructures then it is a universal theory.

In order to prove that PSCA is preserved by substructures we need some auxiliary results.

**Lemma 2.4.** Given $A_1(\bar{x}), \ldots, A_n(\bar{x})$ polysize decidable matrices and $f_1(\bar{x}), \ldots, f_{m+1}(\bar{x})$ function symbols, there is a function symbol $f(\bar{x})$ such that

\[ \text{PSCA} \vdash (A_1(\bar{x}) \land f(\bar{x}) = f_1(\bar{x})) \lor \ldots \lor (A_n(\bar{x}) \land f(\bar{x}) = f_{m+1}(\bar{x})). \]

The result above shows that PSCA allows the definition of functions by cases, being the cases expressed by polysize decidable matrices.

**Proof.** With the extra assumption that for the polysize decidable matrices $A_i$ there are function symbols $K_{A_i}$ such that $\text{PSCA} \vdash (A_i(\bar{x}) \rightarrow K_{A_i}(\bar{x}) = 1) \land (\neg A_i(\bar{x}) \rightarrow K_{A_i}(\bar{x}) = 0)$, Lemma 2.4 can easily be proved by induction on $n$. For $n = 1$ just take $f(\bar{x}) = h(f_1(\bar{x}), f_1(\bar{x})K_{A_1}(\bar{x}))$ with $h(y, e) = y$, $h(y, x0) = y$ and $h(y, x1) = x$.

Thus, it just remains to argue that polysize decidability matrices can be expressed in PSCA by means of quantifier-free formulas, more precisely for each polysize decidable matrix $A$ there is a function symbol $K_A$ in $\mathcal{L}_{PS}$ such that $\text{PSCA} \vdash (A(\bar{x}) \rightarrow K_A(\bar{x}) = 1) \land (\neg A(\bar{x}) \rightarrow K_A(\bar{x}) = 0)$.

The proof can be done by induction on the complexity of $A$. For $A$ an atomic formula we take $K_A$ as being $Q$ and $K_a(x, y)$ as being $Q(11, Q(x, y)Q(y, x))$. We define $K_{aB} := K_a(0, K_B)$ and $K_{abc} := K_a(11, K_BK_C)$. Let $A(\bar{x}, x)$ be the formula $\forall y (y \leq x \rightarrow B(\bar{z}, y))$.

Take $f(\bar{z}, e) = K_B(\bar{z}, e)$ and

\[ f(\bar{z}, S(x)) = \begin{cases} 
1 & \text{if } K_B(\bar{z}, S(x)) = 1 \\
0 & \text{if } K_B(\bar{z}, S(x)) = 0.
\end{cases} \]

By slow induction on $x$, it is easy to prove that $\forall y (y \leq x \rightarrow B(\bar{z}, y)) \iff f(\bar{z}, x) = 1$. So, since $y \leq x \iff y \leq 1 \times x$, we have $\forall y (y \leq x \rightarrow B(\bar{z}, y)) \iff f(\bar{z}, 1 \times x) = 1$. The result follows taking $K_A(\bar{z}, x)$ as being $K_a(1, f(\bar{z}, 1 \times x))$.

\[ \square \]

**Lemma 2.5.** For each polysize decidable matrix $A(\bar{z}, x)$ there is a function symbol $g$ in $\mathcal{L}_{PS}$ such that $\text{PSCA} \vdash (\exists y \leq x A(\bar{z}, y)) \rightarrow g(\bar{z}, x) \leq x \land A(\bar{z}, g(\bar{z}, x))$.

**Proof.** Take $f(\bar{z}, e) = \begin{cases} 
\epsilon & \text{if } A(\bar{z}, e) \\
1 & \text{otherwise}
\end{cases}$ and

\[ f(\bar{z}, S(y)) = \begin{cases} 
g(\bar{z}, y) & \text{if } f(\bar{z}, y) \leq y \\
S(y) & \text{if } f(\bar{z}, y) \not\leq y \land A(\bar{z}, S(y)) \\
S(y)1 & \text{if } f(\bar{z}, y) \not\leq y \land \neg A(\bar{z}, S(y)).
\end{cases} \]

As a bounding function we can take $1 \times y11$. Let $g(\bar{z}, x)$ be (by definition) $f(\bar{z}, 1 \times x)$. Let us prove that $g$ satisfies the statement in the lemma. It can be proved, by slow induction on $y$ that $f(\bar{z}, y) \leq y \rightarrow A(\bar{z}, f(\bar{z}, y))$, thus we have that $(\forall) g(\bar{z}, y) \leq y \rightarrow A(\bar{z}, g(\bar{z}, y))$. Also,
by slow induction, this time on $x$, we can prove that $(\exists y \leq x A(\bar{z}, y)) \rightarrow f(\bar{z}, x) \leq x$. Easily from the above statement we can prove $(\exists y \leq x A(\bar{z}, y)) \rightarrow g(\bar{z}, x) \leq x$. The result follows from the last assertion and (†).

Let $M$ be a model of PSCA and $\mathcal{N}$ a substructure of $M$. From Lemma 2.5 we can easily argue that the polyspace decidable matrices are absolute between $\mathcal{N}$ and $M$. The argument used in [9] page 143, in the polytime setting, adapts trivially to the present context. To conclude that PSCA is preserved by substructures, we only need to check that induction on notation also holds in $\mathcal{N}$. Since the induction scheme can be reformulated in the following alternative way:

$$A(\epsilon) \land \forall x \subseteq a (A(x) \rightarrow A(x0) \land A(x1)) \rightarrow A(a),$$

by absoluteness of the polyspace decidable matrices, the induction on notation axioms hold in $\mathcal{N}$.

Therefore, since PSCA is a universal theory, applying the Herbrand Theorem and Lemma 2.4, immediately we conclude that:

**Theorem 2.6.** If PSCA $\vdash \forall \bar{x} \exists y A(\bar{x}, y)$ with $A$ a polyspace decidable matrix and $\bar{x}$ and $y$ its only free variables then, there is a function symbol $f$ in $\mathcal{L}_{PS}$ such that PSCA $\vdash \forall \bar{x} A(\bar{x}, f(\bar{x})).$

It is in this precise sense - the provably total functions of PSCA with polyspace graphs are exactly the functions of PSPACE - that we say that PSCA corresponds to polyspace computability.

### 3. A second-order bounded arithmetic theory for polyspace computability

It is known that it is possible to introduce all primitive recursive functions in $\Sigma^0_1$-IND (see the incompleteness paper of Gödel) and it is possible to introduce all the polytime computable functions in $\Sigma^0_1$-NIA (see [9]). As a consequence, PRA can be consider a subtheory of $\Sigma^0_1$-IND and PTCA a subtheory of $\Sigma^1_1$-NIA.

The goal of this section is to present a theory which plays, in the polyspace setting, the role played by $\Sigma^0_1$-IND and $\Sigma^1_1$-NIA in the primitive recursive and polytime settings respectively. More precisely, we are looking for a theory still characterizing polyspace computability but in a more economic language not having all descriptions of PSPACE functions as primitive.

While $\Sigma^0_1$-NIA corresponds to Buss’s system $S^1_2$, the theory $\Sigma^1_{1,b}$-NIA we are going to introduce is inspired in Buss’s system $U^1_{3,2}$. Contrarily to what happens with polytime computability (where a first-order language is enough), to introduce functions described by bounded recursion we need $\Sigma^1_{1,b}$-NIA to be a second-order (bounded) theory. Note that, in a system that does not prove the totality of exponentiation, not every bounded set is given by a binary word. The introduction of second-order bounded variables is by no
means a novelty, Buss in [2] uses second-order bounded variables in the unary notation context.\footnote{We thank the anonymous referee for calling our attention to two related works: (i) Skelley [25] gives a characterization of \textsc{Pspace} via a third-order (bounded) theory; (ii) Kołodziejczyk, Nguyen and Thapen [19] use a strategy similar to ours to express \textsc{Pspace} computability in their framework.}

We start introducing the second-order language and some notation.

