Third-order functionals on partial combinatory algebras

Jetze Zoethout

Department of Mathematics, Utrecht University

March 17, 2021

ABSTRACT

Computability relative to a partial function f on the natural numbers can be formalized using the notion of an oracle for this function f. This can be generalized to arbitrary partial combinatory algebras, yielding a notion of 'adjoining a partial function to a partial combinatory algebra A'. A similar construction is known for second-order functionals, but the third-order case is more difficult. In this paper, we prove several results for this third-order case. Given a third-order functional Φ on a partial combinatory algebra A, we show how to construct a partial combinatory algebra $A[\Phi]$ where Φ is 'computable', and which has a 'lax' factorization property (Theorem 7.3 below). Moreover, we show that, on the level of first-order functions, the effect of making a third-order functional computable can be described as adding an oracle for a first-order function.

1 Introduction

Classical computability on the natural numbers can be extended with notions of computability involving functions. For example, we may add an *oracle* for a non-computable partial function $f \colon \mathbb{N} \rightharpoonup \mathbb{N}$ to our Turing machines, yielding a notion of computability relative to f. Another example is given by Kleene's S1-S9, which devises a system of computation with *higher-order* functionals, whose inputs are also functionals, rather than just numbers.

This paper is concerned with computability on (higher-order) functions in the more general context of *partial combinatory algebras* (PCAs), which can be viewed as abstract 'models of computation'. Some work has previously been done on this subject. The paper [vO06] offers a notion of 'oracle computability' for general PCAs, and the paper [FvO16] generalizes this first paper to second-order functionals. The authors of [FvO16] also mention that the third-order case seems to be far more difficult. The main result of this paper concerns this third-order case. Specifically, given a third-order functional Φ on a PCA A, we show how to construct a PCA $A[\Phi]$ where Φ is 'computable', and which has a 'lax' factorization property (Theorem 7.3 below). Moreover, we show that, on the level of first-order functions, the effect of making a third-order functional computable can be described as adding an oracle for a first-order function.

As we said, this paper is concerned with computability on *functions*. In the paper [vO11], it is shown that, for each PCA A, there is another PCA $\mathcal{B}A$ whose elements are partial *functions* on A. The main strategy of this paper is to view higher-order functionals on A as *lower-order* functionals on $\mathcal{B}A$. In developing this strategy, the paper also serves as a unification of the material from [vO06] and [FvO16] on the one hand, and [vO11] on the other hand. However, there is also a serious obstacle for this strategy. It turns out that the elements of $\mathcal{B}A$ lead a kind of 'double life'. On the one hand, they are elements of $\mathcal{B}A$, but on the other hand, for each $\alpha \in \mathcal{B}A$, there are (other) elements from $\mathcal{B}A$ that *compute* α in a specific sense. These

cannot always be translated into one another, and we will give an explicit example of this phenomenon (Example 6.16).

Let us briefly outline the paper. First of all, in Section 2, we introduce partial combinatory algebras and the realizability toposes (denoted by RT) that can be constructed out out of them. In this paper, PCAs will always be *relative* and *ordered*; the main reason for considering such a general notion of PCAs is that it allows us to construct a geometric surjection $\mathsf{RT}(\mathcal{B}A) \twoheadrightarrow \mathsf{RT}(A)$. The relevant morphisms between PCAs are introduced in Section 3, which yields a preorder-enriched category of PCAs. Section 2 contains no new material, but Section 3 treats a slight innovation with respect to the literature, namely the notion of a *partial* applicative morphism, which is specific to *relative* PCAs. In Section 4, we revisit the paper [vO06], showing how to 'freely adjoin' a partial function $f\colon A \rightharpoonup A$ to a PCA A, yielding a new PCA $A[f]$. The construction is largely the same as in [vO06], but we have made it suitable for the relative ordered case. Next, Section 5 introduces the PCA $\mathcal{B}A$ of partial functions as in [vO11], but with one important deviation. One of the central views of this paper is that $\mathcal{B}A$ is best viewed as a relative, ordered PCA, even if A itself is an 'ordinary' PCA, i.e., lacking a notion of relativity and an order. In this section, we also show how, at the level of realizability toposes, $A[f]$ may be reconstructed using the construction $\mathcal{B}-$, slicing, and image toposes. This construction depends heavily on the fact that there is a geometric surjection $\mathsf{RT}(\mathcal{B}A) \twoheadrightarrow \mathsf{RT}(A)$, and therefore, on our treatment of $\mathcal{B}A$ as a relative, ordered PCA. Then, in Section 6, we treat the second-order case, where we reinterpret the construction from [FvO16] as a construction that actually takes place in $\mathcal{B}A$. We also give an explicit example of the 'obstacle' mentioned above (Example 6.16). Finally, in Section 7, we present the aforementioned results on the third-order case.

