Polynomial time operations in explicit mathematics

Thomas Strahm*

Abstract

In this paper we study (self-)applicative theories of operations and binary words in the context of polynomial time computability. We propose a first order theory PTO which allows full self-application and whose provably total functions on $\mathbb{W} = \{0,1\}^*$ are exactly the polynomial time computable functions. Our treatment of PTO is proof-theoretic and very much in the spirit of reductive proof theory.

1 Introduction

Theories with self-application provide an elementary framework for many activities in (the foundations of) mathematics and computer science. They were first introduced by Feferman [11, 12] as a basis for his systems of explicit mathematics, e.g., the theory T_0; these theories are broadly discussed in the literature from a proof-theoretic and model-theoretic point of view, cf. e.g. the textbooks Beeson [2] and Troelstra and Van Dalen [25] for a survey.

It is the aim of the present work to propose a *first order* theory PTO of operations and binary words, which allows full self-application and whose provably total functions on $\mathbb{W} = \{0,1\}^*$ are exactly the polynomial time computable functions. In spite of its proof-theoretic weakness, PTO has an enormous expressive power due to the presence of full (partial) combinatory logic, i.e. there are terms for every partial recursive function.

When trying to set up a theory with self-application of polynomial strength, one might first try to mimic first order systems of bounded arithmetic – say Buss' S^1_2 – in the applicative setting in a direct way. However, it is shown in Strahm [24] that this naive approach does not work, and one immediately ends up with systems of the same strength as primitive recursive arithmetic PRA; this is due to the presence of unbounded recursion principles in the applicative language. Hence, a direct translation of induction principles from bounded arithmetic is not successful, and a theory had to be found which is better tailored for the applicative framework.

*Research supported by the Swiss National Science Foundation.

The formulation of the proposed theory PTO is very much akin to well-known theories of operations and numbers, namely PTO can be viewed as the polynomial time analogue of the theory $\mathsf{BON} + (\mathsf{Set\text{-}IND}_N)$ of Feferman and Jäger [15]. The choice of a unary predicate W for binary words instead of a predicate N for natural numbers is not mandatory, but more natural in the context of polynomial time computability. Crucial in the formulation of PTO is the principle of so-called *set induction*, which is very natural and – most important – in the spirit of applicative theories.

The proof of the fact that PTO captures exactly polynomial time is established along the lines of reductive proof theory. More precisely, we show that PTO contains Ferreira's system of polynomial time computable arithmetic PTCA (cf. [17, 18]) via a natural embedding. Furthermore, PTO is reducible to the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$, where PTCA^+ denotes the extension of PTCA by NP induction and $(\Sigma\text{-}\mathsf{Ref})$ is the reflection principle for Σ formulas. Σ reflection $(\Sigma\text{-}\mathsf{Ref})$ is equivalent to the collection principle for bounded formulas, $(\Sigma^b_\infty\text{-}\mathsf{CP})$. $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ is known to be a Π_2 conservative extension of PTCA^+ by the work of Buss [5], Cantini [6], or Ferreira [16]. Moreover, PTCA^+ is Π_2 conservative over PTCA by Buchholz and Sieg [3], Cantini [6], and Ferreira [18]. Summing up, the provably total functions of $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ are exactly the polytime functions.

Finally, let us mention that our approach can easily be extended in order to provide applicative theories which capture the nth level of the Grzegorczyk hierarchy.

The plan of the paper is as follows. In Section 2 we introduce the formal framework for partial applicative theories, and we give an exact formulation of the theory PTO. Section 3 is centered around the theory of polynomial time computable arithmetic PTCA^+ plus the Σ reflection principle, and some known proof-theoretic results are addressed. The exact proof-theoretic strength of PTO is established in Section 4: we give an embedding of PTCA into PTO and show how PTO can be reduced to $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. Section 5 deals with various conservative extensions of PTO, and in Section 6 we briefly address suitable applicative theories which capture the Grzegorczyk classes. Section 7 contains a conclusion and an open problem concerning the totality of the application operation. Finally, in the appendix of this paper we include a proof of Theorem 10.

2 The theory PTO

In this section we introduce the theory PTO of polynomial time operations on binary words, and we address some of its basic properties.

The language $\mathcal{L}_{\mathsf{PTO}}$ of PTO is a first order language of partial terms with *individual variables* $a, b, c, x, y, z, u, v, w, f, g, h, \dots$ (possibly with subscripts). In addition, $\mathcal{L}_{\mathsf{PTO}}$ includes *individual constants* k, s (combinators), $\mathsf{p}, \mathsf{p}_0, \mathsf{p}_1$ (pairing and unpairing), $\boldsymbol{\epsilon}, \mathbf{0}, \mathbf{1}$ (empty word, zero, one), $\boldsymbol{*}, \times, \mathsf{p}_W$ (word concatenation and multiplication, predecessor), $\mathsf{c}_\subseteq$ (initial subword relation), d_W (definition by cases on binary words), r_W (bounded primitive recursion). $\mathcal{L}_{\mathsf{PTO}}$ has a binary function symbol $\cdot$ for (partial) term application, unary relation symbols $\downarrow$ (defined) and W (binary words) as well as a binary relation symbol $=$ (equality).

The *individual terms* $(r, s, t, r_1, s_1, t_1, \ldots)$ of $\mathcal{L}_{\mathsf{PTO}}$ are inductively defined as follows:

1. The individual variables and individual constants are individual terms.

2. If s and t are individual terms, then so also is $(s \cdot t)$.

In the following we write (st) or just st instead of $(s \cdot t)$, and we adopt the convention of association to the left, i.e., $s_1 s_2 \ldots s_n$ stands for $(\ldots(s_1 s_2)\ldots s_n)$. We also write (t_1, t_2) for $\mathsf{p} t_1 t_2$ and $(t_1, t_2, \ldots, t_n)$ for $(t_1, (t_2, \ldots, t_n))$. Finally, we often use infix notation for $*$ and $\times$, i.e. $s{*}t$ abbreviates $*st$ and $s{\times}t$ stands for $\times st$.

The *formulas* $(\varphi, \psi, \chi, \varphi_1, \psi_1, \chi_1, \ldots)$ of $\mathcal{L}_{\mathsf{PTO}}$ are inductively defined as follows:

1. Each atomic formula $W(t)$, $t{\downarrow}$ and $(s = t)$ is a formula.

2. If φ and ψ are formulas, then so also are $\neg\varphi$, $(\varphi \vee \psi)$, $(\varphi \wedge \psi)$ and $(\varphi \rightarrow \psi)$.

3. If φ is a formula, then so also are $(\exists x)\varphi$ and $(\forall x)\varphi$.

Our applicative theories are based on *partial* term application. Hence, it is not guaranteed that terms have a value, and $t{\downarrow}$ is read as 't is defined' or 't has a value'. The *partial equality relation* $\simeq$ is introduced by

$$s \simeq t \;:=\; (s{\downarrow} \vee t{\downarrow}) \rightarrow (s = t).$$

We use the following abbreviations concerning the predicate W $(\vec{s} = s_1, \ldots, s_n)$:

$$\begin{array}{rcl}
\vec{s} \in W & := & W(s_1) \wedge \cdots \wedge W(s_n), \\
(\exists x \in W)\varphi & := & (\exists x)(x \in W \wedge \varphi), \\
(\forall x \in W)\varphi & := & (\forall x)(x \in W \rightarrow \varphi), \\
(t : W \rightarrow W) & := & (\forall x \in W)(tx \in W), \\
(t : W^{m+1} \rightarrow W) & := & (\forall x \in W)(tx : W^m \rightarrow W).
\end{array}$$

In addition, let us write $s \subseteq t$ instead of $\mathsf{c}_{\subseteq} st = \mathbf{0}$, and $s \leq t$ for $\mathbf{1}{\times}s \subseteq \mathbf{1}{\times}t$. Finally, $(s = t \,|\, r)$ is an abbreviation for

$$(r \leq t \wedge s \subseteq t \wedge \mathbf{1}{\times}s = \mathbf{1}{\times}r) \vee (t \leq r \wedge s = t).$$

Sets of binary words are naturally understood in our context via their total characteristic functions. Accordingly, we define $P(W)$ by

$$f \in P(W) \;:=\; (\forall x \in W)(fx = \mathbf{0} \vee fx = \mathbf{1}).$$

Before we turn to the exact axiomatization of PTO, let us give an informal interpretation of its syntax. The individual variables are conceived of as ranging over a universe V of computationally amenable objects, which can freely be applied to each other. Self-application is meaningful, but not necessarily total. V is assumed to be combinatory complete, due to the presence of the well-known combinators k and s, and V is closed under pairing. There is a collection of objects $W \subset V$, consisting of finite sequences of $\mathbf{0}$'s and $\mathbf{1}$'s; W is generated from $\boldsymbol{\epsilon}$, $\mathbf{0}$ and $\mathbf{1}$ by the operation $*$ of word concatenation. Furthermore, we have an operation $\times$ of word multiplication, where $w_1 \times w_2$ denotes the word w_1 concatenated with itself length of w_2 times. p_W is supposed to be a predecessor or destructor operation on W, and $\mathsf{c}_{\subseteq}$ denotes the characteristic function of the initial subword relation. d_W acts as a definition by cases operator on W. The relation $w_1 \leq w_2$ means that the length of w_1 is less than or equal to the length of w_2; accordingly, $w_1 \,|\, w_2$ denotes the truncation of w_1 to the length of w_2. This gives meaning to the *bounded* recursor r_W on W, which provides an operation $\mathsf{r}_W fgb$ for primitive recursion from f and g with length bound b.

The underlying logic of PTO is the classical logic of partial terms due to Beeson [2]; it corresponds to E^+ logic with strictness and equality of Troelstra and van Dalen [26]. The non-logical axioms of PTO are divided into the following nine groups.