Let $L^2_b$ be the second-order language obtained from $L$ by adding second-order bounded variables and the relation symbol $\in$ which infixes between a term of $L$ and a second-order bounded variable. The standard structure for this language has domain $(2^{<\omega}, \mathcal{P}(2^{<\omega}))$, i.e., the first-order variables are interpreted as finite sequences of zeros and ones, and the second-order (bounded) variables are subsets $X'$ of $2^{<\omega}$, with any term of $L$, satisfying $x \in X' \rightarrow x \leq t$.

The terms in $L^2_b$ coincide with the terms in $L$ and the class of formulas in $L^2_b$ can be defined as the smallest class of expressions containing the atomic formulas $t_1 \leq t_2$, $t_1 = t_2$, $t_1 \in X'$, and closed under the Boolean operations, the first-order quantifications $\forall x, \exists x$, the first-order bounded quantifications $\forall t \leq t$, $\exists x \leq t$ and the second-order bounded quantifications $\forall X', \exists X'$. Note that in $L^2_b$, $\forall x \leq t A$ and $\exists x \leq t A$ are treated as new formulas and not as mere abbreviations for $\forall x (x \leq t \rightarrow A)$ and $\exists x (x \leq t \land A)$ respectively. It is a technical detail that contributes for an efficient formulation of sequent calculus.

A $\Sigma^{1,b}_1$-formula (respectively $\Pi^{1,b}_1$-formula) is a formula in the language $L^2_b$ of the form: $\exists X_1^{b_1} \ldots \exists X_k^{b_k} A$ (respectively $\forall X_1^{b_1} \ldots \forall X_k^{b_k} A$), where $A$ is a $\Sigma^{1,b}_0$-formula (i.e. with no quantifications of second-order and where all the first-order quantifications are bounded. It may have first and second-order parameters). In the standard model, if the second-order parameters are in the Polynomial Hierarchy (a.k.a. Meyer-Stockmeyer Hierarchy) then the $\Sigma^{1,b}_0$-formulas define predicates in this hierarchy. An extended $\Sigma^{1,b}_1$-formula (respectively extended $\Pi^{1,b}_1$-formula) is a formula that can be built in a finite number of steps, starting with $\Sigma^{1,b}_0$-formulas and allowing conjunctions, disjunctions, first-order bounded quantifications and second-order bounded existential (respectively universal) quantifications.

**Definition 3.1.** $\Sigma^{1,b}_1$-\textsc{Nia} is the second-order theory in the language $L^2_b$, which has the following axioms:

- **Basic axioms**\footnote{The 14 basic axioms of Definition 2.3.},
- $\forall x \forall X' (x \in X' \rightarrow x \leq t)$, with $t$ a term where $x$ does not occur,
- **Bounded comprehension:** $\exists X' \forall x \leq t (x \in X' \leftrightarrow A(x))$, where $t$ is a term in which $x$ does not occur, and $A$ is a $\Sigma^{1,b}_0$-formula that may have other free variables other than $x$ and where the variable $X'$ does not occur,
- **Induction on notation for $\Sigma^{1,b}_1$-formulas:**
  \[ A(e) \land \forall x (A(x) \rightarrow A(x0) \land A(x1)) \rightarrow \forall x A(x), \]
  with $A$ a $\Sigma^{1,b}_1$-formula possibly with free variables other than $x,$
• Replacement for $\Sigma^0_{1,b}$-formulas: $\forall x \leq t \exists X^y A(x, X^y) \rightarrow \exists y^y \forall x \leq t A(x, y^y)$, with $A$ a $\Sigma^0_{1,b}$-formula, $t$ a term where $x$ does not occur, and $\bar{A}$ results from $A$ by replacing all the occurrences of '$s' in $X^y$ by '$(x,s)$ in $y^y$ (where $(,)$ is a pairing function and $r$ is a certain term depending on $t$ and $q$). We are omitting the exact term $r$ in order to facilitate reading (the term depends on the particular definition of the pairing function – see [15] for a concrete implementation of these matters). This is a technical axiom that permits a kind of “permutation” between bounded first-order universal quantifications and bounded second-order existential quantifications.

Applying the replacement scheme, it is straightforward to see that an extended $\Sigma^1_{1,b}$-formula can be expressed via a $\Sigma^1_{1,b}$-formula. Thus, contrarily to the formulation of Buss’s theory $U$, where the presence of extended $\Sigma^1_{1,b}$-formulas in the induction scheme is essential, with the replacement scheme available we can disregard extended formulas in the formulation of $\Sigma^1_{1,b}$-NIA, simplifying subsequent arguments by cut-elimination.

Later, in Section 5, we will need a stronger form of comprehension still available in $\Sigma^1_{1,b}$-NIA. More precisely, we will use the fact that

$$\forall x (A(x) \leftrightarrow B(x)) \rightarrow \forall w \exists X^w \forall x \leq w (x \in X^w \leftrightarrow A(x)),$$

with $A$ an extended $\Sigma^1_{1,b}$-formula and $B$ an extended $\Pi^1_{1,b}$-formula, is derivable in $\Sigma^1_{1,b}$-NIA. We call the previous scheme $\Lambda^1_{1,b}$-bounded comprehension. The details of this quite technical result are available in [15], pages 32–34.

Next we argue that PSPACE is a subtheory of $\Sigma^1_{1,b}$-NIA.

It is possible to present for each $f(\bar{x}) \in L_{PS}/L$ a $\Sigma^1_{1,b}$-formula $F_f(\bar{x}, y)$ in the language $L_2$ and a term $b_f$ in $L$ such that $\Pi^1_{1,b}$-NIA + $\forall \bar{x} \exists y \leq b_f(\bar{x}) F_f(\bar{x}, y)$ and which has the defining properties of $f$ (as given by Definition 2.3)4. The proof is by induction on the complexity of the description of $f$. If $f$ is an initial function or is defined by composition or bounded recursion on notation see the proof presented in [9], pages 150–151, in the context of PTIME. What is new, concerning PSPACE, is the bounded recursion scheme. It remains to prove that if $f$ is the description of bounded recursion defined from $g$, $h$ and $t$, there is a $\Sigma^1_{1,b}$-formula $F_f(\bar{x}, y, z)$ and a term $b_f(\bar{x}, y)$ in the conditions above satisfying, in particular, $\Pi^1_{1,b}$-NIA + $F_f(\bar{x}, z) \rightarrow F_f(\bar{x}, \epsilon, z)$ and $\Pi^1_{1,b}$-NIA + $F_f(\bar{x}, y, r) \land F_f(\bar{x}, y, r, u) \land z = u_{b_f(\bar{x})} \rightarrow F_f(\bar{x}, S(y), z)$. The details can be found in [15], pages 35–40. Here we just give some intuition. In the case of bounded recursion on notation, to express the value of the function on $y$, we need the values of the function on the initial subwords of $y$ (thinking in terms of binary trees we are going through a path till reaching the initial node). With appropriate coding, it is possible to collect all this information in a word bounded by a term (depending on $y$ and on the bounding term $t$ of the function). However, in the case of bounded recursion, the above strategy is no longer possible since to determine the value of the function on $y$, we need the values of the function in all words $x \leq y$ (in terms of binary trees we are going down the words of the same length starting in $y$ till reaching the initial node). An attempt to code the information as before would result on exponential size. This is the reason why $\Sigma^1_{1,b}$-NIA is formulated in a language expressive enough to denote

4By $3^1$ we mean “there is one and only one”, i.e. $\Sigma^1_{1,b}$-NIA + $F_f(\bar{x}, y) \land F_f(\bar{x}, z) \rightarrow y = z$. 
bounded sets instead of just words. See [15], page 37, for a possible way of collecting all the information on a polynomially bounded set (the set allows the generation of a sequence of blocks, coding exactly the ordered values of the function, by generating 1 or 0 depending on the fact that a word is or is not in the set) and for the precise construction of the \(\Sigma_{1}^{1,b}\)-formula \(F_{f}\) and term \(b_{f}\), with \(f\) a function obtained by bounded recursion from \(g, h\) and \(t\). Informally, \(F_{f}(\bar{x}, y, z)\) will say that there exists a bounded set which generate the blocks as above and such that if \(r\) is the \(y\)th block, then \(z\) is coded in \(r\).

The proof that \(F_{f}\) and \(b_{f}\) satisfy the expected conditions uses the fact that in \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\) is valid slow induction for \(\Sigma_{0}^{1,b}\)-formulas. See the details in [15], pages 33–40.

