A proof-theoretic characterization of the basic feasible functionals

Thomas Strahm*

Version of July 2004

Final version to appear in **Theoretical Computer Science**

Abstract

We provide a natural characterization of the type two Mehlhorn-Cook-Urquhart basic feasible functionals as the provably total type two functionals of our (classical) applicative theory PT introduced in [30], thus providing a proof of a result claimed in the conclusion of [30]. This further characterization of the basic feasible functionals underpins their importance as a key candidate for the notion of type two feasibility.

1 Introduction

In this paper we deal with applicative theories in the spirit of Feferman's explicit mathematics (cf. [12, 13]). The paper is a successor to Strahm [30] (cf. also [29]), where so-called bounded applicative theories with a strong relationship to classes of computational complexity have been introduced and analyzed. For a more detailed background on applicative theories, we refer the reader to [30] and the articles cited there. Recently, Cantini [7] has studied substantial extensions of the theories introduced in [30] by choice and uniformity principles as well as a form of self-referential truth.

The main emphasis in [30] is on four applicative systems PT, PS, PTLS, and LS, and the determination of their provably total *type one* functions on binary words as the functions computable in polynomial time, polynomial

*Institut für Informatik und angewandte Mathematik, Universität Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland. Email: `strahm@iam.unibe.ch`. Homepage: `http://www.iam.unibe.ch/~strahm`

space, polynomial time *and* linear space, as well as linear space, respectively. The primary concern in the present paper is on the characterization of the provably total *type two* functionals of the system PT. The methods developed in the paper indeed also yield corresponding results for the systems PS, PTLS, and LS, cf. our remarks in the conclusion of this article.

It is a distinguished advantage of applicative theories that they allow for a very intrinsic and direct discussion of higher type issues, since higher types arise naturally in the untyped setting. Moreover, due to the fact that the untyped language does not a priori restrict the class of functionals which can be expressed, it makes perfect sense to consider the class of higher type functionals which are provably total in a given applicative system.

In the last decade, intense research efforts have been made in the area of so-called higher type complexity theory and, in particular, feasible functionals of higher types. This research is still ongoing and it is not yet clear what the right higher type analogue of the polynomial time computable functions is. Most prominent in the previous research is the class of so-called *basic feasible functionals* BFF, which has proved to be a very robust class with various kinds of interesting typed lambda calculus, function algebra, programming language, and, most importantly, oracle Turing machine characterizations.

The basic feasible functionals of type 2, BFF_2, were first studied in Melhorn [25]. More than ten years later in 1989, Cook and Urquhart [11] introduced the basic feasible functionals at all finite types in order to provide functional interpretations of feasibly constructive arithmetic; in particular, they defined a typed formal system PV^ω and used it to establish functional and realizability interpretations of an intuitionistic version of Buss' theory S^1_2. The basic feasible functionals BFF are exactly those functionals which can be defined by PV^ω terms. Subsequently, much work has been devoted to BFF, cf. e.g. Cook and Kapron [10, 20], Irwin, Kapron and Royer [19], Pezzoli [26], Royer [27], and Seth [28]. The survey article Clote [9] contains some of the key results concerning the basic feasible functionals.

The main result obtained in this article states that the *provably total type two functionals of* PT coincide with the *basic feasible functionals of type two*; this result has been announced in the conclusion of [30]. Moreover, in [30], an embedding of PV^ω into PT has been exhibited. The characterization of BFF_2 as the provably total type two functionals of the *classical* applicative theory PT gives further evidence for the naturalness and robustness of BFF_2.

The plan of this paper is as follows. In Section 2 we provide a function algebra definition of the type two basic feasible functionals. Section 3 is devoted to a recapitulation of the applicative theory PT introduced in [30].

In Section 4 we set up a suitable notion of provably total type two functional and show that the functionals in BFF_2 are provably total in PT. The main bulk of the paper is contained in Section 5, where basic feasible functionals are extracted from quasi cut-free PT derivations by means of an extension of the realizability argument used in [30]. We conclude the paper with some remarks concerning related work and extensions of the results obtained in this article.

2 The basic feasible functionals BFF

Let us start by giving the standard function algebra characterization of the basic feasible functionals of type two, BFF_2, cf. e.g. [9]. Alternatively, one could define BFF_2 as those functionals which are definable by closed type two terms of the typed lambda calculus PV^ω, cf. e.g. [11, 30].

In the sequel we denote by $\mathbb{W}$ the set of finite binary words $\{\epsilon, 0, 1, 00, 01, \ldots\}$, more compactly, $\mathbb{W} = \{0,1\}^*$. Here ϵ signifies the empty word. As usual, we let s_0 and s_1 denote the binary successor functions which concatenate 0 and 1 to the end of a given binary word, respectively. Moreover, $*$ and $\times$ stand for the binary operations of *word concatenation* and *word multiplication*, respectively, where $x \times y$ denotes the word x, length of y times concatenated with itself.

A *type 1 function* is a mapping from $\mathbb{W}$ to $\mathbb{W}$. We will write $\mathbb{W}^\mathbb{W}$ for the set of all functions from $\mathbb{W}$ to $\mathbb{W}$. A *type 2 functional* is a mapping from $(\mathbb{W}^\mathbb{W})^k \times \mathbb{W}^l$ to $\mathbb{W}$, for some k, l. We call such a mapping a *functional of rank* (k, l). In the sequel we let $F(\vec{f}, \vec{x})$, $G(\vec{f}, \vec{x}), \ldots$ range over type two functionals. An important type 2 functional is the so-called application functional Ap, which is defined as $\mathsf{Ap}(f, x) = f(x)$, for all f, x.

In the following we introduce some schemes for defining functionals. F is defined from $H, G_1, \ldots, G_m$ by *functional composition* if for all $\vec{f}, \vec{x}$,

$$F(\vec{f}, \vec{x}) = H(\vec{f}, G_1(\vec{f}, \vec{x}), \ldots, G_m(\vec{f}, \vec{x})).$$

F is defined from G by *expansion* if for all $\vec{f}, \vec{g}, \vec{x}, \vec{y}$,

$$F(\vec{f}, \vec{g}, \vec{x}, \vec{y}) = G(\vec{f}, \vec{x}).$$

F is defined from G, H_0, H_1, K by *bounded recursion on notation* (BRN) if for all $\vec{f}, \vec{x}, y$,

$$\begin{array}{rcll} F(\vec{f}, \vec{x}, \epsilon) & = & G(\vec{f}, \vec{x}), & \\ F(\vec{f}, \vec{x}, \mathsf{s}_i y) & = & H_i(\vec{f}, \vec{x}, y, F(\vec{f}, \vec{x}, y)), & \qquad (i = 0, 1) \\ F(\vec{f}, \vec{x}, y) & \leq & K(\vec{f}, \vec{x}, y). & \end{array}$$

Here $x \leq y$ signifies that the length of the word x is less than or equal to the length of the word y.

We are now ready to define the class BFF_2 of *basic feasible functionals of type 2*. BFF_2 is the smallest class of functionals such that

(i) the 0-ary function constant to ϵ, the identity function, the binary successor functions s_0 and s_1, word concatenation $*$, and word multiplication $\times$ belong to BFF_2;

(ii) the application functional Ap belongs to BFF_2;

(iii) BFF_2 is closed under functional composition and expansion;

(iv) BFF_2 is closed under bounded recursion on notation (BRN).

Since we will only be dealing with type two functionals in this paper, we will often simply write BFF instead of BFF_2. For an extensive survey on the many characterizations of the type two basic feasible functionals we refer the reader to the paper by Irwin, Kapron, and Royer [19].

3 The applicative theory PT

In this section we will recapitulate the theory PT that we have introduced and analyzed in Strahm [30], and we will survey some of its standard models and extensions.