I. Partial combinatory algebra.

(1) $\mathsf{k}xy = x$,

(2) $\mathsf{s}xy\downarrow \,\wedge\, \mathsf{s}xyz \simeq xz(yz)$.

II. Pairing and projection.

(3) $\mathsf{p}_0(x,y) = x \,\wedge\, \mathsf{p}_1(x,y) = y$.

III. Binary words.

(4) $\boldsymbol{\epsilon} \in W \,\wedge\, \mathbf{0} \in W \,\wedge\, \mathbf{1} \in W$,

(5) $(* : W^2 \to W)$,

(6) $x \in W \to x*\boldsymbol{\epsilon} = x$,

(7) $x \in W \wedge y \in W \,\to\, x*(y*\mathbf{0}) = (x*y)*\mathbf{0} \,\wedge\, x*(y*\mathbf{1}) = (x*y)*\mathbf{1}$,

(8) $x \in W \wedge y \in W \,\to\, x*\mathbf{0} \neq y*\mathbf{1} \,\wedge\, x*\mathbf{0} \neq \boldsymbol{\epsilon} \,\wedge\, x*\mathbf{1} \neq \boldsymbol{\epsilon}$,

(9) $x \in W \wedge y \in W \wedge x*\mathbf{0} = y*\mathbf{0} \,\to\, x = y$,

(10) $x \in W \wedge y \in W \wedge x*\mathbf{1} = y*\mathbf{1} \,\to\, x = y$.

IV. Word multiplication.

(11) $\times : W^2 \to W$,

(12) $x \in W \to x\times\boldsymbol{\epsilon} = \boldsymbol{\epsilon}$,

(13) $x \in W \wedge y \in W \,\to\, x\times(y*\mathbf{0}) = (x\times y)*x \,\wedge\, x\times(y*\mathbf{1}) = (x\times y)*x$.

V. Predecessor on W.

(14) $\mathsf{p}_W : W \to W$,

(15) $\mathsf{p}_W \boldsymbol{\epsilon} = \boldsymbol{\epsilon}$,

(16) $x \in W \ \to\ \mathsf{p}_W(x{*}\mathbf{0}) = x \ \wedge\ \mathsf{p}_W(x{*}\mathbf{1}) = x$,

(17) $x \in W \wedge x \neq \boldsymbol{\epsilon} \ \to\ (\mathsf{p}_W x){*}\mathbf{0} = x \ \vee\ (\mathsf{p}_W x){*}\mathbf{1} = x$.

VI. Initial subword relation.

(18) $x \in W \wedge y \in W \ \to\ \mathsf{c}_{\subseteq} xy = \mathbf{0} \ \vee\ \mathsf{c}_{\subseteq} xy = \mathbf{1}$,

(19) $x \in W \ \to\ (x \subseteq \boldsymbol{\epsilon} \leftrightarrow x = \boldsymbol{\epsilon})$,

(20) $x \in W \wedge y \in W \wedge y \neq \boldsymbol{\epsilon} \ \to\ (x \subseteq y \leftrightarrow x \subseteq \mathsf{p}_W y \vee x = y)$.

VII. Definition by cases on W.

(21) $a \in W \wedge b \in W \wedge a = b \ \to\ \mathsf{d}_W xyab = x$,

(22) $a \in W \wedge b \in W \wedge a \neq b \ \to\ \mathsf{d}_W xyab = y$.

VIII. Bounded primitive recursion on W.

(23) $(f : W \to W) \ \wedge\ (g : W^3 \to W) \ \wedge\ (b : W^2 \to W) \ \to\ (\mathsf{r}_W fgb : W^2 \to W)$,

(24) $(f : W \to W) \ \wedge\ (g : W^3 \to W) \ \wedge\ (b : W^2 \to W) \wedge$
$x \in W \ \wedge\ y \in W \ \wedge\ y \neq \boldsymbol{\epsilon} \ \wedge\ h = \mathsf{r}_W fgb \ \to$
$hx\boldsymbol{\epsilon} = fx \ \wedge\ hxy = gxy(hx(\mathsf{p}_W y)) \,|\, bxy$.

IX. Set induction on W ($\mathsf{S\text{-}I_W}$)

(25) $f \in P(W) \ \wedge\ f\boldsymbol{\epsilon} = \mathbf{0} \ \wedge\ (\forall x \in W)(f(\mathsf{p}_W x) = \mathbf{0} \ \to\ fx = \mathbf{0}) \ \to$
$(\forall x \in W)(fx = \mathbf{0})$.

Observe that in the formulation of bounded primitive recursion r_W on W, we do *not* require b to be a polynomial, but only a total operation on W. This formulation is more natural, and we will see in Section 4.2 that it does not raise the proof-theoretic strength of PTO.

The principle of set induction is crucial for the proof-theoretic strength of PTO. As we will see in Section 4, the premise $f \in P(W)$ allows one to treat set induction in a certain theory of arithmetic, which has polynomial strength only. Set induction has previously played an important role in systems of explicit mathematics with the so-called non-constructive minimum operator, cf. [15, 14, 20, 22].

As usual the axioms of a partial combinatory algebra allow one to define λ *abstraction* and to prove a *recursion theorem* (cf. e.g. [2, 11]). Hence, there is an $\mathcal{L}_{\mathsf{PTO}}$ term t_f for each partial recursive function f, however, PTO does generally not prove the totality of f. In particular, PTO includes a term t_{exp} for exponentiation.

Proposition 1 *For each $\mathcal{L}_{\mathsf{PTO}}$ term t there exists an $\mathcal{L}_{\mathsf{PTO}}$ term $(\lambda x.t)$ whose free variables are those of t, excluding x, so that*

$$\mathsf{PTO} \vdash (\lambda x.t)\downarrow \wedge (\lambda x.t)x \simeq t.$$

Proposition 2 *There exists an $\mathcal{L}_{\mathsf{PTO}}$ term* rec *so that*

$$\mathsf{PTO} \vdash \mathsf{rec}f\downarrow \wedge (\forall x)(\mathsf{rec}fx \simeq f(\mathsf{rec}f)x).$$

In the following let us briefly sketch the standard recursion-theoretic model *PRO* (partial recursive operations) of PTO. The universe of *PRO* consists of the set of finite 0-1 sequences $\mathbb{W} = \{0,1\}^*$, and W is interpreted by $\mathbb{W}$. Application $\cdot$ is interpreted as partial recursive function application, i.e. $x \cdot y$ means $\{x\}(y)$ in *PRO*, where $\{x\}$ is a standard enumeration of the partial recursive functions over $\mathbb{W}$. It is easy to find interpretations of the constants of $\mathcal{L}_{\mathsf{PTO}}$ so that the axioms of PTO are true in *PRO*. Observe that the elements of $P(W)$ are exactly the recursive sets on $\mathbb{W}$ in *PRO*.

There are many more interesting models of the combinatory axioms, which can easily be extended to models of PTO. These include further recursion-theoretic models, term models, generated models and set-theoretic models. For detailed descriptions and results the reader is referred to Beeson [2], Feferman [12] and Troelstra and van Dalen [25].

Let us finish this section by making some comments concerning polynomial time functionals. Cook and Urquhart [10] introduced a class *BFF* of *basic feasible functionals* in all finite types in order to provide functional interpretations of feasibly constructive arithmetic. The type 1 functions of *BFF* coincide with the polynomial time computable functions. It is straightforward from the axioms of PTO and Proposition 1 that there exists an $\mathcal{L}_{\mathsf{PTO}}$ term t_F for each functional F in *BFF* so that the defining equations and the well-typedness of F are derivable in PTO. Further work on *BFF* and feasible functionals in general can be found in Cook and Kapron [9] and Seth [23].

3 The theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$

In the following let us briefly sketch the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$, which we will use in the next section in order to interpret PTO.

The theory PTCA of polynomial time computable arithmetic over binary strings was introduced by Ferreira [17, 18]. PTCA can be viewed as a polynomial time analogue of Skolem's system of primitive recursive arithmetic PRA. The theory PTCA is formulated in the first order language $L_{\mathcal{P}}$, which is based on the elementary language L. The latter contains individual variables $a, b, c, x, y, z, u, v, w, f, g, h, \ldots$ (possibly with subscripts), constants ϵ, 0, 1, the binary function symbols $*$ and $\times$[1] as well as

[1]We again use infix notation for $*$ and $\times$ and often write ts instead of $t * s$

the binary relation symbols $=$ and $\subseteq$; the meaning of these symbols is identical to the one of the corresponding operations in $\mathcal{L}_{\mathsf{PTO}}$. Now $L_{\mathcal{P}}$ is obtained from L by adding a function symbol for each description of a polynomial time computable function, where the terms of L act as bounding terms, similar to Cobham's characterization of the polytime functions (cf. [8]). Terms $(r, s, t, \ldots)$ and formulas $(\varphi, \psi, \chi, \ldots)$ of $L_{\mathcal{P}}$ (both possibly with subscripts) are defined as usual. For the details the reader is referred to [17, 18].

There are two sorts of *bounded quantifiers* which are relevant in the sequel. The *sharply bounded quantifiers* have the form $(\exists x)(x \subseteq t \wedge \ldots)$ or $(\forall x)(x \subseteq t \rightarrow \ldots)$, and in the following we just write $(\exists x \subseteq t)(\ldots)$ and $(\forall x \subseteq t)(\ldots)$. Furthermore, we have *(generally) bounded quantifiers* $(\exists x)(x \leq t \wedge \ldots)$ and $(\forall x)(x \leq t \rightarrow \ldots)$, where $x \leq t$ reads as $1 \times x \subseteq 1 \times t$ as in the previous section. Again we use the usual shorthands as above. If φ is an arbitrary $L_{\mathcal{P}}$ formula, then we write φ^t for the formula which is obtained from φ by replacing each unbounded quantifier $(\mathcal{Q}x)$ by the corresponding bounded quantifier $(\mathcal{Q}x \leq t)$. The following definition contains important classes of $L_{\mathcal{P}}$ formulas.