We can subsume the above discussion - \(\text{PSPACE}\) as a subtheory of \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\) - by the following result:

**Proposition 3.2.** Every model \(M\) of \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\) can be extended to a model \(M'\) of \(\text{PSPACE}\) keeping the first-order domain and the interpretations of symbols of \(L\) and defining the interpretation of each function symbol \(f \in \mathcal{L}_{PS}\backslash L\) by interpreting \(f\) as the function \(\{t(\bar{a}, b) : M \models F_{f}(\bar{a}, b)\}\).

**Proof.** From the study above, by the definition of \(M'\) and of the formulas \(F_{f}\) for every \(f \in \mathcal{L}_{PS}\backslash L\), the basic and defining axioms hold in \(M'\). For the remaining case, the induction scheme, it is enough to show that for every polyspace decidable matrix \(A'((\bar{x})\bar{a})\) there is an extended \(\Sigma_{1}^{1,b}\)-formula \(A((\bar{x})\bar{a})\) such that \(M' \models A'((\bar{a})\bar{a})\) iff \(M \models A((\bar{a})\bar{a})\), for every tuple of parameters \(\bar{a}\) in \(M'\) (see the consideration concerning extended formulas in the beginning of the section). By induction on the complexity of the term, the existence of the previous \(\Sigma_{1}^{1,b}\)-formulas of \(L_{1}\) and terms of \(L\) can be extended from function symbols of \(\mathcal{L}_{PS}\backslash L\) to any term \(t\) of \(\mathcal{L}_{PS}\) having that \(M' \models t(\bar{a}) = b\) iff \(M \models F_{f}(\bar{a}, b)\). Think in \(A'\) (modulo equivalence of formulas) as containing a sequence (possibly empty) of quantifications of the form \(\forall x \leq t(\bar{y})b\) and \(\exists x \leq t(\bar{y})s\) followed by a quantifier-free formula in the conjunctive normal form. The construction of \(A\) from \(A'\) is done in the expected way. For example, if in \(A'\) appears the atomic formula \(t((\bar{x})\bar{a}) \leq q(\bar{a})\) with \(t\) and \(q\) terms of \(\mathcal{L}_{PS}\), in \(A\) we will have \(\exists z \leq b_{t}(\bar{a}),(\exists w \leq b_{q}(\bar{a})(F_{t}(\bar{x}, z) \land F_{q}(\bar{x}, w)) \land z \leq w)\). Negations of atomic formulas, like \(t((\bar{x})\bar{a}) \neq q(\bar{a})\) in \(A'\) will be expressed in \(A\) by \(\exists z \leq b_{t}(\bar{a}) \forall w \leq b_{q}(\bar{a})(F_{t}(\bar{x}, z) \land F_{q}(\bar{x}, w)) \land z \neq w)\). And quantifications of the form \(\forall x \leq t(\bar{y})\) . . . \(\exists x \leq t(\bar{y})\) are replaced by \(\exists z \leq b_{t}(\bar{a})(F_{t}(\bar{x}, z) \land \forall x \leq z)\) or \(\exists z \leq b_{t}(\bar{a})(F_{t}(\bar{x}, z) \land \exists x \leq z)\) respectively.

Similarly to the proof in [2] concerning the theory \(U_{2}\), it can be proved that \(\text{PSPACE}\) is the class of functions provably total in \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\) with \(\Sigma_{1}^{1,b}\)-graphs. The proof uses the free cut elimination theorem, after formulating the theory \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\) into Gentzen’s sequent calculus. We start by presenting the sequent calculus formulation for \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\), denoted by \(\text{LK}_{PS}\).

Besides the initial sequents of the form \(A \Rightarrow A\), with \(A\) an atomic formula, and the sequents for equality, \(\text{LK}_{PS}\) has also the following axioms:

1) \(\Rightarrow A((\bar{x})\bar{a})\), with \(A\) a basic axiom of \(\Sigma_{1}^{1,b}\)-\(\text{NIA}\) and \(\bar{s}\) terms;

2) \(s \in X' \Rightarrow s \leq t;\)
3) \( \exists X \forall y \leq s (y \in X' \leftrightarrow A(y)) \), with \( A \) a \( \Sigma_0^{1,b} \)-formula where the variable \( X' \) does not occur, and besides the usual rules for predicate logic has the following rules for first-order bounded quantifications:

\[
\begin{align*}
\Gamma, t \leq s, \forall x \leq s A(x) & \rightarrow \Delta \\
\Gamma, t \leq s, \forall x \leq s A(x) & \rightarrow \Delta \\
\Gamma, b \leq t, A(b) & \rightarrow \Delta \Gamma \rightarrow \Delta, \forall x \leq tA(x) \\
\Gamma, b \leq t, A(b) & \rightarrow \Delta \Gamma \rightarrow \Delta, \forall x \leq tA(x) \\
\Gamma, x \leq tA(x) & \rightarrow \Delta \Gamma \rightarrow \Delta, \exists x \leq sA(x) \\
\Gamma, x \leq tA(x) & \rightarrow \Delta \Gamma \rightarrow \Delta, \exists x \leq sA(x)
\end{align*}
\]

where \( b \) is an eigenvariable (i.e., it is not free in either \( \Gamma, \Delta \) or \( t \)); has the following rules for second-order bounded quantifications:

\[
\begin{align*}
\Gamma, A(F'^2) & \rightarrow \Delta \\
\Gamma, \forall X' A(X') & \rightarrow \Delta \\
\Gamma, A(C'^2) & \rightarrow \Delta \\
\Gamma, \exists X' A(X') & \rightarrow \Delta \\
\Gamma, A(F'^2) & \rightarrow \Delta \\
\Gamma, \exists X' A(X') & \rightarrow \Delta
\end{align*}
\]

with \( F' \) a second-order variable and \( C' \) a second-order eigenvariable; has the following induction rule:

\[
\begin{align*}
\Gamma, A(x) & \rightarrow \Delta, A(x0) \\
\Gamma, A(x) & \rightarrow \Delta, A(x1)
\end{align*}
\]

with \( A \) a \( \Sigma_1^{1,b} \)-formula (possibly with parameters), \( s \) a term, and \( x \) an eigenvariable; and has the following replacement rule:

\[
\Gamma, x \leq t \rightarrow \Delta, \exists Y' A(x, Y') \\
\Gamma \rightarrow \Delta, \exists Y' \forall x \leq tA(x, Y')
\]

where \( x \) is an eigenvariable, \( A \) is a \( \Sigma_0^{1,b} \)-formula, and \( r \) and \( A \) are as in the replacement scheme.\(^5\)

The point of this sequent calculus formulation of \( \Sigma_1^{1,b} \)-\text{NIA} is that, if \( \text{LK}_{\text{PS}} \vdash \Gamma \rightarrow \Delta \), with \( \Gamma \) and \( \Delta \) formed by \( \Sigma_1^{1,b} \)-formulas, then (by an easy application of the free cut elimination theorem) there is a \( \text{LK}_{\text{PS}} \)-proof of \( \Gamma \rightarrow \Delta \) in which every formula appearing in the proof is a \( \Sigma_1^{1,b} \)-formula. Note that it is the replacement scheme that allows us to have a sequent calculus proof just with \( \Sigma_1^{1,b} \)-formulas instead of extended \( \Sigma_1^{1,b} \)-formulas.

The fundamental theorem to achieve our goal – characterize the provably total functions of \( \Sigma_1^{1,b} \)-\text{NIA} – is the technical result (Theorem 3.4, below), which asserts that there are extended \( \Sigma_1^{1,b} \)-formulas, extended \( \Pi_1^{1,b} \)-formulas and functionals related with polyspace which witness the second-order existential quantifications in the consequents of the sequents in the \( \text{LK}_{\text{PS}} \)-proof above.

\(^5\) A similar replacement rule appears in [1] in a reformulated version of the sequent calculus for \( \Delta_2^1 \).
First some considerations about second-order computations. We denote by $\text{PSPACE}^{\Sigma_b^1}$ the class of functionals computable by a deterministic Turing Machine in polynomial space with oracle $\Sigma_{0}^{1,b}$.6

**Proposition 3.3.** The functionals in the class $\text{PSPACE}^{\Sigma_b^1}$ which are functions, are exactly the polyspace computable functions.