The applicative theory PT is formulated in the language $\mathcal{L}$; it is a language of partial terms with *individual variables* $a, b, c, x, y, z, u, v, f, g, h, \dots$ (possibly with subscripts). $\mathcal{L}$ includes *individual constants* k, s (combinators), $\mathsf{p}, \mathsf{p}_0, \mathsf{p}_1$ (pairing and unpairing), d_W (definition by cases on binary words), ϵ (empty word) $\mathsf{s}_0, \mathsf{s}_1$ (binary successors), p_W (binary predecessor), $\mathsf{c}_\subseteq$ (initial subword relation), as well as the two constants $*$ (word concatenation) and $\times$ (word multiplication). Finally, $\mathcal{L}$ 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 *terms* $r, s, t, \dots$ of $\mathcal{L}$ (possibly with subscripts) are inductively generated from the variables and constants by means of application $\cdot$. We write ts instead of $\cdot(t, s)$ and follow the standard convention of association to the left when omitting brackets in applicative terms. As usual, (s, t) is a shorthand for $\mathsf{p}st$. Moreover, we use the abbreviations 0 and 1 for $\mathsf{s}_0\epsilon$ and $\mathsf{s}_1\epsilon$, respectively. Furthermore, we write $s \subseteq t$ instead of $\mathsf{c}_\subseteq st = 0$ and $s \leq t$ for

$\times 1s \subseteq \times 1t$; $s \subset t$ and $s < t$ are understood accordingly. Finally, $s{*}t$ stands for $*st$, and $s{\times}t$ for $\times st$.

The *formulas* $A, B, C, \dots$ of $\mathcal{L}$ (possibly with subscripts) are built from the atomic formulas $(s = t)$, $s{\downarrow}$ and $\mathsf{W}(s)$ by closing under negation, disjunction, conjunction, implication, as well as existential and universal quantification over individuals.

Our conventions concerning substitutions are as follows. As usual we write $t[\vec{s}/\vec{x}]$ and $A[\vec{s}/\vec{x}]$ for the substitution of the terms $\vec{s}$ for the variables $\vec{x}$ in the term t and the formula A, respectively. In this connection we often write $A(\vec{x})$ instead of A and $A(\vec{s})$ instead of $A[\vec{s}/\vec{x}]$.

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

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

In the following we will use the following natural abbreviations concerning the predicate W ($\vec{s} = s_1, \dots, s_n$):

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

We call an $\mathcal{L}$ formula *positive* if it is built from the atomic formulas by means of disjunction, conjunction as well as existential and universal quantification over individuals; i.e., the positive formulas are exactly the implication and negation free $\mathcal{L}$ formulas. We let $\mathcal{POS}$ stand for the collection of positive formulas. Further, an $\mathcal{L}$ formula is called W *free*, if the relation symbol W does not occur in it.

Most important in the sequel are the so-called *bounded (with respect to* W*) existential formulas* or $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ *formulas* of $\mathcal{L}$. A formula $A(f, x)$ belongs to the class $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ if it has the form $(\exists y \le fx)B(f, x, y)$ for $B(f, x, y)$ a *positive and* W *free* formula. It is important to recall here that bounded quantifiers range over W, i.e., $(\exists y \le fx)B(f, x, y)$ stands for

$$(\exists y \in \mathsf{W})[y \le fx \ \wedge \ B(f, x, y)].$$

Further observe that the matrix B of a $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ formula can have unrestricted existential and universal individual quantifiers, not ranging over W, however.

Assuming that the bounding operation f in a $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ formula has polynomial growth, $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ formulas can be seen as a very abstract applicative analogue of Buss' Σ^b_1 formulas (cf. [6]) or Ferreira's NP formulas (cf. [14, 15]). Notice, however, whereas the latter classes of formulas define exactly the NP predicates, $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ formulas of $\mathcal{L}$ in general define undecidable sets in the standard models of PT described below; indeed already equality between terms is undecidable in these models.

We now introduce the applicative theory PT. The underlying logic of PT is the *classical* logic of partial terms due to Beeson [2, 3]; it corresponds to E^+ logic with strictness and equality of Troelstra and Van Dalen [31]. According to this logic, quantifiers range over defined objects only, so that the usual axioms for $\exists$ and $\forall$ are modified to

$$A(t) \wedge t{\downarrow} \;\rightarrow\; (\exists x)A(x) \quad \text{and} \quad (\forall x)A(x) \wedge t{\downarrow} \;\rightarrow\; A(t),$$

and one further assumes that $(\forall x)(x{\downarrow})$. The *strictness axioms* claim that if a compound term is defined, then so also are all its subterms, and if a positive atomic statement holds, then all terms involved in that statement are defined. Note that $t{\downarrow} \leftrightarrow (\exists x)(t = x)$, so definedness need not be taken as basic symbol. The reader is referred to [2, 3, 31] for a detailed exposition of the logic of partial terms.

The non-logical axioms of PT first of all include the defining axioms for the constants and relations of $\mathcal{L}$, which are divided into the following six groups.

I. Partial combinatory algebra and pairing

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

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

(3) $\mathsf{p_0}(x, y) = x \,\wedge\, \mathsf{p_1}(x, y) = y$.

II. Definition by cases on W

(4) $a \in \mathsf{W} \wedge b \in \mathsf{W} \wedge a = b \;\rightarrow\; \mathsf{d_W}xyab = x$,

(5) $a \in \mathsf{W} \wedge b \in \mathsf{W} \wedge a \neq b \;\rightarrow\; \mathsf{d_W}xyab = y$.

III. Closure, binary successors and predecessor

(6) $\epsilon \in \mathsf{W} \,\wedge\, (\forall x \in \mathsf{W})(\mathsf{s_0}x \in \mathsf{W} \wedge \mathsf{s_1}x \in \mathsf{W})$,

(7) $\mathsf{s}_0 x \neq \mathsf{s}_1 y \ \wedge \ \mathsf{s}_0 x \neq \epsilon \ \wedge \ \mathsf{s}_1 x \neq \epsilon$,

(8) $\mathsf{p_W} : \mathsf{W} \to \mathsf{W} \ \wedge \ \mathsf{p_W}\epsilon = \epsilon$,

(9) $x \in \mathsf{W} \ \to \ \mathsf{p_W}(\mathsf{s}_0 x) = x \ \wedge \ \mathsf{p_W}(\mathsf{s}_1 x) = x$,

(10) $x \in \mathsf{W} \wedge x \neq \epsilon \ \to \ \mathsf{s}_0(\mathsf{p_W} x) = x \vee \mathsf{s}_1(\mathsf{p_W} x) = x$.

IV. Initial subword relation.

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

(12) $x \in \mathsf{W} \ \to \ (x \subseteq \epsilon \leftrightarrow x = \epsilon)$,

(13) $x \in \mathsf{W} \wedge y \in \mathsf{W} \wedge y \neq \epsilon \ \to \ (x \subseteq y \leftrightarrow x \subseteq \mathsf{p_W} y \vee x = y)$,

V. Word concatenation.

(14) $* : \mathsf{W}^2 \to \mathsf{W}$,

(15) $x \in \mathsf{W} \ \to \ x{*}\epsilon = x$,

(16) $x \in \mathsf{W} \wedge y \in \mathsf{W} \ \to \ x{*}(\mathsf{s}_0 y) = \mathsf{s}_0(x{*}y) \ \wedge \ x{*}(\mathsf{s}_1 y) = \mathsf{s}_1(x{*}y)$.

VI. Word multiplication.

(17) $\times : \mathsf{W}^2 \to \mathsf{W}$,

(18) $x \in \mathsf{W} \ \to \ x{\times}\epsilon = \epsilon$,

(19) $x \in \mathsf{W} \wedge y \in \mathsf{W} \ \to \ x{\times}(\mathsf{s}_0 y) = (x{\times}y){*}x \ \wedge \ x{\times}(\mathsf{s}_1 y) = (x{\times}y){*}x$.