Definition 3 Let us define the following eight classes of $L_{\mathcal{P}}$ formulas.

1. QF denotes the set of all quantifier free $L_{\mathcal{P}}$ formulas.
2. A formula is called Δ_0^b if all its quantifiers are *sharply* bounded.
3. A formula is in the class Σ_1^b if it has the form $(\exists x \leq t)\varphi$ for φ a formula in QF.
4. A formula is called *extended* Σ_1^b or $e\Sigma_1^b$ if (i) all its positive existential and negative universal quantifiers are bounded, and (ii) all its positive universal and negative existential quantifiers are *sharply* bounded.
5. An $L_{\mathcal{P}}$ formula is called Σ_∞^b or *bounded* if all its quantifiers are bounded.
6. A Σ_1 formula has the form $(\exists x)\varphi$ for φ in QF; a Π_2 formula is of the shape $(\forall x)(\exists y)\varphi$ for φ in QF.
7. A formula is in the class Σ if all its positive universal and negative existential quantifiers are bounded.

The Δ_0^b formulas are the polynomial time decidable matrices of [17, 18]. Furthermore, the Σ_1^b formulas define exactly the *NP* predicates and the Σ_∞^b formulas the predicates in the Meyer-Stockmeyer polynomial time hierarchy.

The theory of polynomial time computable arithmetic PTCA is a first order theory based on classical logic with equality, and comprising defining axioms for the base language L as well as defining equations for each description of a polytime function in $L_{\mathcal{P}}$. In addition, PTCA includes the notation induction scheme

$$\varphi(\epsilon) \wedge (\forall x)(\varphi(x) \rightarrow \varphi(x0) \wedge \varphi(x1)) \rightarrow (\forall x)\varphi(x)$$

for each $L_{\mathcal{P}}$ formula $\varphi(x)$ in QF. It is well-known that PTCA proves induction for Δ_0^b formulas. For details we refer to [17, 18]. Furthermore, it is straightforward to establish that the provably total functions of PTCA are exactly the polytime functions (cf. [3, 18]).

Let PTCA^+ denote the extension of PTCA, where notation induction is allowed for *NP* predicates, i.e. formulas in Σ_1^b. The system PTCA^+ is closely related to Buss' system S_2^1 (cf. [4]). Induction is provable in PTCA^+ for *extended* Σ_1^b formulas (cf. [17, 18]). In analogy to Parson's result we obtain that PTCA^+ is a conservative extension of PTCA with respect to Π_2 statements. Proofs can be found in [3, 6, 18].

Proposition 4 *Suppose* $\mathsf{PTCA}^+ \vdash (\forall x)(\exists y)\varphi(x,y)$*, where* φ *is a* QF *formula. Then we have* $\mathsf{PTCA} \vdash (\forall x)(\exists y)\varphi(x,y)$*.*

Corollary 5 *Suppose* $\mathsf{PTCA}^+ \vdash (\forall x)(\exists y)\varphi(x,y)$*, where* φ *is a* QF *formula. Then there exists an* $L_{\mathcal{P}}$ *term* $t(x)$ *so that* $\mathsf{PTCA} \vdash (\forall x)\varphi(x,t(x))$*.*

In order to interpret our theory of polynomial time operations on binary words PTO, we will need the crucial principle of Σ *reflection* (Σ-Ref), which has the form

$$(\Sigma\text{-}\mathsf{Ref}) \qquad \varphi \;\rightarrow\; (\exists a)\varphi^a,$$

where φ is a formula in Σ. It is not difficult to see that (Σ-Ref) is equivalent to the *collection principle for bounded formulas* (Σ_∞^b-CP), which reads as

$$(\Sigma_\infty^b\text{-}\mathsf{CP}) \qquad (\forall x \le t)(\exists y)\varphi \;\rightarrow\; (\exists a)(\forall x \le t)(\exists y \le a)\varphi,$$

where φ is a Σ_∞^b formula. It is known that adding Σ reflection (or equivalently bounded collection) to a suitable bounded theory yields a Π_2 conservative extension. This was first proved by Buss [5]. Another elementary model-theoretic proof is due to Ferreira [16]. Finally, a very perspicuous proof-theoretic proof making use of partial cut elimination and an *asymmetric interpretation* has recently been established by Cantini [6].

Proposition 6 *Suppose* $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref}) \vdash (\forall x)(\exists y)\varphi(x,y)$*, where* φ *is a* Σ_∞^b *formula. Then we have* $\mathsf{PTCA}^+ \vdash (\forall x)(\exists y)\varphi(x,y)$*.*

As consequence we get by Corollary 5 the desired conservation result.

Corollary 7 *Suppose* $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref}) \vdash (\forall x)(\exists y)\varphi(x,y)$*, where* φ *is a* QF *formula. Then there exists an* $L_{\mathcal{P}}$ *term* $t(x)$ *so that* $\mathsf{PTCA} \vdash (\forall x)\varphi(x,t(x))$*.*

Let us mention that Σ reflection (Σ-Ref) follows from Weak König's Lemma for trees defined by bounded formulas, (Σ_∞^b-WKL). In fact, the first order strength of (Σ_∞^b-WKL) is exactly (Σ-Ref) (over the base theory PTCA^+), cf. Ferreira [19]. Furthermore, (Σ_∞^b-WKL) is a consequence of strict Π_1^1 reflection, which by Cantini [6] again yields a Π_2 conservative extension of PTCA.

In the following we often write $|s|$ (the length of s) instead of $1 \times s$, $s \subset t$ instead of $s \subseteq t \wedge s \neq t$, and $s < t$ instead of $1 \times s \subset 1 \times t$. The abbreviation $s = t\,|\,r$ is understood in the same way as in the previous section. In addition, p denotes the obvious predecessor function on binary words and $c_{\subseteq}$ is the binary characteristic function of the initial subword relation. Finally, we use the trivial representation of the natural numbers as tally words, which is given by $\overline{0} = \epsilon$ and $\overline{n+1} = \overline{n}1$. We will write n instead of $\overline{n}$ whenever it is clear from the context that we mean n as a tally word and not as a natural number.

We finish this section by adopting some conventions concerning polynomial time sequence coding within PTCA. For the details the reader is again referred to Ferreira [17, 18]. Let $\langle \ldots \rangle$ denote a polytime function for forming n-sequences $\langle t_0, \ldots, t_{n-1} \rangle$ of binary words, and let $lh(t)$ denote the length of the sequence coded by t, i.e. if $t = \langle t_0, \ldots, t_{n-1} \rangle$, then $lh(t) = \overline{n}$. We write $Seq_n(t)$ for $Seq(t) \wedge lh(t) = \overline{n}$. There is a polytime projection function so that $(t)_m$ denotes the mth component of the sequence coded by t if $m \subset lh(t)$; we write $last(t)$ for $(t)_{p(lh(t))}$ and $(t)_{m,n}$ instead of $((t)_m)_n$. Furthermore, let $\frown$ denote the polytime sequence concatenation function. For example, if t is the sequence $\langle t_0, t_1, t_2, t_3 \rangle$, then $lh(t) = 1111$, $(t)_\epsilon = t_0$, $(t)_1 = t_1$, $(t)_{11} = t_2$, $(t)_{111} = t_3$, $last(t) = t_3$ and $t = \langle t_0, t_1 \rangle \frown \langle t_2, t_3 \rangle$. Finally, let $SqBd(a, b)$ denotes a suitable $L_{\mathcal{P}}$ term so that PTCA proves

$$Seq(v) \wedge lh(v) \leq |b|1 \wedge (\forall w \subset lh(v))((v)_w \leq a) \;\rightarrow\; v \leq SqBd(a, b).$$

$SqBd$ is easily constructed from the terms in L. This ends our discussion of the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. In the next sections we establish the exact proof-theoretic strength of PTO and its extensions.

4 The proof-theoretic strength of PTO

In the following we address the main result of this paper, which says that the provably total functions of PTO are exactly the polytime functions. We sketch proof-theoretic lower and upper bounds, and we propose a generalization of set induction which does not go beyond polynomial strength.

4.1 Lower bounds

There is a natural embedding of the language $L_{\mathcal{P}}$ into the language $\mathcal{L}_{\mathsf{PTO}}$. Using the bounded recursion operator r_W, each (description of) a polytime function can be represented in PTO by an $\mathcal{L}_{\mathsf{PTO}}$ term. Furthermore, the recursion equations and the totality of the corresponding function are derivable in PTO. Hence, we have an $\mathcal{L}_{\mathsf{PTO}}$ formula $\varphi^W(\vec{x})$ for each $L_{\mathcal{P}}$ formula φ, where the individual variables of $L_{\mathcal{P}}$ are supposed to range over W, i.e.

$$((\exists y)\varphi(\vec{x}, y))^W \;=\; (\exists y \in W)\varphi^W(\vec{x}, y),$$

and similarly for universal quantifiers. Moreover, each quantifier free formula of $L_{\mathcal{P}}$ can be represented in $\mathcal{L}_{\mathsf{PTO}}$ by a set in the sense of $P(W)$.

Lemma 8 *For every quantifier free formula $\varphi(\vec{x})$ of $L_{\mathcal{P}}$ with at most $\vec{x}$ free there exists an individual term t_φ of $\mathcal{L}_{\mathsf{PTO}}$, so that*

1. $\mathsf{PTO} \vdash (\forall \vec{x} \in W)(t_\varphi \vec{x} = \mathbf{0} \vee t_\varphi \vec{x} = \mathbf{1})$,
2. $\mathsf{PTO} \vdash (\forall \vec{x} \in W)(\varphi^W(\vec{x}) \leftrightarrow t_\varphi \vec{x} = \mathbf{0})$.