**Proof.** If $f \in \text{PSPACE}$ it is immediate that $f \in \text{PSPACE}^{\Sigma_b^1}$. Just notice that we can consider that the second-order tapes stay empty and the oracle is not called to intervene in the computation. For the other inclusion, let $f$ be a function in $\text{PSPACE}^{\Sigma_b^1}$. So, there is $M_1$ a deterministic polyspace Turing machine with a $\Sigma_{0}^{1,b}$ oracle, say $Q(x,X')$, such that when the input is $\bar{w}$, the output is $f(\bar{w})$. Let $M_2$ be a deterministic Turing machine without second-order tapes (and without oracle), which we initialize with $\bar{w}$ and that executes the same steps as $M_1$, except that, when $M_1$ writes symbols in the first-order oracle tape, $M_2$ writes them in a working tape and when $M_1$ writes symbols in the second-order oracle tape, $M_2$ does nothing. Let us see what to do when $M_1$ asks a question to the oracle for the first time. If $Q(x,X')$ is a first-order formula, since it is bounded, it is decidable in polyspace, thus in $M_2$ we just have to introduce the steps that decide $Q(x)$. If $X'$ occurs in $Q(x,X')$ suppose, for a simple example7, that $Q(x,X')$ is of the form $\exists z_1 \leq q_1(x) \forall z_2 \leq q_2(x) r(x, z_1, z_2) \in X'$. Let us argue that there is a polytime alternating Turing machine that with input $x$ decides $Q(x,X')$. Since it is well-known that: (1) $\text{PSPACE}$ is equivalent to polytime in alternating Turing machines, (2) in these machines there are existential and universal states and (3) $q_1$ and $q_2$ can be computed in $\text{PSPACE}$; if we manage to decide $r \in X'$ in $\text{PSPACE}$ there is an alternating Turing machine that decides $Q(x,X')$ is polytime. From $x, z_1$ and $z_2$ we can compute $r$ in $\text{PSPACE}$. To decide $r \in X'$, we can think in a machine initialized with $\bar{w}$, which acts like $M_1$ but when $M_1$ writes in the oracle first-order tape it writes in a working tape and when $M_1$ writes an element in the oracle second-order tape, the new machine writes in binary notation the number of elements already written in the oracle second-order tape by $M_1$. As in [15], we assume that the pointers in the oracle tapes just move right and exactly a cell at a time. This means that, when the counting number (in the new machine) is $r$, the next element to be written in the oracle second-order tape by $M_1$ informs us if $r$ belongs to $X'$ or not. Read it. If it is 0 then $r \notin X'$, if it is 1 then $r \in X'$. Note that we are assuming that $X'$ is written in the oracle tape in the fixed way described in footnote 6. Therefore, the decision of $r \in X'$ is done in $\text{PSPACE}$. Since we saw that $Q(x,X')$ can be decided from $x$ using a polytime

---

6The idea is that the machine allows first and second-order inputs, returns first and second-order outputs and allows questions to the oracle involving first and second-order parameters. A bounded set is written in a (input, output or oracle) second-order machine tape in the following way: in the $j^{th}$ cell of the tape put 1 if the binary word for $j$ belongs to the set, put 0 otherwise. If the first-order input tapes have length less than or equal to $n$ and the second-order input tapes have length less than or equal to $2^n$, the computation occurs in polynomial space if the number of tape cells used (in the working tapes) is less than or equal to $n^k$, with $k \in \mathbb{N}$. We say that the machine computes a function if it has no inputs nor outputs of second-order (having the possibility of having second-order parameters in the $\Sigma_0^{1,b}$ questions to the oracle). For a full description of this kind of Turing machines see [15], page 16.

7The example with bounded first-order existential and universal quantifications and $X'$ illustrates what happens when the $\Sigma_0^{1,b}$-oracle is call to intervene.
Suppose that:  

\[ \theta \] being conditions. For the sequent \( \forall \) every time \( \Delta \) \( A \) detailed proof see [15], pages 50–55.

Suppose that \( P \) Proof.

In order to simplify notation, in the next theorems we omit the bounding term in the second-order variables and we abbreviate \( X_1, ..., X_n \), with \( n \in \mathbb{N} \), by \( 
\hat{X} \) and \( \exists X_1, ..., X_n \) (respectively \( \forall X_1, ..., \forall X_n \) by \( \exists \hat{X} \) (respectively \( \forall \hat{X} \)).

**Theorem 3.4.** Suppose \( L K_{P S} \vdash \Gamma \to \Delta \), where \( \Gamma \) and \( \Delta \) are formed by \( \Sigma_1^{1, b} \)-formulas. Consider \( \Gamma := \exists X_1 \phi_1(\hat{x}, \hat{F}, \hat{X}), ..., \exists X_n \phi_n(\hat{x}, \hat{F}, \hat{X}) \) and \( \Delta := \exists \hat{Y} \psi_1(\hat{x}, \hat{F}, \hat{Y}), ..., \exists \hat{Y} \psi_m(\hat{x}, \hat{F}, \hat{Y}) \) where \( \phi_i \) and \( \psi_j \) are \( \Sigma_1^{1, b} \)-formulas, \( \hat{Y} = Y_1, ..., Y_k \) and \( \psi_1, ..., \psi_m \) have different components of \( \hat{X}, \hat{Y} \) respectively.

Consider \( \phi_i(\hat{x}, \hat{F}, \hat{X}) := \bigwedge_{j=1}^n \phi_j \) and denote by \( \theta(\hat{x}, \hat{F}, \hat{X}, \hat{Y}) \) the formula \( \phi(\hat{x}, \hat{F}, \hat{X}) \to \psi(\hat{x}, \hat{F}, \hat{Y}) \).

Then, there are terms \( t_i(\hat{x}) \), extended \( \Sigma_1^{1, b} \)-formulas \( M_i^f(w, \hat{x}, \hat{F}, \hat{X}) \), extended \( \Pi_1^{1, b} \)-formulas \( M_i^b(w, \hat{x}, \hat{F}, \hat{X}) \) and functions \( \phi_i(\hat{x}, \hat{F}, \hat{X}) \) in \( \text{PSPACE}^{\Sigma_1^{1, b}} \) (1 \( \leq \) \( i \leq \) \( k \)), such that:

\[ \Sigma_1^{1, b} \text{-NIA} \vdash \forall \hat{x} \forall \hat{F} \forall \hat{X} \forall \hat{Y} \forall w \leq t_i(\hat{x}) (M_i^f(w, \hat{x}, \hat{F}, \hat{X}) \leftrightarrow M_i^b(w, \hat{x}, \hat{F}, \hat{X})), (1 \leq i \leq k) \]  

In the previous result, we denote by \( \theta(\hat{x}, \hat{F}, \hat{X}), [w \leq t_i(\hat{x})] : M_i^f(w, \hat{x}, \hat{F}, \hat{X}) \), (...) the formula \( \theta(\hat{x}, \hat{F}, \hat{X}, G, ...) \) where the occurrences of \( s \in G \) are replaced by \( s \leq t_i(\hat{x}) \land M_i^f(s, \hat{x}, \hat{F}, \hat{X}) \).

**Proof.** Let \( \mathcal{P} \) be a \( L K_{P S} \)-proof of \( \Gamma \to \Delta \). By the free cut elimination theorem we can suppose that \( \mathcal{P} \) has just \( \Sigma_1^{1, b} \)-formulas. It is possible to prove, by induction on the number of lines in \( \mathcal{P} \), that for every sequent \( \Pi \to \Delta \) in \( \mathcal{P} \), there exist the terms, the formulas and the functionals described in the theorem.

We illustrate the strategy with the cases of comprehension and replacement. For a detailed proof see [15], pages 50–55.

**Comprehension.** If \( \Pi \to \Delta \) is an initial sequent of the form \( \rightarrow \exists X. Y \leq s (y \in Y \to A(y, \hat{x}, \hat{F})) \), with \( A \) a \( \Sigma_1^{1, b} \)-formula, just define \( M_i^f(w, \hat{x}, \hat{F}) \) and \( M_i^b(w, \hat{x}, \hat{F}) \) as being \( A(w, \hat{x}, \hat{F}) \). Note that a \( \Sigma_1^{1, b} \)-formula is, in particular, an extended \( \Sigma_1^{1, b} \)-formula and an extended \( \Pi_1^{1, b} \)-formula. Take \( t_i : s \land \phi_i(\hat{x}, \hat{F}) : \equiv [w \leq s : A(w, \hat{x}, \hat{F})] \).