Finally, and most crucially, PT includes the induction axioms ($\Sigma^{\mathsf{b}}_{\mathsf{W}}$-$\mathsf{I_W}$). This principle allows induction along W with respect to formulas in the class $\Sigma^{\mathsf{b}}_{\mathsf{W}}$, under the proviso that the bounding operation f has the right type. Accordingly, the scheme ($\Sigma^{\mathsf{b}}_{\mathsf{W}}$-$\mathsf{I_W}$) of $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ *notation induction on* W includes for each formula $A(x) \equiv (\exists y \leq fx)B(f, x, y)$ in the formula class $\Sigma^{\mathsf{b}}_{\mathsf{W}}$,

$$(\Sigma^{\mathsf{b}}_{\mathsf{W}}\text{-}\mathsf{I_W}) \qquad \begin{aligned} & f : \mathsf{W} \to \mathsf{W} \ \wedge \ A(\epsilon) \ \wedge \ (\forall x \in \mathsf{W})(A(x) \to A(\mathsf{s}_0 x) \wedge A(\mathsf{s}_1 x)) \\ & \qquad\qquad \to (\forall x \in \mathsf{W})A(x) \end{aligned}$$

Two fundamental consequences of the partial combinatory algebra axioms (1) and (2) of PT are the theorem about *lambda abstraction* and the *recursion or fixed point theorem*, cf. [30] and [2, 12] for a proof. Clearly, recursion nicely demonstrates the power of self-application. It will be an essential tool for defining functionals in the next section of this paper.

Let us briefly turn to some models of the theory PT. First of all, the model *PRO* of *partial recursive operations* is the standard recursion-theoretic model of PT. The universe of *PRO* consists of the set of all 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 $\mathcal{L}$ so that all the axioms of PT are true in *PRO*.

A further important model of PT is the open term model $\mathcal{M}(\lambda\eta)$. This model is based on the usual $\lambda\eta$ reduction of the untyped lambda calculus (cf. [1, 17]) and exploits the well-known equivalence between combinatory logic with extensionality and $\lambda\eta$. In order to deal with the constants different from k and s, one extends $\lambda\eta$ reduction by the obvious reduction clauses for these new constants and checks that the so-obtained new reduction relation enjoys the Church Rosser property.[1]

The universe of the model $\mathcal{M}(\lambda\eta)$ now consists of the set of all $\mathcal{L}$ terms. Equality $=$ means reduction to a common reduct and W is interpreted as the set of all $\mathcal{L}$ terms t so that t reduces to a "canonical" word $\overline{w}$ for some $w \in \mathbb{W}$.[2] Finally, the constants are interpreted as indicated above and application of t to s is simply the term ts. As usual, we write $\mathcal{M}(\lambda\eta) \models A$ in order to express that the formula A is true in $\mathcal{M}(\lambda\eta)$. Let us observe that in the term model $\mathcal{M}(\lambda\eta)$, two important additional principles are satisfied, namely the axioms (Tot) for *totality of application* and (Ext) for *extensionality of operations*,

$$(\mathsf{Tot}) \qquad (\forall x, y)(xy\downarrow) \qquad\qquad (\mathsf{Ext}) \qquad (\forall f, g)[(\forall x)(fx = gx) \to f = g]$$

There are many more interesting models of the combinatory axioms, which can easily be extended to models of PT. These include further recursion-theoretic models, term models, continuous models, generated models, and set-theoretic models. For detailed descriptions and results the reader is referred to Beeson [2], Feferman [13], and Troelstra and van Dalen [32].

[1] Actually, suitable interpretations for the constants $\mathsf{c}_{\subseteq}$, $*$ and $\times$ can also be given using the other constants of $\mathcal{L}$; this is easily accomplished by making use of the above-mentioned recursion or fixed point theorem.

[2] For each $w \in \mathbb{W}$, we let $\overline{w}$ denote the canonical closed $\mathcal{L}$ term for w which is constructed form ϵ by means of the successor operations s_0 and s_1; in the sequel we sometimes identify $\overline{w}$ with w when working in the language $\mathcal{L}$.

4 BFF in PT

In this section we first clarify the notion of a provably total type two functional in a given applicative theory. Then we show that the basic feasible functionals of type two are provably total in PT. Indeed, this result already follows from our embedding of PV^ω into PT in [30], but we recapitulate the argument below in order to make the paper self-contained.

Assume that $\mathcal{M}$ is a standard structure for $\mathcal{L}$, i.e., a structure where the predicate W for binary words obtains a standard interpretation. What does it mean for a functional F of rank (k, l) to be definable in $\mathcal{M}$? To answer this question, let us temporarily write $\mathcal{W}^\mathcal{W}$ for the set of all individuals f in the universe $|\mathcal{M}|$ of $\mathcal{M}$ so that $f : \mathsf{W} \to \mathsf{W}$ is true in $\mathcal{M}$. Further, for such an f, write $\hat{f}$ for the function from $\mathbb{W}$ to $\mathbb{W}$ that is defined by f in $\mathcal{M}$.

Now we call a type 2 functional F of rank (k, l) *definable in* $\mathcal{M}$, if there exists a *closed* $\mathcal{L}$ term t_F so that we have for all $f_1, \ldots, f_k$ in $\mathcal{W}^\mathcal{W}$ and all $w_1, \ldots, w_l$ in $\mathbb{W}$ that

$$\mathcal{M} \models t_F f_1 \ldots f_k \overline{w}_1 \ldots \overline{w}_l = \overline{F(\hat{f}_1, \ldots, \hat{f}_k, w_1, \ldots, w_l)}.^3$$

Observe that if $g_1, \ldots, g_k, h_1, \ldots, h_k \in \mathcal{W}^\mathcal{W}$ such that $\hat{g}_1 = \hat{h}_1, \ldots, \hat{g}_k = \hat{h}_k$, then definability of F in $\mathcal{M}$ via t_F yields for all $w_1, \ldots, w_l \in \mathbb{W}$,

$$\mathcal{M} \models t_F g_1 \ldots g_k \overline{w}_1 \ldots \overline{w}_l = t_F h_1 \ldots h_k \overline{w}_1 \ldots \overline{w}_l.$$

As a final preparatory step towards the crucial notion of a provably total type 2 functional, let us use the following abbreviation in the language $\mathcal{L}$,

$$t : (\mathsf{W}^\mathsf{W})^k \times \mathsf{W}^l \to \mathsf{W} \;:=\; (\forall \vec{f} : \mathsf{W} \to \mathsf{W})(\forall \vec{x} \in \mathsf{W}) t\vec{f}\vec{x} \in \mathsf{W}.$$

Here $\vec{f}$ and $\vec{x}$ have length k and l, respectively. Now let T be an $\mathcal{L}$ theory and F a type 2 functional of rank (k, l). We call F *provably total in* T, if there exists a closed $\mathcal{L}$ term t_F such that

(i) $\mathsf{T} \vdash t_F : (\mathsf{W}^\mathsf{W})^k \times \mathsf{W}^l \to \mathsf{W}$, and, in addition,

(ii) t_F defines F in the open term model $\mathcal{M}(\lambda\eta)$.

In (ii) we have chosen $\mathcal{M}(\lambda\eta)$, since the open term model is the standard model of the theory PT + (Tot) + (Ext), for which proof-theoretic upper

[3] Note that if t_F defines F in $\mathcal{M}$, then it defines each functional F' which differs from F on function arguments not in $\mathcal{W}^\mathcal{W}$ only. For this reason, we will identify F with all such F's when using the notion of definability of a type two functional in a model $\mathcal{M}$.

bounds will be established in the next section. If one is only interested in PT without (Ext) and (Tot), one could equally well choose the recursion-theoretic model *PRO* in (ii). Moreover, if the reader finds it unnatural that not all set-theoretic functions from $\mathbb{W}$ to $\mathbb{W}$ live in $\mathcal{M}(\lambda\eta)$ or *PRO*, it is worth mentioning that there are suitable extensions of these models, with codes added for *all* functions from $\mathbb{W}$ to $\mathbb{W}$, cf. e.g. Feferman [12]. Moreover, the arguments given in this paper are easily seen to work for these extended models.

We are now ready to show that all functionals in BFF are provably total in our applicative theory PT. The argument given below makes crucial use of the recursion theorem and $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ notation induction.