It is an immediate consequence of this lemma that notation induction for quantifier free formulas carries over to set induction in $\mathcal{L}_{\mathsf{PTO}}$. Hence, we have the following embedding of PTCA into PTO.

Theorem 9 *We have for every $L_{\mathcal{P}}$ formula $\varphi(\vec{x})$ with at most $\vec{x}$ free:*

$$\mathsf{PTCA} \vdash \varphi(\vec{x}) \quad \Longrightarrow \quad \mathsf{PTO} \vdash \vec{x} \in W \rightarrow \varphi^W(\vec{x}).$$

This finishes our discussion of the lower bound for PTO.

4.2 Upper bounds

In the following we show that PTO can be embedded into $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$, which is known to be a Π_2 conservative extension of PTCA by the results of Section 3. As a consequence, we obtain that the provably total functions of PTO are computable in polynomial time.

The main step in establishing an embedding of PTO into $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ is to find an $L_{\mathcal{P}}$ formula $App(x, y, z)$ which interprets $xy \simeq z$. Together with an interpretation of the constants of $\mathcal{L}_{\mathsf{PTO}}$ this will yield a translation of $\mathcal{L}_{\mathsf{PTO}}$ into $L_{\mathcal{P}}$ in a standard way. In the definition of App we will make use of a construction similar to Feferman [12], p. 200, Feferman and Jäger [14], p. 258 or Beeson [2], p. 144. In particular, App will be represented as a fixed point of a Σ_1 positive inductive definition. The details of this construction are very relevant due to the weakness of $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$.

In order to describe a suitable inductive operator form below, it will be convenient to work with an extension $L_{\mathcal{P}}(Q)$ of $L_{\mathcal{P}}$ by a ternary relation symbol Q which does not belong to $L_{\mathcal{P}}$. If $\varphi(Q)$ is an $L_{\mathcal{P}}(Q)$ formula and $\psi(x, y, z)$ an $L_{\mathcal{P}}$ formula, then $\varphi(\psi)$ denotes the result of substituting $\psi(r, s, t)$ for every occurrence of $Q(r, s, t)$ in the formula $\varphi(Q)$.

In the following let us first turn to the interpretation of the recursion operator r_W. Toward this end, assume that $A(f, x, y)$ is a fixed $L_{\mathcal{P}}(Q)$ formula with at most f, x, y free. Then we define for each natural number n greater than 0 an $L_{\mathcal{P}}(Q)$ formula $A_n(f, x_1, \ldots, x_n, y)$ by recursion on n as follows:

$$\begin{array}{rcl} A_1(f, x_1, y) & := & A(f, x_1, y), \\ A_{n+1}(f, x_1, \ldots, x_{n+1}, y) & := & (\exists z)(A_n(f, x_1, \ldots, x_n, z) \wedge A(z, x_{n+1}, y)). \end{array}$$

If $A(f,x,y)$ is assumed to interpret $fx \simeq y$, then $A_n(f,x_1,\dots,x_n,y)$ interprets $fx_1,\dots x_n \simeq y$. We will drop the subscript n whenever it is clear from the context.

Now we are ready to define the $L_{\mathcal{P}}(Q)$ formula $Rec_A(f,g,b,x,y,z)$. It describes the graph of the function which is defined from f and g by bounded primitive recursion with length bound b in the sense of A. The exact formulation of Rec_A is as follows:

$$
\begin{aligned}
& Rec_A(f,g,b,x,y,z) \ := \\
& \quad (\exists v)[Seq(v) \ \wedge \ lh(v) = |y|1 \ \wedge \ A(f,x,(v)_\epsilon) \ \wedge \\
& \qquad (\forall w \subseteq y)(w \neq \epsilon \rightarrow \\
& \qquad\quad (\exists u_1,u_2)[A_3(g,x,w,(v)_{|p(w)|},u_1) \wedge A_2(b,x,w,u_2) \wedge (v)_{|w|} = u_1|u_2]) \\
& \qquad \wedge \ z = (v)_{|y|}].
\end{aligned}
$$

In a next step we define a Q-positive $L_{\mathcal{P}}(Q)$ formula $\mathcal{A}(Q,x,y,z)$, a so-called inductive operator form; a fixed point of $\mathcal{A}$ will later serve as an interpretation of the application operation. Let us choose pairwise different binary words $\hat{\mathsf{k}}$, $\hat{\mathsf{s}}$, $\hat{\mathsf{p}}$, $\hat{\mathsf{p}}_0$, $\hat{\mathsf{p}}_1$, $\hat{*}$, $\hat{\times}$, $\hat{\mathsf{p}}_W$, $\hat{\mathsf{c}}_\subseteq$, $\hat{\mathsf{d}}_W$ and $\hat{\mathsf{r}}_W$, which do not belong to $Seq \cup \{\epsilon, 0, 1\}$. In addition, put $\hat{\boldsymbol{\epsilon}} = \epsilon$, $\hat{\mathbf{0}} = 0$ and $\hat{\mathbf{1}} = 1$. Then we define $\mathcal{A}(Q,x,y,z)$ to be the disjunction of the following formulas (1)–(26):

(1) $x = \hat{\mathsf{k}} \wedge z = \langle \hat{\mathsf{k}}, y \rangle$,

(2) $Seq_2(x) \ \wedge \ (x)_0 = \hat{\mathsf{k}} \ \wedge \ (x)_1 = z$,

(3) $x = \hat{\mathsf{s}} \ \wedge \ z = \langle \hat{\mathsf{s}}, y \rangle$,

(4) $Seq_2(x) \ \wedge \ (x)_0 = \hat{\mathsf{s}} \ \wedge \ z = \langle \hat{\mathsf{s}}, (x)_1, y \rangle$,

(5) $Seq_3(x) \ \wedge \ (x)_0 = \hat{\mathsf{s}} \ \wedge \ (\exists v,w)\,(Q((x)_1,y,v) \wedge Q((x)_2,y,w) \wedge Q(v,w,z))$,

(6) $x = \hat{\mathsf{p}} \ \wedge \ z = \langle \hat{\mathsf{p}}, y \rangle$,

(7) $Seq_2(x) \ \wedge \ (x)_0 = \hat{\mathsf{p}} \ \wedge \ z = \langle (x)_1, y \rangle$,

(8) $x = \hat{\mathsf{p}}_0 \ \wedge \ y = \langle z, (y)_1 \rangle$,

(9) $x = \hat{\mathsf{p}}_1 \ \wedge \ y = \langle (y)_0, z \rangle$,

(10) $x = \hat{*} \ \wedge \ z = \langle \hat{*}, y \rangle$,

(11) $Seq_2(x) \ \wedge \ (x)_0 = \hat{*} \ \wedge \ z = (x)_1 * y$,

(12) $x = \hat{\times} \ \wedge \ z = \langle \hat{\times}, y \rangle$,

(13) $Seq_2(x) \ \wedge \ (x)_0 = \hat{\times} \ \wedge \ z = (x)_1 \times y$,

(14) $x = \hat{\mathsf{p}}_W \ \wedge \ z = p(y)$,

(15) $x = \hat{\mathsf{c}}_\subseteq \ \wedge\ z = \langle \hat{\mathsf{c}}_\subseteq, y \rangle$,

(16) $\mathit{Seq}_2(x) \ \wedge\ (x)_0 = \hat{\mathsf{c}}_\subseteq \ \wedge\ z = c_\subseteq((x)_1, y)$,

(17) $x = \hat{\mathsf{d}}_W \ \wedge\ z = \langle \hat{\mathsf{d}}_W, y \rangle$,

(18) $\mathit{Seq}_2(x) \ \wedge\ (x)_0 = \hat{\mathsf{d}}_W \ \wedge\ z = \langle \hat{\mathsf{d}}_W, (x)_1, y \rangle$,

(19) $\mathit{Seq}_3(x) \ \wedge\ (x)_0 = \hat{\mathsf{d}}_W \ \wedge\ z = \langle \hat{\mathsf{d}}_W, (x)_1, (x)_2, y \rangle$,

(20) $\mathit{Seq}_4(x) \ \wedge\ (x)_0 = \hat{\mathsf{d}}_W \ \wedge\ (x)_3 = y \ \wedge\ z = (x)_1$,

(21) $\mathit{Seq}_4(x) \ \wedge\ (x)_0 = \hat{\mathsf{d}}_W \ \wedge\ (x)_3 \neq y \ \wedge\ z = (x)_2$,

(22) $x = \hat{\mathsf{r}}_W \ \wedge\ z = \langle \hat{\mathsf{r}}_W, y \rangle$,

(23) $\mathit{Seq}_2(x) \ \wedge\ (x)_0 = \hat{\mathsf{r}}_W \ \wedge\ z = \langle \hat{\mathsf{r}}_W, (x)_1, y \rangle$,

(24) $\mathit{Seq}_3(x) \ \wedge\ (x)_0 = \hat{\mathsf{r}}_W \ \wedge\ z = \langle \hat{\mathsf{r}}_W, (x)_1, (x)_2, y \rangle$,

(25) $\mathit{Seq}_4(x) \ \wedge\ (x)_0 = \hat{\mathsf{r}}_W \ \wedge\ z = \langle \hat{\mathsf{r}}_W, (x)_1, (x)_2, (x)_3, y \rangle$,

(26) $\mathit{Seq}_5(x) \ \wedge\ (x)_0 = \hat{\mathsf{r}}_W \ \wedge\ \mathit{Rec}_Q((x)_1, (x)_2, (x)_3, (x)_4, y, z)$.