**Replacement.** Suppose \( \Pi \to \Delta \) is \( \Gamma' \to \Delta' \), \( \exists Y. \varphi(z, Y) \), obtained from the sequent \( \Gamma' \), \( z \leq t \to \Delta' \), \( \exists Y. \varphi(z, Y) \), with \( \varphi \) a \( \Sigma_1^{1, b} \)-formula (the rule is only applied to formulas of this complexity). For simplicity of notation we assume that \( \Delta' \) has only the single formula \( \exists Y Y(\hat{x}, \hat{X}, Z) \), with \( \psi_1 \) a \( \Sigma_1^{1, b} \)-formula. By induction hypothesis for the last sequent there exist \( (M_i^f)^1(w, \hat{x}, \hat{X}) \), \( (M_i^b)^1(w, \hat{x}, \hat{X}) \), \( r_i(z, \hat{x}) \) and \( \psi_i(z, \hat{x}) \), for \( i \in \{1, 2, \} \), in the desired conditions. For the sequent \( \Gamma' \to \exists Z \psi_1(\hat{x}, \hat{X}, Z) \), \( \exists Y. \varphi(z, G) \) define \( M_i^f(w, \hat{x}, \hat{X}) \) as being alternating Turing machine, it can be decided in \( \text{PSPACE} \). In \( M_2 \) just introduce the steps to decide \( Q(x, X') \). We assume that after every oracle query, \( M_1 \) clears the oracle tape and \( M_2 \) saves the state of \( M_1 \). In the next call to the oracle, when simulating to see if \( r \in X' \), we initialize the machine with this state rather than with \( \vec{w} \). Repeating the process every time \( M_1 \) consults the oracle, we have that \( M_2 \) computes \( f \) in polynomial space. Thus, \( f \in \text{PSPACE} \).
\[ \exists z \leq t(\neg \varphi(z, \{z \leq t'_2(z) : (M'_2(z,w,z,\bar{\bar{z}},\bar{\bar{\bar{x}}})) \}\land w \leq t'_2(z) \land (M'_2(w,z,\bar{\bar{x}},\bar{\bar{\bar{x}}}))^b) ; t_1(\bar{\bar{\bar{x}}}) \] as being \[ t'_1(t(\bar{\bar{\bar{x}}}), \bar{\bar{\bar{x}}}) ; \theta \] as being \[ \emptyset \text{ if } \forall z \leq t(\varphi(z, \theta(z), \bar{\bar{\bar{x}}})) \text{ or } \theta \] as being \[ \emptyset \text{ if } \forall z \leq t(\neg \varphi(z, \theta(z), \bar{\bar{\bar{x}}}))^b ; \text{ define } M'_2(w, \bar{\bar{x}}, \bar{\bar{\bar{x}}} \text{ as being } \exists z \leq t(\neg \varphi(z, \theta(z), \bar{\bar{\bar{x}}}))((M'_2(z,s,z,\bar{\bar{x}},\bar{\bar{\bar{x}}}) \land w \leq t(\bar{\bar{x}}) \land s \leq t'_2(z) \land s \leq \theta(z)) \text{ and } M'_2(w, \bar{\bar{x}}, \bar{\bar{\bar{x}}} \text{ like the above } \Sigma \text{-versions with the expected changes}.} \]

From the previous result we get the following theorem:

**Theorem 3.5.** If \( \Sigma_{1}^{1,b} \text{-NIA} + \forall x \exists y A(x,y) \), with \( A \) a \( \Sigma_{1}^{1,b} \)-formula, then there exists a function \( f \in \text{PSPACE} \) such that, for all \( \sigma \in 2^{\omega} \), we have \( A(\bar{\sigma}, f(\bar{\sigma})) \).

Moreover, there exists \( \delta \), a formula that is equivalent (in \( \Sigma_{1}^{1,b} \text{-NIA} \)) to both an extended \( \Sigma_{1}^{1,b} \)-formula and an extended \( \Pi_{1}^{1,b} \)-formula, and a term \( t(\bar{\bar{x}}) \), such that:

1) \( f(\bar{\sigma}) = \tau(\bar{\bar{x}}, \tau) \),
2) \( \Sigma_{1}^{1,b} \text{-NIA} + \forall x \exists y A(\bar{\bar{x}}, y) \rightarrow A(\bar{\bar{x}}, y) \),
3) \( \Sigma_{1}^{1,b} \text{-NIA} + \forall x \exists y \leq t(\bar{\bar{x}}, y) \),
4) \( \Sigma_{1}^{1,b} \text{-NIA} + \forall x \exists y \theta(\bar{\bar{x}}, y) \).

**Proof.** Since \( \Sigma_{1}^{1,b} \text{-NIA} + \forall x \exists y A(\bar{\bar{x}}, y) \), with \( A \) a \( \Sigma_{1}^{1,b} \)-formula it is possible to prove that there exists \( t(\bar{\bar{x}}) \) such that \( \Sigma_{1}^{1,b} \text{-NIA} + \forall x \exists y \leq t(\bar{\bar{x}}) \text{A}(\bar{\bar{x}}, y) \). Thus \( LK_{PS} \) deduce the sequent \( \rightarrow \exists y \leq t(\bar{\bar{x}}) \text{A}(\bar{\bar{x}}, y) \). Being \( A \) a \( \Sigma_{1}^{1,b} \)-formula, suppose it is \( \exists U_{1} \ldots \exists U_{i} A \), with \( A \) a \( \Sigma_{0}^{1,b} \)-formula. So there exists a \( LK_{PS} \)-proof of \( \rightarrow \exists U_{1} \ldots \exists U_{i} \exists y \leq t(\bar{\bar{x}}) \). The result follows applying Theorem 3.4 and defining \( f(\bar{\sigma}) \) as being \( \mu y \leq t(\bar{\sigma}) \text{A}(\bar{\bar{x}}, y, \varphi(\bar{\bar{x}}), \ldots, \varphi(\bar{\bar{x}})) \), \( \theta(\bar{\bar{x}}, y) \rightarrow A(\bar{\bar{x}}, y) \), \( \{z \leq t_1(\bar{\bar{x}}) \leq M'_2(w,\bar{\bar{x}}) \}, \ldots, \{w \leq t_1(\bar{\bar{x}}) \leq \theta(\bar{\bar{x}}, y) \rightarrow A(\bar{\bar{x}}, y) \}, \) \( \{w \leq t_1(\bar{\bar{x}}) \leq M'_2(w,\bar{\bar{x}}) \}, \ldots, \{w \leq t_1(\bar{\bar{x}}) \leq \theta(\bar{\bar{x}}, y) \rightarrow A(\bar{\bar{x}}, y) \} \) and \( t(\bar{\bar{x}}) : t(\bar{\bar{x}}) \equiv t(\bar{\bar{x}}) \). By Theorem 3.4. To confirm that \( f \) is a polysize function use Proposition 3.3.

So, the provably total functions in \( \Sigma_{1}^{1,b} \text{-NIA} \), with \( \Sigma_{1}^{1,b} \)-graphs, are exactly the functions of \( \text{PSPACE} \).

4. Adding bounded collection

Having in view setting up a system of weak analysis connected with polysize computability, in this (intermediate) section we enrich the theory \( \Sigma_{1}^{1,b} \text{-NIA} \) with a bounded collection scheme. The idea is that the new scheme does not increase the computational power of the enriched theory and allows (as shown in the next section) the inclusion of

---

\( \Sigma_{1}^{1,b} \) is either \( \Sigma_{1}^{1} \) or \( \Pi_{1}^{1} \) according to our conveniences (e.g. to have the formulas in the right classes).