Theorem 1 *The basic feasible functionals are provably total in* PT.

Proof Clearly, the initial functions (i) of BFF are easily shown to be provably total in PT. In addition, the application functional Ap is represented by the $\mathcal{L}$ term $\lambda f, x.fx$. Further, the provably total functions of PT are readily seen to be closed under functional composition and expansion. Hence, the crucial step of the proof consists in establishing closure under bounded recursion on notation (BRN).

Firstly, we will need the cut-off operator $|$ in order to describe bounded recursion in PT. Informally speaking, $t \mid s$ is t if $t \leq s$ and s else. More formally, we can make use of definition by cases $\mathsf{d_W}$ and the characteristic function $\mathsf{c}_{\subseteq}$ in order to define $|$; then $t \mid s$ simply is an abbreviation for the $\mathcal{L}$ term $\mathsf{d_W} ts(\mathsf{c}_{\subseteq}(1{\times}t)(1{\times}s))0$.

Assume now that the basic feasible functional F of rank (k,l) has been defined from G, H_0, H_1, and K by bounded recursion on notation. By induction hypothesis, we know that the latter functionals are provably total in PT via $\mathcal{L}$ terms t_G, t_{H_0}, t_{H_1}, and t_K, respectively. Next, we invoke the *recursion theorem* and definition by cases on W in order to find a closed $\mathcal{L}$ term t_F, so that we have for all $\vec{f}, \vec{x}$, and $y \in \mathsf{W}$,

$$\begin{aligned} t_F\vec{f}\vec{x}\epsilon &\simeq t_G\vec{f}\vec{x} \mid t_K\vec{f}\vec{x}\epsilon, \\ t_F\vec{f}\vec{x}(\mathsf{s}_i y) &\simeq t_{H_i}\vec{f}\vec{x}y(t_F\vec{f}\vec{x}y) \mid t_K\vec{f}\vec{x}(\mathsf{s}_i y) \qquad (i = 0,1). \end{aligned}$$

Assume now, in addition, that $\vec{f} : \mathsf{W} \to \mathsf{W}$ and $\vec{x} \in \mathsf{W}$, and consider the $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ formula $A(y)$ given as follows,

$$A(y) \;:=\; (\exists z \leq t_K\vec{f}\vec{x}y)(t_F\vec{f}\vec{x}y = z).$$

Observe that our assumptions readily yield $t_K\vec{f}\vec{x} : \mathsf{W} \to \mathsf{W}$. Using the above recursion equations and the fact that t_G, t_{H_0}, t_{H_1}, and t_K are already known to have the correct type, provably in PT, we can immediately derive by $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ notation induction on W the statement $(\forall y \in \mathsf{W})A(y)$. All together we have shown that PT proves $t_F : (\mathsf{W}^{\mathsf{W}})^k \times \mathsf{W}^l \to \mathsf{W}$. This ends our proof that the basic feasible functionals are provably total in PT. □

5 Extracting Bff's from PT derivations

In this section we will show that the lower bound established in the previous section is indeed sharp, i.e., each provably total type 2 functional of PT is basic feasible. Our upper bound argument is in fact a refinement of the argument used in Strahm [30] in order to show that the provably total type 1 functions of PT are computable in polynomial time. As we have already mentioned, we will directly treat the extension of PT by the axioms (Tot) and (Ext), which we will call PT^+ in the sequel. Observe that PT^+ proves $t\downarrow$ for each $\mathcal{L}$ term t, so that the logic of partial terms can be replaced by usual first order classical predicate calculus with equality.

Similarly to [30], the upper bound argument proceeds in two steps. Firstly, a sequent-style reformulation of PT^+ is used to show that cut formulas in PT^+ derivations can be restricted to be *positive*. The second crucial step consists in providing a realizability interpretation in the standard open term model $\mathcal{M}(\lambda\eta)$ of PT^+ in order to extract type two functionals in Bff from quasi-normal PT^+ derivations.

In the following we let $\Gamma, \Delta, \Lambda, \ldots$ range over finite *sequences* of formulas in the language $\mathcal{L}$; a *sequent* is a formal expression of the form $\Gamma \Rightarrow \Delta$. As usual, the natural interpretation of the sequent $A_1, \ldots, A_n \Rightarrow B_1, \ldots, B_m$ is $(A_1 \wedge \cdots \wedge A_n) \to (B_1 \vee \cdots \vee B_m)$. Our sequent-style reformulation of PT^+ is presented in detail in [30] so that that we can confine ourselves to a brief sketch here. We presuppose the context-sharing version of Gentzen's sequent calculus LK as our logical basis. The main task is to set up a sequent-style reformulation of PT^+ so that all main formulas of non-logical axioms and rules are *positive*. This is easily achieved for axioms (1)–(19) of PT as well as the equality and extensionality axioms, cf. [30]. Moreover, the axiom schema ($\Sigma^{\mathsf{b}}_{\mathsf{W}}$-$\mathsf{I}_{\mathsf{W}}$) of PT^+ for $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ notation induction on W is replaced by a suitable rule of inference in the Gentzen-style formulation of PT^+. For that purpose, let $A(u)$ be of the form $(\exists y \le tu)B(u, y)$ for B being a positive and W free formula. Then an instance of the ($\Sigma^{\mathsf{b}}_{\mathsf{W}}$-$\mathsf{I}_{\mathsf{W}}$) notation induction rule is given as follows,

$$\frac{\Gamma, \mathsf{W}(u) \Rightarrow \mathsf{W}(tu), \Delta \quad \Gamma \Rightarrow A(\epsilon), \Delta \quad \Gamma, \mathsf{W}(u), A(u) \Rightarrow A(\mathsf{s}_i u), \Delta}{\Gamma, \mathsf{W}(s) \Rightarrow A(s), \Delta}$$

Here u denotes a fresh variable not occurring in Γ, Δ and i ranges over $0, 1$, i.e., the rule of inference has four premises. Clearly, the main formulas of this rule are positive.

It should be clear that we have provided an adequate sequent-style reformulation of PT^+; in particular, the axiom schema ($\Sigma^{\mathsf{b}}_{\mathsf{W}}$-$\mathsf{I}_{\mathsf{W}}$) as given in Section 3 of this paper is readily derivable by means of the corresponding rule of inference stated above, where as usual the presence of side formulas is crucial. In the following we often identify PT^+ with its Gentzen-style version and write $\mathsf{PT}^+ \vdash \Gamma \Rightarrow \Delta$ in order to express that the sequent $\Gamma \Rightarrow \Delta$ is derivable in PT^+. Moreover, we will use the notation $\mathsf{PT}^+ \underset{\star}{\vdash} \Gamma \Rightarrow \Delta$ if the sequent $\Gamma \Rightarrow \Delta$ has a proof in PT^+ so that all cut formulas appearing in this proof are *positive*.

Due to the fact that all the main formulas of non-logical axioms and rules of PT^+ are positive, we now obtain the desired partial cut elimination theorem for PT^+. Its proof is immediate from the well-known proof of the cut elimination theorem for LK (cf. e.g. Girard [16]) and is therefore omitted.

Theorem 2 (Partial cut elimination for PT^+) *We have for all sequents $\Gamma \Rightarrow \Delta$ that $\mathsf{PT}^+ \vdash \Gamma \Rightarrow \Delta$ entails $\mathsf{PT}^+ \underset{\star}{\vdash} \Gamma \Rightarrow \Delta$.*

The second crucial step in our upper bound argument consists in a realizability interpretation applied to quasi cut-free PT^+ derivations of sequents of suitable formulas. Whereas in [30] we could confine ourselves to sequents of *positive* formulas, in the present context we need to consider a larger class of formulas since we want to extract computational information about the provably total functionals of type 2. Recall that the totality statements in which we are interested have the general form $t : (\mathsf{W}^{\mathsf{W}})^k \times \mathsf{W}^l \to \mathsf{W}$; this can be rewritten in sequent and free variable form in the following manner,

$$f_1 : \mathsf{W} \to \mathsf{W}, \ldots, f_k : \mathsf{W} \to \mathsf{W},\, x_1 \in \mathsf{W}, \ldots, x_l \in \mathsf{W} \;\Rightarrow\; t f_1 \ldots f_k x_1 \ldots x_l \in \mathsf{W}$$

This motivates the following definition. A formula A belongs to the class $\mathcal{C}_1$, if A is in $\mathcal{POS}$, or there are formulas B, C in $\mathcal{POS}$ so that A has the form $(B \to C)$ or $(\forall x)(B \to C)$. Clearly, the above sequent consists of $\mathcal{C}_1$ formulas only, and, moreover, the formula on the right hand side of $\Rightarrow$ is positive.