This finishes the definition of the Q-positive $L_{\mathcal{P}}(Q)$ formula $\mathcal{A}(Q, x, y, z)$. Note that $\mathcal{A}$ is in fact a Σ_1 definition (modulo (Σ-Ref)). Hence, we know from standard recursion theory (cf. e.g. Hinman [21]) that the least fixed point of $\mathcal{A}$ is an r.e. set. The usual proof of this fact uses a careful construction *from below* by defining some sort of computability predicate, similar to the proof of Kleene's normal form theorem. Since we have all the sequence coding available in our weak setting, it is more or less straightforward to see that this construction can be carried through in PTCA^+. The details, however, are long and tedious. Moreover, one easily verifies that the so-obtained r.e. set - call it App - defines a fixed point of $\mathcal{A}$, where an obvious application of (Σ-Ref) is needed. PTCA^+ + (Σ-Ref) does not prove the minimality of App, of course. Instead, it is not difficult to establish the functionality of App. Summing up, we have the following theorem, whose proof is contained in the appendix of this paper.

Theorem 10 *There exists a Σ_1 formula $\mathit{App}(x, y, z)$ of $L_{\mathcal{P}}$ with free variables as shown so that PTCA^+ + (Σ-Ref) proves:*

1. $(\forall x, y, z)(\mathcal{A}(\mathit{App}, x, y, z) \leftrightarrow \mathit{App}(x, y, z))$.

2. $(\forall x, y, z_1, z_2)(\mathit{App}(x, y, z_1) \wedge \mathit{App}(x, y, z_2) \rightarrow z_1 = z_2)$.

Now the stage is set in order to describe a translation $(\cdot)^*$ from $\mathcal{L}_{\mathsf{PTO}}$ into $L_{\mathcal{P}}$. Let us first define an $L_{\mathcal{P}}$ formula $V_t^*(x)$ for each individual term t of $\mathcal{L}_{\mathsf{PTO}}$ so that the variable x does not occur in t. The formula $V_t^*(x)$ says that x is the value of t under the interpretation $*$. The exact definition is by induction on the complexity of t:

1. If t is an individual variable, then $V_t^*(x)$ is $(t = x)$.
2. If t is an individual constant, then $V_t^*(x)$ is $(\hat{t} = x)$.
3. If t is the individual term (rs), then

$$V_t^*(x) \;:=\; (\exists y_1, y_2)(V_r^*(y_1) \,\wedge\, V_s^*(y_2) \,\wedge\, App(y_1, y_2, x)).$$

In a second step we define the $*$ translation of an $\mathcal{L}_{\mathsf{PTO}}$ formula φ as follows:

4. If φ is the formula $W(t)$ or $t\!\downarrow$, then φ^* is

$$(\exists x)V_t^*(x).$$

5. If φ is the formula $(s = t)$, then φ^* is

$$(\exists x)(V_s^*(x) \,\wedge\, V_t^*(x)).$$

6. If φ is the formula $\neg\psi$, then φ^* is $\neg(\psi^*)$.
7. If φ is the formula $(\psi \; j \; \chi)$ for $j \in \{\vee, \wedge, \rightarrow\}$, then φ^* is $(\psi^* \; j \; \chi^*)$.
8. If φ is the formula $(\mathcal{Q}x)\psi$ for $\mathcal{Q} \in \{\exists, \forall\}$, then φ^* is $(\mathcal{Q}x)\psi^*$.

This finishes the description of the translation $(\cdot)^*$ from $\mathcal{L}_{\mathsf{PTO}}$ into $L_{\mathcal{P}}$. In a further step we have to verify the $*$ translation of the PTO axioms (1)–(25) in the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. In the following we only discuss axiom (23) for bounded primitive recursion and axiom (25) for set induction on W, $(\mathsf{S\text{-}I_W})$. The remaining axioms are easily verified by making use of Theorem 10.

Let us first turn to the bounded recursor r_W, and let us show the totality of r_W in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. We will realize the crucial role of Σ reflection $(\Sigma\text{-}\mathsf{Ref})$ for the first time.

Lemma 11 *The $*$ translation of axiom (23) about r_W is provable in the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$, i.e. $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ proves*

$$[(f : W \rightarrow W) \,\wedge\, (g : W^3 \rightarrow W) \,\wedge\, (b : W^2 \rightarrow W) \,\rightarrow\, (\mathsf{r}_W f g b : W^2 \rightarrow W)]^*.$$

Proof In the sequel we work informally in the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ and assume

$$(f : W \rightarrow W)^*, \tag{1}$$
$$(b : W^2 \rightarrow W)^*, \tag{2}$$
$$(g : W^3 \rightarrow W)^*. \tag{3}$$

If we spell out (1), (2) and (3) according to the translation $*$, we obtain

$$(\forall x)(\exists z)App(f, x, z), \tag{4}$$
$$(\forall x, w)(\exists z)App_2(b, x, w, z), \tag{5}$$
$$(\forall x, w, v)(\exists z)App_3(g, x, w, v, z). \tag{6}$$

It is our aim to show $(\mathsf{r}_W fgb : W^2 \to W)^*$, i.e.

$$(\forall x, w)(\exists z) App(\langle \hat{\mathsf{r}}_W, f, g, b, x \rangle, w, z), \tag{7}$$

which by Theorem 10 is equivalent to

$$(\forall x, w)(\exists z) Rec_{App}(f, g, b, x, w, z). \tag{8}$$

In the sequel fix arbitrary x_0 and y_0. Furthermore, by (4) choose z_0 so that $App(f, x_0, z_0)$. Now we obtain from (5) and Σ reflection (Σ-Ref) an a_1 so that

$$(\forall w \subseteq y_0)(\exists z \leq a_1) App_2^{a_1}(b, x_0, w, z). \tag{9}$$

If we set $a_2 = z_0 a_1$, then (6) and another application of (Σ-Ref) provide us with an a_3 so that

$$(\forall w \subseteq y_0)(\forall v \leq a_2)(\exists z \leq a_3) App_3^{a_3}(g, x_0, w, v, z). \tag{10}$$

Now set $a_4 = SqBd(a_2, y_0)$ and consider the statement $\widetilde{Rec}_{App}(f, g, b, x_0, y, z)$, which is given by the formula

$$\begin{aligned}
&\widetilde{Rec}_{App}(f, g, b, x_0, y, z) \;:= \\
&\quad (\exists v \leq a_4)[Seq(v) \;\wedge\; lh(v) = |y|1 \;\wedge\; (v)_\epsilon = z_0 \;\wedge \\
&\qquad\quad (\forall w \subseteq y)(w \neq \epsilon \to \\
&\qquad\qquad (\exists u_1 \leq a_3)(\exists u_2 \leq a_1)[App_3^{a_3}(g, x_0, w, (v)_{|p(w)|}, u_1) \;\wedge \\
&\qquad\qquad\quad App_2^{a_1}(b, x_0, w, u_2) \;\wedge \\
&\qquad\qquad\qquad (v)_{|w|} = u_1 | u_2]) \\
&\qquad\quad \wedge\; z = (v)_{|y|}].
\end{aligned}$$

In the following let us write $\varphi(y)$ for the $L_{\mathcal{P}}$ formula which is given by

$$y \subseteq y_0 \;\to\; (\exists z \leq a_2)\widetilde{Rec}_{App}(f, g, b, x_0, y, z).$$

Then one easily verifies that (9) and (10) imply

$$\varphi(\epsilon) \;\wedge\; (\forall y)(\varphi(y) \;\to\; \varphi(y0) \wedge \varphi(y1)). \tag{11}$$

Since $\varphi(y)$ is an extended Σ_1^b formula of $L_{\mathcal{P}}$, induction is available in PTCA^+ for φ. Hence, (11) implies $\varphi(y_0)$, from which we immediately derive

$$(\exists z) Rec_{App}(f, g, b, x_0, y_0, z). \tag{12}$$

Since x_0 and y_0 were arbitrary, we have shown (8), and this finishes our proof. $\square$

In a next step we show that the $*$ translation of set induction is provable in the system $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. Again the presence of Σ reflection (Σ-Ref) is crucial: the requirement $f \in P(W)$ allows one to "reflect" Σ_1 induction by Σ_1^b induction.

Lemma 12 *The* $*$ *translation of set induction* $(\mathsf{S\text{-}I_W})$ *is provable in the system* $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$*, i.e.* $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ *proves*

$$[f \in P(W) \wedge f\boldsymbol{\epsilon} = \mathbf{0} \wedge (\forall x \in W)(f(\mathsf{p}_W x) = \mathbf{0} \to fx = \mathbf{0}) \to (\forall x \in W)(fx = \mathbf{0})]^*.$$

Proof Let us work informally in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. Assume the $*$ translations of $f \in P(W)$, $f\boldsymbol{\epsilon} = \mathbf{0}$ and $(\forall x \in W)(f(\mathsf{p}_W x) = \mathbf{0} \to fx = \mathbf{0})$. Hence, we get

$$(\forall x)(\exists! y) App(f, x, y), \tag{1}$$

$$App(f, \epsilon, 0), \tag{2}$$

$$(\forall x)[App(f, x, 0) \to App(f, x0, 0) \wedge App(f, x1, 0)]. \tag{3}$$

Now fix an arbitrary x_0. By Σ reflection $(\Sigma\text{-}\mathsf{Ref})$ there exists an a so that

$$(\forall x \subseteq x_0)(\exists y \leq a) App^a(f, x, y). \tag{4}$$

As an immediate consequence we get that

$$(\forall x \subseteq x_0)(\forall y)[App(f, x, y) \leftrightarrow App^a(f, x, y)]. \tag{5}$$

Let us now write $\varphi(x)$ for the extended Σ^b_1 statement

$$x \subseteq x_0 \to App^a(f, x, 0).$$

Then one easily derives $(\forall x)\varphi(x)$ by Σ^b_1 induction, making use of (2), (3) and (5). Hence, we have obtained

$$App^a(f, x_0, 0), \tag{6}$$

and since x_0 was arbitrary, we have derived the $*$ translation of $(\forall x \in W)(fx = \mathbf{0})$ in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. This finishes our proof. $\square$

The reader may have noticed that in the proofs of Lemma 11 and Lemma 12 we did not make use of the full strength of the Σ reflection principle $(\Sigma\text{-}\mathsf{Ref})$. In fact, reflection is only needed for formulas of the shape $(\forall x \leq y)\varphi$, so that each positive universal and each negative existential quantifier in φ is sharply bounded. We can also dispense with the initial universal bounded quantifier, expect for obtaining the bound a_3 in equation (10) of the proof of Lemma 11. Similar remarks will apply to the treatment of the theory PTO^+ in Section 5, cf. the proof of Lemma 17. However, the *full* Σ reflection principle will be needed for analyzing the theory $\mathsf{PTO}^+ + (\Sigma^+\text{-}\mathsf{CP}_W)$ at the end of Section 5. For reasons of notational simplicity, we refrained from displaying the fine structure of Σ reflection in the formulation of theorems and proofs. This is perfectly justified by the fact that full Σ reflection does not take us beyond polynomial strength, cf. Corollary 7.