\( \emptyset \)By \( \mu A \) we mean the least \( \varepsilon \) satisfying \( A \). It is possible to prove that the minimization scheme for formulas that are equivalent to both an extended \( \Sigma_{1}^{1,b} \)-formula and an extended \( \Pi_{1}^{1,b} \)-formula holds in \( \Sigma_{1}^{1,b} \text{-NIA} \). See [15], page 34.
Bounded theories for polyspace computability

recursive comprehension (an essential scheme to the formalization of analysis). The inspiration comes again from the polytime setting (see [10]) where the principle of bounded collection

\[ B^{\Sigma_1^{1, b}} : \forall x \leq t \exists y A(x, y) \rightarrow \exists z \forall x \leq t \exists y \leq z A(x, y), \]

where \( A \) is a formula of \( L \) with all quantifications bounded and \( z \) is a new variable, is appended to \( \Sigma_1^{1, b} \)-NIA. See also [3, 11].

In the current context of polyspace computability, we choose a form of bounded collection slightly strengthened. See the comments in the end of the paper.

To prove, in this section, that the system \( \Sigma_1^{1, b} \)-NIA enriched with (the stronger form of) bounded collection still characterizes polyspace computability, we rely on a conservation result over \( \Sigma_1^{1, b} \)-NIA and on the cut-elimination technique.

The bounded formulas of \( L_\infty^b \), also known as the \( \Sigma_1^{1, b} \)-formulas, consist of the smallest class of formulas containing the atomic formulas and closed under Boolean connectives, first-order bounded quantifications and second-order (bounded) quantifications.

We are going to enrich \( \Sigma_1^{1, b} \)-NIA with the following principle of bounded collection.

\[ B^{\Sigma_1^{1, b}} : \forall X \exists y A(y, X') \rightarrow \exists z \forall X' \exists y \leq z A(y, X'), \]

where \( A \) is a bounded formula (possibly with parameters) and \( z \) is a new variable.

Note that the principle \( B^{\Sigma_1^{1, b}} \) can be derived in the system \( \Sigma_1^{1, b} \)-NIA + \( B^{\Sigma_1^{1, b}} \).

**Theorem 4.1.** The theory \( \Sigma_1^{1, b} \)-NIA + \( B^{\Sigma_1^{1, b}} \) is conservative over the theory \( \Sigma_1^{1, b} \)-NIA with respect to formulas of the form \( \forall \bar{X} \exists y A(\bar{x}, y) \), with \( A \) a \( \Sigma_1^{1, b} \)-formula.

**Proof.** The theory \( \Sigma_1^{1, b} \)-NIA + \( B^{\Sigma_1^{1, b}} \) can be formulated in the sequent calculus described above, \( L_{KPS} \), together with the following rule for bounded collection (\( B^{\Sigma_1^{1, b}} \)-rule):

\[
\frac{\Gamma \rightarrow \Delta, \exists y A(y, C')}{\Gamma \rightarrow \Delta, \exists z \forall X' \exists y \leq z A(y, X')}\]

where \( A \) is a \( \Sigma_1^{1, b} \)-formula (possibly with other free variables), \( C' \) is a second-order eigenvariable, and \( y \) does not occur in the term \( t \).

Suppose that \( \Sigma_1^{1, b} \)-NIA + \( B^{\Sigma_1^{1, b}} \), with \( A \) a \( \Sigma_1^{1, b} \)-formula. Thus, in the sequent calculus above, there is a proof of \( \rightarrow \forall \bar{X} \exists y A(\bar{x}, y) \) and so, a proof of \( \rightarrow \exists y A(\bar{x}, y) \).

The free cut elimination theorem ensures the existence of a proof \( \rightarrow \exists y A(\bar{x}, y) \) without free cuts. As a consequence, all formulas occurring in the sequents in \( \mathcal{P} \) are of the form \( \exists y B(y, \bar{x}, X' \bar{y}) \), with \( B \) a \( \Sigma_1^{1, b} \)-formula (the existential quantifiers “\( \exists y \)” may be absent in which case we have a bounded formula).

Let \( \Gamma \rightarrow \Delta \) be a sequent in \( \mathcal{P} \), where \( \Gamma \) is \( \exists x_1 B_1(x_1, \bar{x}, X^{p(1)}) \), \ldots, \( \exists x_n B_n(x_n, \bar{x}, X^{p(n)}) \) and \( \Delta \) is \( \exists y_1 C_1(y_1, \bar{x}, X^{p(1)}) \), \ldots, \( \exists y_k C_k(y_k, \bar{x}, X^{p(k)}) \), where \( B_1, \ldots, B_n, C_1, \ldots, C_k \) are \( \Sigma_1^{1, b} \)-formulas. To ease notation, we can assume without loss of generality and we are also only displaying the variables \( \bar{x} \) in the term \( p(\bar{x}) \).

Let us prove, by induction on the number of lines of \( \mathcal{P} \), that from bounds of the antecedents we can get (in \( \Sigma_1^{1, b} \)-NIA) bounds for the consequents in the following way:
graphs are the polyspace computable functions.

By 2), there is \( v \). To prove, as we want, that

\[
\Sigma_1^{1b}\text{-NI}A \vdash \forall u \exists v \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\Gamma \vdash \Delta^{(n)}),
\]

where \( \Gamma^{(n)} \) abbreviates \( \exists z \leq u B_1(x_1, \tilde{x}, \tilde{X}^{(n)}) \land \ldots \land \exists x_n \leq u B_n(x_n, \tilde{x}, \tilde{X}^{(n)}) \) and \( \Delta^{(n)} \) abbreviates \( \exists y \leq v C(y_1, \tilde{x}, \tilde{X}^{(n)}) \vee \ldots \lor \exists y_k \leq v C_k(y_k, \tilde{x}, \tilde{X}^{(n)}) \). Note that applying this result to the last sequent of \( \mathcal{P} \), we conclude that \( \Sigma_1^{1b}\text{-NI}A \) proves \( \forall u \exists v \forall \tilde{x} \leq u \exists y \leq v A(\tilde{x}, y) \). Thus \( \Sigma_1^{1b}\text{-NI}A \vdash \forall \exists A(\tilde{x}, y) \).

The proof above (by induction on the number of lines of \( \mathcal{P} \)) has some trivial cases. Here we illustrate two situations: when the sequent is obtained from the cut-rule and when it is obtained from the \( B^1\Sigma_0^{1b} \)-rule. For a complete proof of the result, considering all the cases, see [15], pages 58–60.

**Cut-rule.**

\[
\frac{\Gamma \to \Delta, A \quad A \to \Delta}{\Gamma \to \Delta}
\]

If \( A \) is a \( \Sigma_0^{1b} \)-formula the result is immediate taking \( v := v_1 \to v_2 \), where \( v_1 \) and \( v_2 \) the bounds that exist by induction hypothesis.

Suppose that \( A \) is of the form \( \exists z D(z, \tilde{x}, X^{(n)}) \) with \( D \) a \( \Sigma_0^{1b} \)-formula. By induction hypothesis we have:

1) \( \Sigma_1^{1b}\text{-NI}A \vdash \forall u \exists v \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\Gamma \vdash \Delta^{(n)} \lor \exists z \leq v D(z, \tilde{x}, \tilde{X}^{(n)})) \)

2) \( \Sigma_1^{1b}\text{-NI}A \vdash \forall u \exists v \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\exists z \leq u D(z, \tilde{x}, \tilde{X}^{(n)}) \land \Gamma \vdash \Delta^{(n)}). \)

Our goal is to prove that \( \Sigma_1^{1b}\text{-NI}A \vdash \forall u \exists v \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\Gamma \vdash \Delta^{(n)}). \)

Fix \( u \). By 1) there is \( v_1 \) such that \( \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\Gamma \vdash \Delta^{(n)} \lor \exists z \leq v_1 D(z, \tilde{x}, \tilde{X}^{(n)})) \). Assume that \( u \leq v_1 \) (if not just replace \( v \) by \( v_1 \) that the assertion above remains valid). By 2), there is \( v_2 \) such that \( \forall \tilde{x} \leq u \exists \forall \tilde{X}^{(n)}(\exists z \leq v_1 D(z, \tilde{x}, \tilde{X}^{(n)}) \land \Gamma \vdash \Delta^{(n)} \land \Gamma \vdash \Delta^{(n)} \land \Gamma \vdash \Delta^{(n)} \land \Gamma \vdash \Delta^{(n)} \). We get \( \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\exists z \leq v_1 D(z, \tilde{x}, \tilde{X}^{(n)}) \land \Gamma \vdash \Delta^{(n)}) \), using the fact that if \( u \leq u' \) then \( \Gamma \vdash \Delta^{(n)} \to \Gamma^{(n)} \). Let \( v := v_1 \to v_2 \). We conclude that \( \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\Gamma \vdash \Delta^{(n)} \land \Gamma \vdash \Delta^{(n)} \land \Gamma \vdash \Delta^{(n)} \land \Gamma \vdash \Delta^{(n)} \).