The following corollary directly follows from the above partial cut elimination theorem and a quick inspection of the axioms and rules of PT^+. It will be crucial for our realizability arguments below.

Corollary 3 *Assume that Γ and Δ are finite sequences of formulas in $\mathcal{C}_1$ and $\mathcal{POS}$, respectively, such that $\mathsf{PT}^+ \vdash \Gamma \Rightarrow \Delta$. Then $\Gamma \Rightarrow \Delta$ has a PT^+ derivation all of whose sequents consist of $\mathcal{C}_1$ formulas on the left of $\Rightarrow$ and $\mathcal{POS}$ formulas on the right of $\Rightarrow$.*

In a next step we now want to define realizability for formulas in the class $\mathcal{C}_1$. As already mentioned above, we will make use of the standard open term model $\mathcal{M}(\lambda\eta)$ of PT^+. We first spell out our realizability notion for the class of positive formulas and then extend it to all $\mathcal{C}_1$ formulas.

Realizers $\rho, \sigma, \tau, \ldots$ of positive formulas are simply elements of the set $\mathbb{W}$ of binary words. Below we presuppose a polynomial time pairing operation $\langle \cdot, \cdot \rangle$ on $\mathbb{W}$ with associated projections $(\cdot)_0$ and $(\cdot)_1$. Further, for each natural number i let us write i_2 for the binary notation of i. The crucial notion $\rho \;\mathbf{r}\; A$ ("ρ realizes A") for $\rho \in \mathbb{W}$ and A a positive formula, is given inductively as spelled out below. It corresponds to the definition of realizability in [30].[4]

$$
\begin{array}{lll}
\rho \;\mathbf{r}\; \mathsf{W}(t) & \text{if} & \mathcal{M}(\lambda\eta) \models t = \overline{\rho}, \\
\rho \;\mathbf{r}\; (t_1 = t_2) & \text{if} & \rho = \epsilon \text{ and } \mathcal{M}(\lambda\eta) \models t_1 = t_2, \\
\rho \;\mathbf{r}\; (A \wedge B) & \text{if} & \rho = \langle \rho_0, \rho_1 \rangle \text{ and } \rho_0 \;\mathbf{r}\; A \text{ and } \rho_1 \;\mathbf{r}\; B, \\
\rho \;\mathbf{r}\; (A \vee B) & \text{if} & \rho = \langle i, \rho_0 \rangle \text{ and either } i = 0 \text{ and } \rho_0 \;\mathbf{r}\; A \text{ or} \\
 & & \qquad\qquad\qquad\quad i = 1 \text{ and } \rho_0 \;\mathbf{r}\; B, \\
\rho \;\mathbf{r}\; (\forall x)A(x) & \text{if} & \rho \;\mathbf{r}\; A(t) \text{ for all terms } t, \\
\rho \;\mathbf{r}\; (\exists x)A(x) & \text{if} & \rho \;\mathbf{r}\; A(t) \text{ for some term } t.
\end{array}
$$

If Γ denotes the sequence of positive formulas $A_1, \ldots, A_n$ and $\vec{\rho} = \rho_1, \ldots, \rho_n$, then we write $\vec{\rho} \;\mathbf{r}\; \Gamma$ if $\rho_i \;\mathbf{r}\; A_i$ for all $1 \leq i \leq n$. Moreover, if Δ denotes the sequence $B_1, \ldots, B_m$ of positive formulas, then we say that ρ disjunctively realizes the sequence Δ, in symbols, $\rho \;\mathbf{r}_\vee\; \Delta$, if $\rho = \langle i_2, \rho_0 \rangle$ for some $1 \leq i \leq m$ and $\rho_0 \;\mathbf{r}\; B_i$. Hence, according to the notion $\rho \;\mathbf{r}_\vee\; \Delta$, the sequence Δ is understood disjunctively, i.e. as the succedent of a given sequent.

We proceed by extending our realizability notion from positive formulas to formulas in the class $\mathcal{C}_1$. Realizers $\Theta, \Phi, \Psi, \ldots$ of formulas in $\mathcal{C}_1 \setminus \mathcal{POS}$ are *arbitrary* functions from $\mathbb{W}$ to $\mathbb{W}$. In the following definition, A and B denote formulas in the class $\mathcal{POS}$.

$$
\begin{array}{lll}
\Theta \;\mathbf{r}\; (A \to B) & \text{if} & \rho \;\mathbf{r}\; A \text{ entails } \Theta(\rho) \;\mathbf{r}\; B \text{ for all } \rho, \\
\Theta \;\mathbf{r}\; (\forall x)(A(x) \to B(x)) & \text{if} & \Theta \;\mathbf{r}\; (A(t) \to B(t)) \text{ for all terms } t.
\end{array}
$$

Similarly as above, if $\Gamma = A_1, \ldots, A_n$ denotes a sequence of formulas in $\mathcal{C}_1 \setminus \mathcal{POS}$ and $\vec{\Theta} = \Theta_1, \ldots, \Theta_n$, then we write $\vec{\Theta} \;\mathbf{r}\; \Gamma$ in order to express that $\Theta_i \;\mathbf{r}\; A_i$ for all $1 \leq i \leq n$.

[4] The only minor difference to [30] is the infinitary clause for $\forall$, which is inessential for positive formulas but necessary in the realizability of $\mathcal{C}_1$ formulas below.

Let us conclude our definition of the notion of realizability by observing that it preserves equality in $\mathcal{M}(\lambda\eta)$, i.e., if Θ and ρ realize $A(s)$ and $B(s)$, respectively, and $\mathcal{M}(\lambda\eta) \models s = t$, then also $A(t)$ and $B(t)$ are realized by Θ and ρ, respectively.

Let us introduce some final pieces of notation before we state the crucial realizability theorem for PT^+. For an $\mathcal{L}$ formula A we write $A[\vec{u}]$ in order to express that all the free variables occurring in A are contained in the list $\vec{u}$. The analogous convention is used for finite sequences of $\mathcal{L}$ formulas. Moreover, let Γ be a finite sequence of $\mathcal{C}_1$ formulas and assume that $A_{i_1}, \dots, A_{i_k}$ and $B_{j_1}, \dots, B_{j_l}$ are the unique subsequences of Γ so that A_{i_r} is in $\mathcal{C}_1 \setminus \mathcal{POS}$ and B_{j_s} is in $\mathcal{POS}$ for all $1 \leq s \leq k$ and $1 \leq r \leq l$. If $\vec{\Theta} = \Theta_1, \dots, \Theta_k$ and $\vec{\rho} = \rho_1, \dots, \rho_l$, then the notation $\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma$ simply expresses that $\vec{\Theta}\ \mathbf{r}\ A_{i_1}, \dots, A_{i_k}$ and $\vec{\rho}\ \mathbf{r}\ B_{j_1}, \dots, B_{j_l}$.

The following realizability theorem is an extension of the corresponding realizability theorem for PT^+ in Strahm [30]. There are, however, some subtle points in the proof to be taken care of, which could be handled in a more direct manner in [30]. In particular, bounding arguments using monotonicity have to be avoided in the context of the basic feasible functionals.