We are now in a position to state the following embedding theorem.

Theorem 13 *We have for all* $\mathcal{L}_{\mathsf{PTO}}$ *formulas* φ*:*

$$\mathsf{PTO} \vdash \varphi \implies \mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref}) \vdash \varphi^*.$$

From Corollary 7 and Theorem 9 we get the following equivalences. Here '$\equiv$' denotes a natural adaptation to our setting of Feferman's [13] notion of proof-theoretic equivalence.

Corollary 14 *We have the following proof-theoretic equivalences:*

$$\mathsf{PTO} \equiv \mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref}) \equiv \mathsf{PTCA}.$$

From Corollary 7 and the fact that an $\mathcal{L}_{\mathsf{PTO}}$ formula of the form $(\forall \vec{x} \in W)(t\vec{x} \in W)$ translates into a Π_2 statement under $(\cdot)^*$, we get the following crucial corollary.

Corollary 15 *Suppose that t is a closed term of $\mathcal{L}_{\mathsf{PTO}}$ and*

$$\mathsf{PTO} \vdash (\forall \vec{x} \in W)(t\vec{x} \in W).$$

Then t defines a polytime function on $\mathbb{W}$.

5 The theory PTO^+

In this section we propose an extension PTO^+ of PTO, which results from PTO by strengthening set induction to a form of complete induction on W which is related to *NP* induction, though it is formally much stronger. Furthermore, we briefly address a collection principle which does not raise the proof-theoretic strength of PTO^+.

In the following let the $\mathcal{L}_{\mathsf{PTO}}$ formula $N(f, g, x)$ be given by

$$N(f, g, x) \;:=\; (\exists y \leq fx)(gxy = \mathbf{0}).^2$$

In addition, $P(W^2)$ denotes the obvious generalization of $P(W)$ to binary (curried) characteristic functions on W, i.e.

$$f \in P(W^2) \;:=\; (\forall x, y \in W)(fxy = \mathbf{0} \vee fxy = \mathbf{1}).$$

Then PTO^+ is defined to be PTO, where set induction $(\mathsf{S}\text{-}\mathsf{I}_\mathsf{W})$ is replaced by the induction axiom $(\mathsf{N}\text{-}\mathsf{I}_\mathsf{W})$:

$$\begin{aligned}(f : W \to W) \wedge g \in P(W^2) \wedge N(f, g, \boldsymbol{\epsilon}) \wedge (\forall x \in W)(N(f, g, \mathsf{p}_W x) \to N(f, g, x))\\ \to (\forall x \in W) N(f, g, x).\end{aligned}$$

It is easy to see that set induction $(\mathsf{S}\text{-}\mathsf{I}_\mathsf{W})$ in fact follows from the above induction principle $(\mathsf{N}\text{-}\mathsf{I}_\mathsf{W})$.

We know from Theorem 9 that PTCA is contained in PTO via the translation $(\cdot)^W$. By making use of Lemma 8, it is now straightforward to verify that PTO^+ validates the *NP* induction principle of PTCA^+ with respect to $(\cdot)^W$. Hence, the following analogue of Theorem 9 holds.

[2] Bounded quantifiers are understood to be restricted to W

Theorem 16 *We have for every $L_{\mathcal{P}}$ formula $\varphi(\vec{x})$ with at most $\vec{x}$ free:*

$$\mathsf{PTCA}^{+} \vdash \varphi(\vec{x}) \implies \mathsf{PTO}^{+} \vdash \vec{x} \in W \to \varphi^{W}(\vec{x}).$$

On the other hand, we will now show that PTO^{+} is not stronger than PTO. In particular, we establish the $*$ translation of $(\mathsf{N}\text{-}\mathsf{I}_{\mathsf{W}})$ in $\mathsf{PTCA}^{+} + (\Sigma\text{-}\mathsf{Ref})$.

Lemma 17 *The $*$ translation of $(\mathsf{N}\text{-}\mathsf{I}_{\mathsf{W}})$ is provable in $\mathsf{PTCA}^{+} + (\Sigma\text{-}\mathsf{Ref})$.*

Proof In the following let us work informally in $\mathsf{PTCA}^{+} + (\Sigma\text{-}\mathsf{Ref})$, and assume the $*$ translation of the premise of $(\mathsf{N}\text{-}\mathsf{I}_{\mathsf{W}})$. The assumptions $(f : W \to W)^{*}$ and $(g \in P(W^2))^{*}$ yield

$$(\forall x)(\exists! z) App(f, x, z), \tag{1}$$
$$(\forall x, y)(\exists! z) App_2(g, x, y, z). \tag{2}$$

In the sequel fix an arbitrary x_0. By (1) and $(\Sigma\text{-}\mathsf{Ref})$ there exists an a_1 so that

$$(\forall x \subseteq x_0)(\exists z \leq a_1) App^{a_1}(f, x, z). \tag{3}$$

In addition, (2) and $(\Sigma\text{-}\mathsf{Ref})$ provide us with an a_2 so that

$$(\forall x \subseteq x_0)(\forall y \leq a_1)(\exists z \leq a_2) App_2^{a_2}(g, x, y, z). \tag{4}$$

In the following we write $\varphi(f, g, x)$ for the formula

$$(\exists z \leq a_1)(\exists y \leq z)[App_1^{a_1}(f, x, z) \wedge App_2^{a_2}(g, x, y, 0)].$$

Then it is straightforward to check from (3) and (4) that

$$(\forall x \subseteq x_0)[N^{*}(f, g, x) \leftrightarrow \varphi(f, g, x)]. \tag{5}$$

On the other hand, we have assumed

$$N^{*}(f, g, \epsilon), \tag{6}$$
$$(\forall x)(N^{*}(f, g, x) \to N^{*}(f, g, x0) \wedge N^{*}(f, g, x1)). \tag{7}$$

Hence, we can derive $(\forall x)\psi(x)$ by Σ_1^b induction from (5), (6) and (7), where $\psi(x)$ denotes the formula

$$x \subseteq x_0 \to \varphi(f, g, x).$$

We have shown $N^{*}(f, g, x_0)$, and since x_0 was arbitrary, this finishes our proof. □

The following analogue of Theorem 13 has been established.

Theorem 18 *We have for all $\mathcal{L}_{\mathsf{PTO}}$ formulas φ:*

$$\mathsf{PTO}^{+} \vdash \varphi \implies \mathsf{PTCA}^{+} + (\Sigma\text{-}\mathsf{Ref}) \vdash \varphi^{*}.$$

From Corollary 7 and Theorem 16 we can derive the same corollaries as in the previous section.

Corollary 19 *We have the following proof-theoretic equivalences:*

$$\mathsf{PTO}^+ \equiv \mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref}) \equiv \mathsf{PTCA}.$$

Corollary 20 *Suppose that t is a closed term of $\mathcal{L}_{\mathsf{PTO}}$ and*

$$\mathsf{PTO}^+ \vdash (\forall \vec{x} \in W)(t\vec{x} \in W).$$

Then t defines a polytime function on $\mathbb{W}$.

We finish this section by formulating a collection principle in $\mathcal{L}_{\mathsf{PTO}}$ which does not raise the proof-theoretic strength of PTO^+ either. The class of Σ^+ formulas of $\mathcal{L}_{\mathsf{PTO}}$ is inductively generated as follows:

1. Each atomic formula $W(t)$, $t\downarrow$ and $(s = t)$ is a Σ^+ formula.
2. If φ and ψ are Σ^+ formulas, then so also are $(\varphi \vee \psi)$ and $(\varphi \wedge \psi)$.
3. If φ is a Σ^+ formula, then so also are $(\forall x \leq y)\varphi$ and $(\exists x)\varphi$.

Now the scheme of Σ^+ collection on W, $(\Sigma^+\text{-}\mathsf{CP}_W)$, has the form

$$(\Sigma^+\text{-}\mathsf{CP}_W) \qquad (\forall x \leq y)(\exists z \in W)\varphi \;\rightarrow\; (\exists u \in W)(\forall x \leq y)(\exists z \leq u)\varphi,$$

where φ is a Σ^+ formula of $\mathcal{L}_{\mathsf{PTO}}$.

Now it is easy to verify that $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ validates the $*$ translation of each instance of $(\Sigma^+\text{-}\mathsf{CP}_W)$ and, therefore, $\mathsf{PTO}^+ + (\Sigma^+\text{-}\mathsf{CP}_W)$ does not go beyond polynomial strength, too. Here the full strength of $(\Sigma\text{-}\mathsf{Ref})$ is needed in order to handle $(\Sigma^+\text{-}\mathsf{CP}_W)$, of course.