**B\( ^1\Sigma_0^{1b} \)-rule.**

\[
\frac{\Gamma \to \Delta, \exists y A(y, C)}{\Gamma \to \Delta, \exists y A(y, C)}
\]

By induction hypothesis we know that

\[
\Sigma_1^{1b}\text{-NI}A \vdash \forall u \exists v \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\forall y^{(n)}(\Gamma \vdash \Delta^{(n)} \lor \exists z \leq v A(y, y^{(n)}))).
\]

To prove, as we want, that

\[
\Sigma_1^{1b}\text{-NI}A \vdash \forall u \exists v \forall \tilde{x} \leq u \forall \tilde{X}^{(n)}(\Gamma \vdash \Delta^{(n)} \lor \exists z \leq v \forall y^{(n)} \exists y \leq z A(y, y^{(n)})),
\]

consider an arbitrary \( u \), take \( v \) as the element that exists by induction hypothesis, and let \( z := v \).

From the above result, noticing that \( \Sigma_1^{1b} \)-formulas are, in particular, \( \Sigma_0^{1b} \)-formulas, immediately we conclude that the provably total functions of \( \Sigma_1^{1b}\text{-NI}A + B^1\Sigma_0^{1b} \) with \( \Sigma_1^{1b} \)-graphs are the polyspace computable functions.
5. A weak analysis theory for polyspace computability

The aim of this section is to propose a weak theory of analysis, BTPSA, still characterizing polyspace computability. In [10], an homologous theory, BTFA, is presented, not for PSPACE complexity, but in the context of feasible (polytime) computability. The strategy in the polytime setting will guide us towards our goal.

The theory BTPSA, as we are going to see, includes the previous system $\Sigma_1^{1,b}$-NIA + $B^1\Sigma_\infty^{1,b}$ and is stated in a language that permits variables ranging over infinite sets. Let $L_2$ be a second-order language with equality which differs from $L^2_1$ only by the presence of second-order (unbounded) variables, denoted by $X, Y, Z, \ldots$, instead of the previous second-order “bounded” variables $X', Y', \ldots$. The class of formulas of $L_2$ is the smallest class of expressions containing the atomic formulas $t_1 \subseteq t_2, t_1 = t_2, t_1 \in X$, and closed under the Boolean operations, the first-order quantifications $\forall x, \exists x$ and the second-order quantifications $\forall X, \exists X$. Since arguments via sequent calculus are not going to be used, in BTPSA we do not treat $\forall x \leq t$ and $\exists x \leq t$ as primitive quantifiers but mere abbreviations of $\forall x(x \leq t \rightarrow \ldots)$ and $\exists x(x \leq t \land \ldots)$ respectively. The definitions of $\Sigma_0^{1,b}$ (respectively $\Sigma_1^{1,b}, \Pi_1^{1,b}, \Sigma_\infty^{1,b}$)-formulas in $L_2$ (and its extended versions) are given by obvious modifications of the homologous definitions in $L^2_1$, namely replacing $\forall X', \exists X'$ by the second-order quantifications $\forall X \leq t, \exists X \leq t$ where $X \leq t$ abbreviates $\forall z(z \in X \rightarrow z \leq t)$.

A structure for $L_2$ has domain $(M, S)$, with the first-order variables taking values in $M$ and the second-order variables varying over $S$, a given subset of $P(M)$. The standard model is $(2^{<\omega}, P(2^{<\omega}))$. Note that although we work in a second-order language, our logic is of first-order kind (first-order logic in a two-sorted language): our semantics only specifies $S$ to be a subset of $P(M)$, not necessarily all of $P(M)$.

Consider the following axiom, known as the recursive comprehension scheme:

$$\forall x(\exists yA(x, y) \leftrightarrow \forall yB(x, y)) \rightarrow \exists x\forall y(\exists x \in X \leftrightarrow \exists yA(x, y)),$$

with $A$ a $\Sigma_1^{1,b}$-formula and $B$ a $\Pi_1^{1,b}$-formula, possibly with other free variables.

In the standard model, this scheme ensures that all recursive sets exist. Although it may seem that adding this scheme to our weak (subexponential) theory will increase its computational power, we will prove that this is not the case. Note that the existence of a set is guaranteed only in the case the theory has enough resources to prove the equivalence in the antecedent of the scheme.

**Definition 5.1.** BTPSA (acronym for Base Theory for Polynomial Space Analysis) is the second-order theory, in the language $L_2$, with the following axioms: basic axioms\(^{10}\), induction on notation for extended $\Sigma_1^{1,b}$-formulas, the bounded collection scheme $B^1\Sigma_\infty^{1,b}$ and the recursive comprehension scheme mentioned above.

Intuitively $L^2_2 \subseteq L_2$, in the sense that every expression in $L^2_2$ can be formulated in $L_2$. Also, it can be seen that every model of BTPSA satisfies the axioms of $\Sigma_1^{1,b}$-NIA + $B^1\Sigma_\infty^{1,b}$.

To achieve our goal (i.e. to prove that BTPSA characterizes polyspace computability) we will do the inverse, i.e., to get models of BTPSA from models of $\Sigma_1^{1,b}$-NIA + $B^1\Sigma_\infty^{1,b}$. Notice

\(^{10}\)The 14 basic axioms of Definition 2.3.
the model-theoretic strategy instead of the proof-theoretic techniques by cut-elimination in previous conservation results.

**Lemma 5.2.** Let $\mathcal{M}$ be a model of the theory $\Sigma_{1,b}^{\text{I}}$-NIA + $\text{B}^1\Sigma_{1,b}^{\text{I}}$ with domain $(M, S_b)$. Then there is $S \subseteq \mathcal{P}(\mathcal{M})$ such that $\mathcal{M}'$, with domain $(M, S)$, is a model of BTPSA and $S_b = \{X^a : X \in S \land a \in M\}$, where $X^a$ collects the elements of $X$ with length less than or equal to $a$.

**Proof.** In order to get $\mathcal{M}'$ from $\mathcal{M}$ the idea is to “close” $\mathcal{M}$ for recursive comprehension. Let $\mathcal{S}$ be formed by the subsets $X \subseteq M$ for which there is a $\Sigma_{1,b}^{\text{I}}$-formula $\varphi$, a $\Pi_{1,b}^{\text{I}}$-formula $\psi$ and elements $\vec{a}, \vec{b}$ in $M$ and $\vec{A}, B$ in $\mathcal{S}_b$ such that $X = \{x \in M : \mathcal{M} \models \exists y (\psi(x, y, \vec{a}, \vec{A}(x,y))) = \{x \in M : \mathcal{M} \models \forall y \psi(x, y, \vec{b}, \vec{B}(x,y))\}$. The proof that $\mathcal{S}_b \subseteq \{X^a : X \in S \land a \in M\}$ follows immediately because whenever $C^a \in \mathcal{S}_b$, we have $C^a \in S$. For the other inclusion consider $C \in S$ and $c \in M$. We want to prove that $C^c \in S_b$. By definition of $S$ there are formulas $\varphi, \psi$ and elements $\vec{a}, \vec{b}, \vec{A}, \vec{B}$ (to simplify notation we omit the bounded term in the second-order parameters) - the ones defining the model-theoretic strategy instead of the proof-theoretic techniques by cut-elimination - the ones defining the model $\mathcal{M}$ being a model of $\text{B}^1\Sigma_{1,b}^{\text{I}}$. Therefore, since $\Sigma_{1,b}^{\text{I}}$-bounded comprehension is available in $\Sigma_{1,b}^{\text{I}}$-NIA (see the comments below Definition 3.1) and $\mathcal{M}$ is a model of $\Sigma_{1,b}^{\text{I}}$-NIA, we have that $\mathcal{F}^c = \{x \leq c : \exists y \leq dA(x, y, \vec{a}, \vec{A})\}$ is an element in $S_b$. Since $\mathcal{F}^c = C^c$, we have $C^c \in \mathcal{S}_b$.