Theorem 4 (Extended realizability for PT^+) *Let Γ be a finite sequence of formulas in $\mathcal{C}_1$ and let Δ be a finite sequence of formulas in $\mathcal{POS}$, and assume that $\mathsf{PT}^+ \vdash_{\star} \Gamma[\vec{u}] \Rightarrow \Delta[\vec{u}]$. Then there exists a basic feasible functional F so that we have for all terms $\vec{s}$, and all $\vec{\Theta}$ and $\vec{\rho}$ of appropriate length:*

$$\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}] \quad \Longrightarrow \quad F(\vec{\Theta}, \vec{\rho})\ \mathbf{r}_{\vee}\ \Delta[\vec{s}].$$

Proof The claim is proved by induction on the length of quasi cut-free derivations of sequents consisting of $\mathcal{C}_1$ formulas on the left and $\mathcal{POS}$ formulas on the right. It is important that our realizing functions are invariant under substitutions of terms $\vec{s}$ for the free variables $\vec{u}$ in the sequent $\Gamma[\vec{u}] \Rightarrow \Delta[\vec{u}]$. This fact is always immediate and, therefore, in order to simplify notation, we sometimes suppress substitutions in our discussion below.

First of all, the treatment of all logical and non-logical axioms of PT^+ and the rules of inference for $\vee, \wedge, \exists, \forall$ as well as *cut* and structural rules is identical to the proof of Theorem 15 in Strahm [30], with the only difference that now the realizing $\textsc{Bff}'s$ in general have function arguments. Hence, we refer the reader to [30] for a detailed treatment of these axioms and rules.

In the following let us address the rule for introduction of $\rightarrow$ on the left hand side of a sequent. Assume that our last inference is of the form

$$\frac{\Gamma \Rightarrow A, \Delta \qquad \Gamma, B \Rightarrow \Delta}{\Gamma, A \rightarrow B \Rightarrow \Delta},$$

and that F_0 and F_1 are the two realizing functionals in BFF for the left and the right premise of this rule, respectively, given to us by the induction hypothesis. Then we define the realizing functional F for the conclusion of this rule by

$$F(\vec{\Theta}, \Psi, \vec{\rho}) \;=\; \begin{cases} \langle (F_0(\vec{\Theta}, \vec{\rho}))_0 - 1, (F_0(\vec{\Theta}, \vec{\rho}))_1 \rangle & \text{if } F_0(\vec{\Theta}, \vec{\rho})_0 \neq 1, \\ F_1(\vec{\Theta}, \vec{\rho}, \Psi(F_0(\vec{\Theta}, \vec{\rho})_1)) & \text{otherwise.} \end{cases}$$

Clearly, F realizes the conclusion of the rule, and, moreover, F is in BFF. Observe that we do not have to consider the rule for introduction of $\rightarrow$ on the right hand side, due the the special form of our sequents. In fact, the latter rule would not be realizable for obvious reasons.

Let us now turn to the treatment of the $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ notation induction rule on W. The corresponding analysis is similar to the one given in [30]. However, the context of the basic feasible functionals requires more elaboration on certain subtle points. For example, in [30] we have implicitly used the fact that each polynomial time computable function is majorized by a monotone polynomial, a fact which does not hold for the BFF's. In the following, let us describe the treatment of $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ induction on W in all detail. According to the four premises of this rule, we have quasi cut-free PT^+ derivations of the four sequents

$$\begin{aligned} &\Gamma,\, \mathsf{W}(u) \;\Rightarrow\; \mathsf{W}(tu),\, \Delta, \\ &\Gamma \;\Rightarrow\; A(\epsilon),\, \Delta, \\ &\Gamma,\, \mathsf{W}(u),\, A(u) \;\Rightarrow\; A(\mathsf{s}_i u),\, \Delta, \quad (i = 0, 1) \end{aligned}$$

for $A(u)$ being of the form $(\exists y \leq tu) B(u, y)$ with B positive and W free. Hence, the induction hypothesis guarantees the existence of four basic feasible functionals F, G_ϵ, G_0, and G_1, so that we have for all $\mathcal{L}$ terms $\vec{s}$ and all $\vec{\Theta}, \vec{\rho}, \sigma, \tau$,

(1) $\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}] \implies F(\vec{\Theta}, \vec{\rho}, \sigma)\ \mathbf{r}_\vee\ \mathsf{W}(t[\vec{s}](\sigma)),\, \Delta[\vec{s}]$,

(2) $\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}] \implies G_\epsilon(\vec{\Theta}, \vec{\rho})\ \mathbf{r}_\vee\ A[\vec{s}, \epsilon],\, \Delta[\vec{s}]$,

(3) $\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}],\ \tau\ \mathbf{r}\ A[\vec{s}, \sigma] \implies G_i(\vec{\Theta}, \vec{\rho}, \sigma, \tau)\ \mathbf{r}_\vee\ A[\vec{s}, \mathsf{s}_i\sigma],\, \Delta[\vec{s}]$ $(i = 0, 1)$

It is our aim to find a basic feasible realizing functional for the conclusion of the notation induction rule, i.e., a functional H in BFF so that we have for all $\vec{\Theta}, \vec{\rho}, \sigma$,

(4) $$\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}] \quad\implies\quad H(\vec{\Theta}, \vec{\rho}, \sigma)\ \mathbf{r}_\vee\ A[\vec{s}, \sigma],\, \Delta[\vec{s}].$$

The desired functional H is defined by recursion on notation on σ as follows:

$$H(\vec{\Theta}, \vec{\rho}, \epsilon) = G_\epsilon(\vec{\Theta}, \vec{\rho}),$$

$$H(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) = \begin{cases} H(\vec{\Theta}, \vec{\rho}, \sigma) & \text{if } H(\vec{\Theta}, \vec{\rho}, \sigma)_0 \neq 1, \\ F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) & \text{if } H(\vec{\Theta}, \vec{\rho}, \sigma)_0 = 1 \text{ and} \\ & F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_0 \neq 1, \\ G_i(\vec{\Theta}, \vec{\rho}, \sigma, H(\vec{\Theta}, \vec{\rho}, \sigma)_1) & \text{otherwise.} \end{cases}$$

It is now easy to verify (4) by (meta) notation induction on σ, using our assertions (1)–(3) from the induction hypothesis.

In order to show that H is indeed basic feasible, we have to exhibit a bounding functional K in BFF so that

$$H(\vec{\Theta}, \vec{\rho}, \sigma) \leq K(\vec{\Theta}, \vec{\rho}, \sigma) \tag{5}$$

for all $\vec{\Theta}, \vec{\rho}, \sigma$. Indeed, it is clearly enough to bound H under the assumption $\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}]$, and we will see that our bounding functional K does not depend on $\vec{s}$. As we have already mentioned above, in the sequel we must avoid the use of monotonicity arguments as they have been employed in [30].

We start our considerations concerning bounding by first defining an auxiliary functional $\widetilde{H}$, *which differs from H in the third case of the above case distinction only.* We will show that $\widetilde{H}$ is basic feasible and use this fact to rewrite H in such a way that an appropriate bounding functional for H will fall out at once. $\widetilde{H}$ is defined by recursion on notation in the following manner:

$$\widetilde{H}(\vec{\Theta}, \vec{\rho}, \epsilon) = G_\epsilon(\vec{\Theta}, \vec{\rho}),$$

$$\widetilde{H}(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) = \begin{cases} \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma) & \text{if } \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_0 \neq 1, \\ F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) & \text{if } \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_0 = 1 \text{ and} \\ & F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_0 \neq 1, \\ G_i(\vec{\Theta}, \vec{\rho}, \sigma, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_1) & \text{if } \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_0 = 1 \text{ and} \\ & F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_0 = 1 \text{ and} \\ & G_i(\vec{\Theta}, \vec{\rho}, \sigma, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_1)_0 = 1, \\ \langle 0, 0 \rangle & \text{otherwise.} \end{cases}$$

Observe that if the definition of $\widetilde{H}$ enters the last case of the above case distinction, then $\widetilde{H}(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)$ is set to $\langle 0, 0 \rangle$ and remains constant afterwards. Moreover, for all subwords τ of σ, $\widetilde{H}(\vec{\Theta}, \vec{\rho}, \tau)$ equals $H(\vec{\Theta}, \vec{\rho}, \tau)$ and, hence, property (4) above also holds for $\widetilde{H}$ instead of H for such τ's.