6 Extensions to the Grzegorczyk hierarchy

Our approach described in the previous sections seems to be general enough. Let a_m denote the mth branch of the Ackermann function, and put $\mathcal{A}_n := \{a_m : 3 \leq m \leq n\}$ for $n \geq 3$. If we add the functions in $\mathcal{A}_n$ as base functions to our system, we get applicative theories G_n $(n \geq 3)$ so that the provably total functions of G_n are exactly the number-theoretic functions in the nth level of the Grzegorczyk hierarchy. In particular, G_3 captures the elementary functions, and it is proof-theoretically equivalent to $\mathsf{I}\Delta_0 + \mathsf{exp}$ in the terminology of Paris and Wilkie.

All these results are established in complete analogy to the results of the previous sections. Again it is possible to provide reductions to suitable subsystems of arithmetic, and it is not difficult to verify that Σ reflection $(\Sigma\text{-}\mathsf{Ref})$ can conservatively be added to the theory under consideration.

We finish this section by mentioning that in the case of the theories G_n it might be more natural to replace the predicate W by the usual predicate N for the natural numbers.

7 Final discussion

We have presented a theory PTO of polynomial time operations and binary words in the context of explicit mathematics. PTO contains Ferreira's theory PTCA, and it can be embedded into the system PTCA^+ plus the crucial principle of Σ reflection (Σ-Ref), thus yielding that the provably total functions of PTO are exactly the polytime functions. We have proposed an extension PTO^+ of PTO which is not stronger than PTO. Finally, we have sketched applicative theories G_n ($n \geq 3$) which capture the nth level of the Grzegorczyk hierarchy.

The theories PTO and PTO^+ are based on a *partial* form of term application, and the proof-theoretic reduction described in Section 4.2 makes substantial use of this fact. The question arises whether the assumption of a *total* application operation does raise the strength of PTO. More precisely, what is the exact proof-theoretic strength of $\mathsf{PTO} + (\mathsf{Tot})$, where (Tot) denotes the *axiom of totality*,

$$(\mathsf{Tot}) \qquad\qquad (\forall x, y)(xy\downarrow).$$

It is known that totality (Tot) does not raise the strength of various applicative theories of strength at least PRA, including systems with the so-called non-constructive minimum operator (cf. Jäger and Strahm [22]). The proof-theoretic strength of such systems is generally established by formalizing *total* term models in suitable systems of arithmetic, where essential use is made of the fact that *Church Rosser properties* of certain reduction relations can be formalized there.

If we consider a suitable total term model of PTO which is based on the usual reduction relation for total combinatory logic, then we do not know whether the corresponding Church Rosser property is provable in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. The usual proof that the combinatory reduction relation is Church Rosser is certainly formalizable in PRA, and a more sophisticated proof can already be carried through in $\mathsf{I}\Delta_0 + \mathsf{exp}$.[3] In particular, $\mathsf{G}_n + (\mathsf{Tot})$ *is not stronger than* G_n *(*$n \geq 3$*)*. However, we do not yet know whether $\mathsf{PTO} + (\mathsf{Tot})$ is stronger than PTO, although we strongly conjecture that the provably total functions of $\mathsf{PTO} + (\mathsf{Tot})$ are still computable in polynomial time.

Recently, Cantini [7] has established – among other things – that the provably total functions of the system $\mathsf{PTO} + (\mathsf{Tot})$ have *polynomial growth rate* only. His analysis of $\mathsf{PTO} + (\mathsf{Tot})$ makes use of partial cut elimination and an asymmetric interpretation with respect to the W predicate. However, it does not follow from Cantini's argument that the provably total functions of $\mathsf{PTO} + (\mathsf{Tot})$ are computable in *polynomial time*.

[3] This has recently been established by Duccio Pianigiani (manusript in preparation).

Appendix

In this appendix we give a proof of Theorem 10. In particular, we show that the operator form $\mathcal{A}(Q, x, y, z)$ has a Σ_1 fixed point *App* which is functional, provably in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. As already indicated, *App* will be constructed from below by making use of a specific computability predicate $Comp_{\mathcal{A}}(c)$, expressing that c is a computation sequence with respect to the operator form $\mathcal{A}$. Informally, a computation sequence c with respect to $\mathcal{A}$ is a sequence $c = \langle (c)_0, \ldots, (c)_{p(lh(c))} \rangle$ so that each $(c)_a$ is a sequence $\langle (c)_{a,0}, (c)_{a,1}, (c)_{a,2} \rangle$ of length 3 with the intended meaning that $(c)_{a,0}$ applied to $(c)_{a,1}$ yields $(c)_{a,2}$ in the sense of $\mathcal{A}$, and moreover, this is computed or "proved" by $\langle (c)_0, \ldots, (c)_{p(a)} \rangle$.

Let us first define $L_{\mathcal{P}}$ formulas $App_n(f, x_1, \ldots, x_n, y, a, c)$[4] for each $n \geq 1$ by induction on n as follows:

$$
\begin{aligned}
App_1(f, x_1, y, a, c) \quad &:= \quad (\exists b \subset a)((c)_b = \langle f, x_1, y \rangle), \\
App_{n+1}(f, x_1, \ldots, x_{n+1}, y, a, c) \quad &:= \\
&(\exists z \leq c)(\exists b \subset a)[App_n(f, x_1, \ldots, x_n, z, a, c) \land (c)_b = \langle z, x_{n+1}, y \rangle].
\end{aligned}
$$

The intended meaning of $App_n(f, x_1, \ldots, x_n, y, a, c)$ is that $f x_1 \ldots x_n \simeq y$ with respect to the sequence c restricted to the entries with index smaller than a.

Remark 21 *$App_n(f, x_1, \ldots, x_n, y, a, c)$ is an extended Σ^b_1 formula.*

In a next step we define an $L_{\mathcal{P}}$ formula $Rec_{App}(f, g, b, x, y, z, a, c)$. It defines the graph of the function which is defined from f and g by bounded primitive recursion with length bound b in the sense of the computation sequence c with entry indices smaller than a.

$$
\begin{aligned}
& Rec_{App}(f, g, b, x, y, z, a, c) := \\
& \quad (\exists v \leq c)[Seq(v) \ \land \ lh(v) = |y|1 \ \land \ App_1(f, x, (v)_\epsilon, a, c) \ \land \\
& \qquad\quad (\forall w \subseteq y)(w \neq \epsilon \rightarrow \\
& \qquad\qquad (\exists u_1, u_2)[App_3(g, x, w, (v)_{|p(w)|}, u_1, a, c) \ \land \ App_2(b, x, w, u_2, a, c) \\
& \qquad\qquad\qquad \land \ (v)_{|w|} = u_1 | u_2]) \\
& \qquad\quad \land \ z = (v)_{|y|}].
\end{aligned}
$$

Remark 22 *$Rec_{App}(f, g, b, x, y, z, a, c)$ is an extended Σ^b_1 formula.*

In the following let us write $\mathcal{A}_i(x, y, z)$ for the ith clause of the operator form $\mathcal{A}$ for $i \neq 5$ and $i \neq 26$. We are ready to define the $L_{\mathcal{P}}$ formula $Comp_{\mathcal{A}}$, which expresses that c is a computation sequence in the sense of the operator form $\mathcal{A}$.

$$
Comp_{\mathcal{A}}(c) \ := \ Seq(c) \ \land \ (\forall a \subset lh(c))[Seq_3((c)_a) \ \land \ C((c)_{a,0}, (c)_{a,1}, (c)_{a,2}, a)],
$$

[4] In the sequel it will always be clear from the number of parameters shown whether we mean $App_n(f, x_1, \ldots, x_n, y, a, c)$ or $App_n(f, x_1, \ldots, x_n, y)$.

where $C(x, y, z, a)$ is the disjunction of the $\mathcal{A}_i(x, y, z)$ for $i \neq 5$ and $i \neq 26$ plus the two disjuncts

(5') $Seq_3(x) \wedge (x)_0 = \hat{\mathsf{s}} \wedge$
$$(\exists v, w \leq c)[App_1((x)_1, y, v, a, c) \wedge App_1((x)_2, y, w, a, c) \wedge App_1(v, w, z, a, c)],$$

(26') $Seq_5(x) \wedge (x)_0 = \hat{\mathsf{r}}_W \wedge Rec_{App}((x)_1, (x)_2, (x)_3, (x)_4, y, z, a, c)$.

Remark 23 *$Comp_{\mathcal{A}}(c)$ is an extended Σ^b_1 formula.*

Now we are in a position to define the $L_{\mathcal{P}}$ formula $App(x, y, z)$, which expresses that there is a computation sequence c whose last entry is $\langle x, y, z\rangle$.

$$App(x, y, z) := (\exists c)[Comp_{\mathcal{A}}(c) \wedge last(c) = \langle x, y, z\rangle].$$

Remark 24 *$App(x, y, z)$ is equivalent to a Σ_1 formula, provably in PTCA^+.*

Remark 25 The reader might ask why we did at all make use of the operator form $\mathcal{A}(Q, x, y, z)$ in Section 4.2 instead of giving the above definition directly. The reason is conceptual clarity: the only properties which we used in order to establish the embedding of PTO into $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$ are the *fixed point* property and the *functionality* property, i.e. the two claims of Theorem 10. This is in full accordance with previous treatments of applicative theories, cf. e.g. Feferman and Jäger [15].

It remains to show that (i) App is a fixed point of the operator form $\mathcal{A}$, and (ii) App is functional, and in addition, (i) and (ii) are provable in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. In the following we work informally in the theory $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$, and we first want to show that App is functional.

Lemma 26 $\mathsf{PTCA} \vdash (\forall x, y, z_1, z_2)(App(x, y, z_1) \wedge App(x, y, z_2) \rightarrow z_1 = z_2)$.