It remains to prove that $\mathcal{M}'$ is a model of BTPSA. The case of the basic axioms is trivial, because the interpretation of the constants, the function and relation symbols is the same in $\mathcal{M}$ and $\mathcal{M}'$ and both models have the same first-order domain. The study of the other axioms follows more or less in a straightforward manner from the following technical fact (which can be proved by induction on the complexity of $A$): Given $A(\vec{a}, U)$ a $\Sigma_{1,b}$-formula, there is a term $q(\vec{a})$ with the following property: given $\vec{c}$ elements in $\mathcal{M}$ and $\vec{C}$ $\subseteq \mathcal{P}(\mathcal{M})$ subsets in $\mathcal{S}$ then $\mathcal{M}' \models A(\vec{c}, \vec{C}) \iff \mathcal{M} \models A(\vec{c}, \vec{C})$ whenever $q(\vec{c}, \vec{C}) \leq b$. We illustrate the application of the fact studying the recursive comprehension suppose that $\mathcal{M}' \models \forall x (\exists y A(x, y, \vec{a}, \vec{A})) \iff \mathcal{M} \models \forall y B(x, y, \vec{b}, \vec{B})$ with $A$ a $\Sigma_{1,b}$-formula and $B$ a $\Pi_{1,b}$-formula. We want to prove that $\mathcal{M}' \models \exists x \forall x (x \in X \iff \exists y A(x, y, \vec{a}, \vec{A}))$. Note that the formulas $A(x, y, \vec{a}, \vec{U})$ and $B(x, y, \vec{a}, \vec{U})$ are, in particular, $\Sigma_{1,b}$-formulas. Applying the fact above to the formula $A$, there exists a term $q(x, y, \vec{a})$ such that given $s, r, \vec{a} \in M$ and $\vec{A} \in \mathcal{S}$ we have $(\forall) \mathcal{M}' \models A(s, r, \vec{a}, \vec{A}) \iff \mathcal{M} \models A(s, r, \vec{a}, \vec{A})$ whenever $q(s, r, \vec{a}) \leq b$. Applying the same fact to $B$, there exists a term $p(x, y, \vec{a})$ such that given $s, r, \vec{a} \in M$ and $\vec{B} \in \mathcal{S}$ we have $\mathcal{M}' \models B(s, r, \vec{a}, \vec{B}) \iff \mathcal{M} \models A(s, r, \vec{a}, \vec{A})$ whenever $p(s, r, \vec{a}) \leq b$. Since $\mathcal{M}' \models \forall x \exists y A(x, y, \vec{a}, \vec{A}) \iff \mathcal{M} \models \forall x B(x, y, \vec{b}, \vec{B})$ we know that $\mathcal{M} \models \forall x (\exists y A(x, y, \vec{a}, \vec{A}(x, y))) \iff \mathcal{M} \models \forall x B(x, y, \vec{b}, \vec{B}(x, y))$. Take $X = \{x \in M : \mathcal{M} \models \exists y A(x, y, \vec{a}, \vec{A}(x, y))\} = \{x \in M : \mathcal{M} \models \forall x B(x, y, \vec{b}, \vec{B}(x, y))\}$. By the definition of $S$, $X \in \mathcal{S}$. From $(\forall)$ we know that $\mathcal{M}' \models \exists y A(x, y, \vec{a}, \vec{A}) \iff \mathcal{M} \models \exists y A(x, y, \vec{a}, \vec{A}(x, y))$, so $X = \{x \in M : \mathcal{M}' \models \exists y A(x, y, \vec{a}, \vec{A})\}$. Thus $\mathcal{M}' \models \exists x \forall x (x \in X \iff \exists y A(x, y, \vec{a}, \vec{A}))$. \hfill $\Box$

**Theorem 5.3.** $\text{BTPSA} + \forall x \exists y A(x, y)$, with $A$ a $\Sigma_{1,b}$-formula, then $\Sigma_{1,b}^{\text{I}}$-NIA + $\forall x \exists y A(x, y)$. 

R. Bianconi, G. Ferreira, E. Silva
Proof. Suppose that $\Sigma^{1,\beta}_{1}-\text{NIA} \not\vdash \forall \bar{x} \exists y A(\bar{x}, y)$, with $A$ a $\Sigma^{1,\beta}_{\infty}$-formula. Therefore, by Theorem 4.1, we also have $\Sigma^{1,\beta}_{1}-\text{NIA} + B^{1}_{\infty} \not\vdash \forall \bar{x} \exists y A(\bar{x}, y)$. By the completeness theorem, there is a model $M$ of $\Sigma^{1,\beta}_{1}-\text{NIA} + B^{1}_{\infty}$, with domain $(M, S_{\beta})$, and $\bar{a} \in M$ such that $M \models \forall y \neg A(\bar{a}, y)$. Using the previous lemma, take $S \subseteq P(M)$ such that $M^{\ast}$ with domain $(M, S)$ is a model of $\text{BTPSA}$ and $S_{\beta} = \{X^a : X \in S \land a \in M\}$. Clearly, $\Sigma^{1,\beta}_{\infty}$-formulas are absolute between $M$ and $M^{\ast}$. Therefore, we also have $M^{\ast} \models \forall y \neg A(\bar{a}, y)$. Since $M^{\ast}$ is a model of $\text{BTPSA}$, by soundness we conclude that $\text{BTPSA} \not\vdash \forall \bar{x} \exists y A(\bar{x}, y)$. 

As a consequence, the provably total functions of $\text{BTPSA}$, with $\Sigma^{1,\beta}_{1}$-graphs, are still the polyspace computable functions. For more on $\text{BTPSA}$, including its interpretability in Robinson’s theory $Q$, see [14].

Although the formalization of analysis in weak systems is out of the scope of the present paper, we just want to point to the reader that the system $\text{BTPSA}$ is strong enough to formalize the Riemann integral, up to the fundamental theorem of calculus, for functions with a modulus of uniform continuity. See [16, 13]. Such formalization does not seem to be possible in the polytime computability setting ($\text{BTFA}$), unless some classes of computational complexity collapse. See [12]. Nevertheless, a non trivial amount of analysis ([6]) is already available in $\text{BTFA}$.

The reason we opted to equip $\text{BTPSA}$ with a bounded collection scheme stronger than the one available in $\text{BTFA}$ is twofold. First, as we saw, (although stronger) it is still conservative over $\Sigma^{1,\beta}_{1}-\text{NIA}$. Secondly, it is known that to prove that the following strict $\Pi^{1}_{1}$-reflection principle (a strong form of weak Konig’s lemma):

$$\forall X \exists x A(X, x) \rightarrow \exists z \forall X \exists x \leq z A(X, x),$$

where $A$ is a $\Sigma^{1,\beta}_{1}$-formula, is first-order conservative over $\text{BTPSA}$, we need the stronger bounded collection scheme $B^{1}_{\infty} \Sigma^{1,\beta}_{1}$. See [5, 16]. And in $\text{BTPSA}$ enriched with the previous reflection principle, the Riemann integral is available for continuous functions (no need for the modulus of uniform continuity assumption).

Acknowledgments

The second author is very grateful to Fernando Ferreira for the help and guidance in achieving the results of this paper. Many thanks also to the anonymous referee for the detailed revision and comments that so much improved the final version of this paper.

References


Bounded theories for polysize computability


Received

Ricardo Bianconi, Departamento de Matemática, Instituto de Matemática e Estatística, Universidade de São Paulo, Rua do Matão, 1010, CEP 05508-090, São Paulo, SP, Brazil
E-mail: bianconi@ime.usp.br

Gilda Ferreira, Departamento de Matemática, Universidade Lusófona de Humanidades e Tecnologias, Av. do Campo Grande, 376, 1749-024 Lisboa, Portugal and CMAF-Universidade de Lisboa
E-mail: gildafer@cii.fc.ul.pt

Emmanuel Silva, Tribunal Regional Federal da 5ª Região, Cais do Apolo, s/n - Edifício Ministro Djaci Falcão, Bairro do Recife - Recife - PE, CEP 50030-908, Brazil
E-mail: eflsilva@trf3.jus.br