It is our aim now to find a bounding functional $\widetilde{K}$ for $\widetilde{H}$, again under the proviso $\vec{\Theta}, \vec{\rho}\ \mathbf{r}\ \Gamma[\vec{s}]$. The crucial case in bounding $\widetilde{H}$ is case three in the above case distinction. There $\widetilde{H}(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)$ is defined to be $G_i(\vec{\Theta}, \vec{\rho}, \sigma, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_1)$ under the assumptions

$$\text{(6)} \qquad \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_0 = 1, \quad G_i(\vec{\Theta}, \vec{\rho}, \sigma, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_1)_0 = 1, \quad F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_0 = 1.$$

These facts together with (1)–(3) and our discussion above readily entail the following two assertions:

$$\text{(7)} \qquad \widetilde{H}(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_1\ \mathbf{r}\ A[\vec{s}, \mathsf{s}_i\sigma] \quad \text{and} \quad F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_1\ \mathbf{r}\ \mathsf{W}(t[\vec{s}](\mathsf{s}_i\sigma)).$$

But now we have to recall that the formula $A[\vec{s}, \mathsf{s}_i\sigma]$ has the shape

$$(\exists y \in \mathsf{W})[y \leq t[\vec{s}](\mathsf{s}_i\sigma) \ \wedge\ B[\vec{s}, y, \mathsf{s}_i\sigma]],$$

with B positive and W free; hence, the only occurrence of W in $A[\vec{s}, \mathsf{s}_i\sigma]$ stems from the leading bounded existential quantifier. But the bounding term $t[\vec{s}](\mathsf{s}_i\sigma)$ of this quantifier evaluates to $F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_1$ in $\mathcal{M}(\lambda\eta)$ according to (7). It is now a matter of routine to find a basic feasible functional I[5] so that under our assumption (6), we have

$$\widetilde{H}(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) \ =\ G_i(\vec{\Theta}, \vec{\rho}, \sigma, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_1) \ \leq\ I(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma).$$

Hence, we were able to bound $\widetilde{H}$ in case we are in the third case of the above case distinction.

In order to find the final bounding functional $\widetilde{K}$ for $\widetilde{H}$, we first note that the basic feasible functionals are closed under the bounded maximum functional, cf. Cook and Kapron [10]. More precisely, if N is a BFF, then the functional M defined by

$$M(\vec{\Theta}, \vec{\rho}, \sigma) \ =\ \max_{\tau \subseteq \sigma} N(\vec{\Theta}, \vec{\rho}, \tau)$$

is in BFF, too, where max is understood with respect to the (tally) length of binary words. Hence, we can now spell out the desired bounding functional $\widetilde{K}$ for $\widetilde{H}$, where as above, $*$ denotes concatenation of binary words.

$$\widetilde{K}(\vec{\Theta}, \vec{\rho}, \sigma) \ =\ \max_{\tau \subseteq \sigma} \Big(F(\vec{\Theta}, \vec{\rho}, \tau) * I(\vec{\Theta}, \vec{\rho}, \tau) \Big) * \langle 0, 0 \rangle$$

Inspecting the definition of $\widetilde{H}$, one readily sees that $\widetilde{K}$ does its job. Thus, we have established that $\widetilde{H}$ is a basic feasible functional.

[5] The definition of I makes use of F and is defined according to the specific form of the $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ formula A; the precise definition is tedious but obvious, given our simple notion of realizability for positive formulas.

Finally, in the light of our discussion following the definition of $\widetilde{H}$, an easy (meta) inductive argument shows that the recursion equations for our main functional H defined above can be rewritten by means of $\widetilde{H}$ in the following manner:

$$
\begin{array}{rcl}
H(\vec{\Theta}, \vec{\rho}, \epsilon) & = & G_\epsilon(\vec{\Theta}, \vec{\rho}), \\
H(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) & = & \begin{cases}
H(\vec{\Theta}, \vec{\rho}, \sigma) & \text{if } H(\vec{\Theta}, \vec{\rho}, \sigma)_0 \neq 1, \\
F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma) & \text{if } H(\vec{\Theta}, \vec{\rho}, \sigma)_0 = 1 \text{ and} \\
 & F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_0 \neq 1, \\
G_i(\vec{\Theta}, \vec{\rho}, \sigma, H(\vec{\Theta}, \vec{\rho}, \sigma)_1) & \text{if } H(\vec{\Theta}, \vec{\rho}, \sigma)_0 = 1 \text{ and} \\
 & F(\vec{\Theta}, \vec{\rho}, \mathsf{s}_i\sigma)_0 = 1 \text{ and} \\
 & G_i(\vec{\Theta}, \vec{\rho}, \sigma, H(\vec{\Theta}, \vec{\rho}, \sigma)_1)_0 = 1, \\
G_i(\vec{\Theta}, \vec{\rho}, \sigma, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \sigma)_1) & \text{otherwise.}
\end{cases}
\end{array}
$$

Clearly, an adequate basic feasible bounding functional K for H satisfying (5) is given by the obvious definition

$$K(\vec{\Theta}, \vec{\rho}, \sigma) \;=\; \widetilde{K}(\vec{\Theta}, \vec{\rho}, \sigma) * \max_{\tau \subseteq \sigma} \max_{i=0,1} \Big(G_i(\vec{\Theta}, \vec{\rho}, \tau, \widetilde{H}(\vec{\Theta}, \vec{\rho}, \tau)_1) \Big)$$

Observe that the first three cases in the above rewriting of H are covered by the $\widetilde{K}$ functional, whereas the last case is taken care of by the second functional in the above definition of K. Hence, K is indeed a bounding functional for H.

This ends our proof that H is in BFF and, hence, the treatment of the $\Sigma^{\mathsf{b}}_{\mathsf{W}}$ notation induction rule. The proof of the extended realizability theorem is thus complete. $\Box$

Corollary 5 *Let t be a closed $\mathcal{L}$ term and assume that the sequent*

$$f_1 : \mathsf{W} \to \mathsf{W}, \ldots, f_k : \mathsf{W} \to \mathsf{W},\ x_1 \in \mathsf{W}, \ldots, x_l \in \mathsf{W} \;\Rightarrow\; t f_1 \ldots f_k x_1 \ldots x_l \in \mathsf{W}$$

is derivable in PT^+, for distinct variables $f_1, \ldots, f_k$ and $x_1, \ldots, x_l$. Then there exists a basic feasible functional F of rank (k, l) so that t defines F in the open term model $\mathcal{M}(\lambda\eta)$.

Proof Assuming that the above sequent in provable in PT^+, by Theorem 2 we know that it has derivation with positive cuts only. By the realizability theorem we obtain a basic feasible functional G of rank (k, l) so that we have for all terms $r_1 \ldots r_k, s_1 \ldots s_l$ and all $\Theta_1, \ldots, \Theta_k, \rho_1, \ldots, \rho_l$,

$$G(\Theta_1, \ldots, \Theta_k, \rho_1, \ldots, \rho_l)_1 \;\mathbf{r}\; \mathsf{W}(t r_1 \ldots r_k s_1 \ldots s_l),$$

provided that Θ_i $\mathbf{r}$ $r_i : \mathsf{W} \to \mathsf{W}$ and ρ_j $\mathbf{r}$ $\mathsf{W}(s_j)$ for $1 \leq i \leq k$ and $1 \leq j \leq l$. Hence, given in addition that r_i is in $\mathcal{W}^{\mathcal{W}}$[6] and $s_j = \overline{w}$ for some w in $\mathbb{W}$, the latter condition is readily satisfied by choosing $\Theta_i = \hat{r}_i$ and $\rho_j = w$. Thus, our desired basic feasible functional F of rank (k, l) is given by

$$F(\Theta_1, \ldots, \Theta_k, \rho_1, \ldots, \rho_l) = G(\Theta_1, \ldots, \Theta_k, \rho_1, \ldots, \rho_l)_1$$

for all $\Theta_1, \ldots, \Theta_k$ and $\rho_1, \ldots, \rho_l$. This ends the proof of the corollary. $\square$

Finally, together with Theorem 1, we have now proved the main result of this article.