PROOF We assume $Comp_{\mathcal{A}}(b) \wedge Comp_{\mathcal{A}}(c)$ and show the Δ^b_0 statement

$$v \subset lh(c) \rightarrow (\forall u \subseteq v)(\forall w \subset lh(b))[(b)_w = \langle (c)_{u,0}, (c)_{u,1}, (b)_{w,2}\rangle \rightarrow (b)_{w,2} = (c)_{u,2}]$$

by induction on v. Then our claim immediately follows. If $v = \epsilon$, then one of the clauses $\mathcal{A}_i$ for some i different from 5 and 26 applies, and our assertion is immediate. For the induction step let us assume that our assertion holds for some v; in order to verify it for $v1$, we have to distinguish several cases. If we are again in the case of one of the clauses $\mathcal{A}_i$ for i different from 5 and 26, then our claim follows as above. If clause (5') for the s combinator applies, then we are immediately done by the induction hypothesis. Finally, if we are in the case of clause (26') for r_W, then our assertion follows from the induction hypothesis and an obvious subsidiary induction. This settles our claim about the functionality of App. $\square$

It remains to show that $App(x, y, z)$ defines a fixed point of the positive operator $\mathcal{A}(Q, x, y, z)$, provably in $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref})$. We split the proof of the fixed point property into the two implications (i) $\mathcal{A}(App, x, y, z) \rightarrow App(x, y, z)$, and (ii) $App(x, y, z) \rightarrow \mathcal{A}(App, y, z)$.

Lemma 27 $\mathsf{PTCA}^+ + (\Sigma\text{-}\mathsf{Ref}) \vdash (\forall x,y,z)(\mathcal{A}(App,x,y,z) \to App(x,y,z))$.

PROOF Let us assume $\mathcal{A}(App,x,y,z)$. Then exactly one of the clauses (1)–(26) applies. If we have $\mathcal{A}_i(x,y,z)$ for an i different from 5 and 26, then we are done by the computation sequence $c = \langle\langle x,y,z\rangle\rangle$. Now suppose that clause (5) applies. Then we have $Seq_3(x) \wedge (x)_0 = \hat{\mathsf{s}}$, and there exist binary words v and w so that

$$App((x)_1,y,v) \;\wedge\; App((x)_2,y,w) \;\wedge\; App(v,w,z).$$

The above three conjuncts provide $\mathcal{A}$ computation sequences c_0, c_1 and c_2, and obviously the sequence $c = c_0 \frown c_1 \frown c_2 \frown \langle\langle x,y,z\rangle\rangle$ witnesses $App(x,y,z)$ as desired. Finally, we have to consider clause (26) for r_W. Therefore, assume

$$Seq_5(x) \;\wedge\; (x)_0 = \hat{\mathsf{r}}_W \;\wedge\; Rec_{App}((x)_1,(x)_2,(x)_3,(x)_4,y,z).$$

Then there exists a v, and by $(\Sigma\text{-}\mathsf{Ref})$ an a so that we have

$$\begin{aligned}
&Seq(v) \;\wedge\; lh(v) = |y|1 \;\wedge\; App^a((x)_1,(x)_4,(v)_\epsilon) \;\wedge\\
&\quad (\forall w \subseteq y)(w \neq \epsilon \to\\
&\qquad (\exists u_1,u_2 \leq a)[App_3^a((x)_2,(x)_4,w,(v)_{|p(w)|},u_1) \wedge App_2^a((x)_3,(x)_4,w,u_2) \wedge\\
&\qquad\qquad (v)_{|w|} = u_1|u_2])\\
&\quad \wedge\; z = (v)_{|y|}.
\end{aligned}$$

Now it is straightforward to establish the statement

$$\begin{aligned}
&y' \subseteq y \;\to\\
&\quad (\exists c \leq t(y',a))[Comp_{\mathcal{A}}(c) \wedge Rec_{App}((x)_1,(x)_2,(x)_3,(x)_4,y',(v)_{|y'|},p(lh(c)),c)]
\end{aligned}$$

by induction on y', where $t(y',a)$ is a suitable L term which provides an upper bound for the length of c (as a binary word). For example, choose the term $t(y',a)$ as $(aaaaa\overline{8} \times y'1)$. By setting $y' = y$, there now exists an $\mathcal{A}$ computation sequence c_y so that $Rec_{App}((x)_1,(x)_2,(x)_3,(x)_4,y,z,p(lh(c_y)),c_y)$. Our argument is finished, since the sequence $c'_y = c_y \frown \langle\langle x,y,z\rangle\rangle$ witnesses $App(x,y,z)$. $\Box$

Our last aim is to show the other direction of the fixed point property.

Lemma 28 $\mathsf{PTCA} \vdash (\forall x,y,z)(App(x,y,z) \to \mathcal{A}(App,x,y,z))$.

PROOF Suppose $App(x,y,z)$ holds for some binary words x,y and z. Hence, there exists a sequence c so that

$$Comp_{\mathcal{A}}(c) \;\wedge\; last(c) = \langle x,y,z\rangle.$$

If $\mathcal{A}_i(x,y,z)$ holds for some i different from 5 and 26, then our claim is trivial. If $\langle x,y,z\rangle$ was computed according to clause (5'), then an obvious decomposition of c yields the desired result. Finally, let us consider the case where we have

$$Seq_5(x) \;\wedge\; (x)_0 = \hat{\mathsf{r}}_W \;\wedge\; Rec_{App}((x)_1,(x)_2,(x)_3,(x)_4,y,z,p(lh(c)),c).$$

Then an easy decomposition of c yields $Rec_{App}((x)_1,(x)_2,(x)_3,(x)_4)$ as desired. $\Box$

This ends the proof of Theorem 10, and in fact also our paper.

References

[1] Barendregt, H. P. *The Lambda Calculus*, revised ed. North Holland, Amsterdam, 1984.

[2] Beeson, M. J. *Foundations of Constructive Mathematics: Metamathematical Studies*. Springer, Berlin, 1985.

[3] Buchholz, W., and Sieg, W. A note on polynomial time computable arithmetic. *Contemporary Mathematics 106* (1990), 51–55.

[4] Buss, S. R. *Bounded Arithmetic*. Bibliopolis, Napoli, 1986.

[5] Buss, S. R. A conservation result concerning bounded theories and the collection axiom. *Proceedings of the AMS 100*, 4 (1987), 709–715.

[6] Cantini, A. Asymmetric interpretation for bounded theories. *Mathematical Logic Quarterly*. To appear.

[7] Cantini, A. On the computational content of theories of operations with total application, June 1995. Handwritten notes.

[8] Cobham, A. The intrinsic computational difficulty of functions. In *Logic, Methodology and Philosophy of Science II*. North Holland, Amsterdam, 1964, pp. 24–30.

[9] Cook, S. A., and Kapron, B. M. Characterizations of the basic feasible functionals of finite type. In *Feasible Mathematics*, S. R. Buss and P. J. Scott, Eds. Birkhäuser, Basel, 1990, pp. 71–95.

[10] Cook, S. A., and Urquhart, A. Functional interpretations of feasibly constructive arithmetic. *Annals of Pure and Applied Logic 63*, 2 (1993), 103–200.

[11] Feferman, S. A language and axioms for explicit mathematics. In *Algebra and Logic*, J. Crossley, Ed., vol. 450 of *Lecture Notes in Mathematics*. Springer, Berlin, 1975, pp. 87–139.

[12] Feferman, S. Constructive theories of functions and classes. In *Logic Colloquium '78*, M. Boffa, D. van Dalen, and K. McAloon, Eds. North Holland, Amsterdam, 1979, pp. 159–224.

[13] Feferman, S. Hilbert's program relativized: proof-theoretical and foundational studies. *Journal of Symbolic Logic*, 53 (1988), 364–384.

[14] Feferman, S., and Jäger, G. Systems of explicit mathematics with non-constructive μ-operator. Part II. *Annals of Pure and Applied Logic*. To appear.

[15] Feferman, S., and Jäger, G. Systems of explicit mathematics with non-constructive μ-operator. Part I. *Annals of Pure and Applied Logic 65*, 3 (1993), 243–263.

[16] Ferreira, F. A note on a result of Buss concerning bounded theories and the collection scheme. Submitted to *Portugaliae Mathematica*.

[17] Ferreira, F. *Polynomial Time Computable Arithmetic and Conservative Extensions*. PhD thesis, Pennsylvania State University, 1988.

[18] Ferreira, F. Polynomial time computable arithmetic. *Contemporary Mathematics 106* (1990), 137–156.

[19] Ferreira, F. A feasible theory for analysis. *The Journal of Symbolic Logic 59*, 3 (1994), 1001–1011.

[20] Glass, T., and Strahm, T. Systems of explicit mathematics with non-constructive μ-operator and join. *Annals of Pure and Applied Logic*. To appear.

[21] Hinman, P. G. *Recursion-Theoretic Hierarchies*. Springer, Berlin, 1978.

[22] Jäger, G., and Strahm, T. Totality in applicative theories. *Annals of Pure and Applied Logic 74*, 2 (1995), 105–120.

[23] Seth, A. *Complexity theory of higher type functionals*. PhD thesis, Tata Institute of Fundamental Research, Bombay, 1994.

[24] Strahm, T. Theories with self-application of strength PRA. Master's thesis, Institut für Informatik und angewandte Mathematik, Universität Bern, 1992.

[25] Troelstra, A., and van Dalen, D. *Constructivism in Mathematics*, vol. II. North Holland, Amsterdam, 1988.

[26] Troelstra, A., and van Dalen, D. *Constructivism in Mathematics*, vol. I. North-Holland, Amsterdam, New York, 1988.

Address

Thomas Strahm, Institut für Informatik und angewandte Mathematik, Universität Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland, strahm@iam.unibe.ch

April 10, 1996