Corollary 6 *The provably total type two functionals of* PT *coincide with the basic feasible functionals of type two. Moreover, this characterization remains true in the presence of totality of application* (Tot) *and extensionality of operations* (Ext).

6 Conclusion

In this article we have established that the provably total type two functionals of our classical applicative theory PT coincide with the basic feasible functionals of type two. This proof-theoretic characterization of the basic feasible functionals is hoped to provide further evidence for the naturalness and robustness of the class BFF_2.

In his PhD thesis [28], Anil Seth has used a version of Buss' S^1_2 augmented by function variables in order to give a proof-theoretic characterization of BFF_2 in the spirit of Buss' [6] delineation of the polynomial time computable functions. Despite of the importance of Seth's approach, we believe that the theory PT is somewhat more natural for studying notions of computability in higher types: indeed, as we have already argued, the finite types arise *directly* in PT, and it is not necessary to augment the language by new primitives as in the case of bounded arithmetic. Moreover, deriving BFF_2 in PT is coding free and much more pleasant than in corresponding systems of bounded arithmetic.

It follows from our embedding of PV^ω into PT in Strahm [30] that in fact the basic feasible functionals in *arbitrary finite types* are provably total in PT. Thus the question arises whether the converse also holds above type 2, i.e., whether each PT provably total functional of type greater than two is basic feasible. We strongly conjecture that the answer to this question is positive.

[6] As above, $\mathcal{W}^{\mathcal{W}}$ denotes the set of all terms s so that $\mathcal{M}(\lambda\eta) \models s : \mathsf{W} \to \mathsf{W}$.

Indeed, it is possible to adapt the modified realizability interpretation used in Section 9 of Cantini [7] in order to show that the provably total higher type functionals of an *intuitionistic* version of PT are basic feasible. However, it is not obvious how to reduce the classical theory PT to its intuitionistic version *so that statements expressing totality of arbitrary higher type functionals are preserved.*[7]

In [30] we have also introduced and analyzed the systems PS, PTLS, and LS which are related to polynomial space, *simultaneously* polynomial time and linear space, and linear space, respectively. Using the function algebra characterization of these complexity classes (cf. e.g. Theorem 1 in [30]) it is straightforward to come up with corresponding higher type systems which are patterned in the same manner as PV^ω. Moreover, the characterization result for the provably total type two functionals of PT directly carries over to PS, PTLS, and LS and the corresponding classes of type two functionals.

Last but not least, let us mention the important activities in the program of so-called implicit computational complexity and tiered formalisms in the sense of Bellantoni, Cook, and Leivant (cf. e.g. [4, 21, 23]). There questions regarding higher types have recently been of interest, see for example Leivant [22], Bellantoni, Niggl, Schwichtenberg [5], and Hofmann [18]. For applicative theories based on safe induction, see Cantini [8].

Recently and independently, Leivant [24] has given a proof-theoretic characterization of BFF in terms of second order logic with positive comprehension We will compare our approach with Leivant's elsewhere.

Acknowledgments.

We would like to thank the two anonymous referees for helpful comments on a earlier version of this paper.

[7] As far as the provably total functionals of type 2 are concerned, it seems that the forcing technique used in [7] can be used in order to reduce PT to a suitable extension of intuitionistic PT (cf. [7]) so that the type 2 content is preserved. Moreover, the provably total functionals of the latter extension can be shown to be basic feasible by combining techniques of [7] with the ideas used in the present paper in order to avoid the use of monotonicity. But as we have shown in our paper, the type two content of classical PT can be read off *directly*, without using this heavy detour. To conclude this side remark, we mention that the forcing interpretation of [7] does not seem to preserve totality assertions *above type 2*.

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] Beeson, M. J. Proving programs and programming proofs. In *Logic, Methodology and Philosophy of Science VII*, Barcan Marcus et. al., Ed. North Holland, Amsterdam, 1986, pp. 51–82.

[4] Bellantoni, S., and Cook, S. A new recursion-theoretic characterization of the poly-time functions. *Computational Complexity 2* (1992), 97–110.

[5] Bellantoni, S., Niggl, K.-H., and Schwichtenberg, H. Higher type recursion, ramification and polynomial time. *Annals of Pure and Applied Logic 104*, 1–3 (2000), 17–30.

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

[7] Cantini, A. Choice and uniformity in weak applicative theories. Invited talk, Logic Colloquium 2001, 31 pages. Submitted for publication.

[8] Cantini, A. Polytime, combinatory logic and positive safe induction. *Archive for Mathematical Logic 41*, 2 (2002), 169–189.

[9] Clote, P. Computation models and function algebras. In *Handbook of Computability Theory*, E. Griffor, Ed. Elsevier, 1999, pp. 589–681.

[10] 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.

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

[12] 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.

[13] 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.

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

[15] Ferreira, F. Polynomial time computable arithmetic. In *Logic and Computation, Proceedings of a Workshop held at Carnegie Mellon University, 1987*, W. Sieg, Ed., vol. 106 of *Contemporary Mathematics*. American Mathematical Society, Providence, Rhode Island, 1990, pp. 137–156.

[16] Girard, J.-Y. *Proof Theory and Logical Complexitiy.* Bibliopolis, Napoli, 1987.

[17] Hindley, J. R., and Seldin, J. P. *Introduction to Combinators and λ-Calculus.* Cambridge University Press, 1986.

[18] Hofmann, M. Type systems for polynomial-time computation. Habilitation Thesis, Darmstadt, 1999. Appeared as LFCS Technical Report ECS-LFCS-99-406.

[19] Irwin, R., Kapron, B., and Royer, J. On characterizations of the basic feasible functionals, Part I. *Journal of Functional Programming 11* (2001), 117–153.

[20] Kapron, B., and Cook, S. A new characterization of type 2 feasibility. *SIAM Journal on Computing 25* (1996), 117–132.

[21] Leivant, D. A foundational delineation of poly-time. *Information and Computation 110* (1994), 391–420.

[22] Leivant, D. Predicative recurrence in finite type. In *Logical Foundations of Computer Science*, A. Nerode and Y. Matiyasevich, Eds., vol. 813 of *Lecture Notes in Computer Science*. Springer, 1994, pp. 227–239.

[23] Leivant, D. Ramified recurrence and computational complexity I: Word recurrence and poly-time. In *Feasible Mathematics II*, P. Clote and J. Remmel, Eds. Birkhäuser, 1994, pp. 320–343.

[24] Leivant, D. Implicit computational complexity for higher type functionals (Extended abstract). In *CSL '02*, J. Bradfield, Ed., vol. 2471 of *Lecture Notes in Computer Science*. Springer, 2002, pp. 367–381.

[25] Melhorn, K. Polynomial and abstract subrecursive classes. *Journal of Computer and System Science 12* (1976), 147–178.

[26] Pezzoli, E. On the computational complexity of type 2 functionals. In *Computer Science Logic '97*, vol. 1414 of *Lecture Notes in Computer Science*. Springer, 1998, pp. 373–388.

[27] Royer, J. Semantics vs. syntax vs. computations: Machine models for type-2 polynomial-time bounded functionals. *Journal of Computer and System Science 54* (1997), 424–436.

[28] Seth, A. *Complexity Theory of Higher Type Functionals.* PhD thesis, Tata Institute of Fundamental Research, Bombay, 1994.

[29] Strahm, T. *Proof-theoretic Contributions to Explicit Mathematics.* Habilitationsschrift, University of Bern, 2001.

[30] Strahm, T. Theories with self-application and computational complexity. *Information and Computation 185* (2003), 263–297.

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

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