2 Relative ordered PCAs

A *partial combinatory algebra* is a non-empty set A equipped with a *partial* binary operation, called application. We think of the image of a pair (a, b) under this application map as the result, if defined, of applying the algorithm (with code or Gödel number) a to the input b. In order to capture this computational intuiton, this application will need to satisfy a few requirements, to be specified below. In the current setting, we add two extra features to this application map. First of all, we equip our PCAs with a partial order. We think of $a' \leq a$ as saying that a' gives more information than a, or that a' is a refinement of a. Second, we specify a *filter*, which is a subset of A satisfying certain properties. We think of the elements of this filter as the 'computable' elements, or as those (codes of) algorithms that can actually be carried out. Let us start by defining structures equipped with an application map and a partial order.

Definition 2.1. A *partial applicative poset* (abbreviated PAP) is a triple $A = (A, \cdot, \leq)$ where $(A, \leq)$ is a poset and $\cdot$ is a partial binary map $A \times A \rightharpoonup A$, called the *application map*, such that the following axiom is satisfied:

(A) the application map has downwards closed domain and preserves the order, i.e., if $a' \leq a$, $b' \leq b$ and $a \cdot b$ is defined, then $a' \cdot b'$ is defined as well, and $a' \cdot b' \leq a \cdot b$.

The PAP A is called *total* if the application map is total, and *discrete* if $\leq$ is the discrete order.

Axiom (A) fits the informal intuition about the order on A: if a' and b' contain at least as

much information as a resp. b, and $a \cdot b$ is already defined, then $a' \cdot b'$ should also be defined and contain at least as much information as $a \cdot b$.

Before we proceed to add the second extra feature mentioned above, let us agree on some notation and describe some basic constructions for PAPs. First of all, we will usually omit the dot for application, and just write ab for $a \cdot b$. Since the application map is not required to be associative, and in fact usually *will not* be associative, the bracketing of expressions is relevant. Here we adopt the convention, as is customary, that application associates to the left, meaning that abc is an abbreviation of $(ab)c$.

Second, since the application map is partial, we will have to deal with expressions that may or may not be defined. If e is a possibly undefined expression, then we write $e\downarrow$ to indicate that e is in fact defined. We take this to imply that all subexpressions of e are defined as well. If e and e' are possibly undefined expressions, then we write $e' \preceq e$ to mean: if $e\downarrow$, then $e'\downarrow$ as well, and $e' \leq e$. Observe that axiom (A) above may now be rewritten as: if $a' \leq a$ and $b' \leq b$, then $a'b' \preceq ab$. On the other hand, we write $e' \leq e$ to indicate that e' and e are in fact defined, and $e' \leq e$. Observe that, in the discrete case, $e' \preceq e$ reduces to *Kleene inequality*: if e is defined, then e' is also defined and denotes the same value.

Similarly, we write $e' \simeq e$ iff $e' \preceq e$ and $e \preceq e'$; in other words, $e\downarrow$ precisely when $e'\downarrow$, and in this case e and e' assume the same value. In other words, $\simeq$ is the familiar *Kleene equality*. On the other hand, $e' = e$ expresses the stronger statement that e' and e are in fact defined, and equal to each other.

The following object will become increasingly important in the remainder of the paper.

Definition 2.2. Let A be a PAP.

(i) We define $\mathcal{B}A$ as the set of all partial functions $\alpha \colon A \rightharpoonup A$ such that $a \leq b$ implies $\alpha(a) \preceq \alpha(b)$ for all $a, b \in A$.

(ii) For $\alpha, \beta \in \mathcal{B}A$, we say that $\alpha \leq \beta$ if $\alpha(a) \preceq \beta(a)$ for all $a \in A$.

In other words, $\alpha \in \mathcal{B}A$ if and only if its domain is downwards closed, and α is order-preserving on its domain. The statement $\alpha \leq \beta$ means that the domain of α extends the domain of β, and $\alpha \leq \beta$ holds pointwise on the domain of β. Clearly, this makes $\mathcal{B}A$ into a poset; later, we shall see that it can be equipped with a PCA structure. Observe that, in the discrete case, $\mathcal{B}A$ is simply the set of all partial functions on A, and the order is the *reverse* subfunction relation.

Warning 2.3. Some authors (including myself in other papers) discussing the discrete case use the 'opposite' convention for Kleene inequality, writing $e' \succeq e$ where we write $e' \preceq e$. The reason for doing so is that in this way, the corresponding order on partial functions is the actual subfunction relation, and not the reverse one. In the context of *ordered* PCAs, however, the current convention is the right one to adopt. Indeed, in the case where expressions are defined, one should like $e' \preceq e$ to imply $e' \leq e$, and not $e' \geq e$. Moreover, the order defined on $\mathcal{B}A$ matches our intuition about orders. Indeed, $\alpha \leq \beta$ means that α provides more information than β, either by specifying more values than β, or by adding information to values already specified by β.

Example 2.4. If $(A, \leq)$ is a poset with finite meets, then it can be made into a total PAP by setting $ab = a \wedge b$.

Example 2.5. Let $A = (A, \cdot, \leq)$ be a PAP. We write DA for the set of *downwards closed* subsets of A, i.e., the set of all $\alpha \subseteq A$ satisfying: if $a' \leq a$ and $a \in \alpha$, then $a' \in \alpha$. Clearly, DA

is partially ordered by inclusion. We make DA into a PAP by defining an application map as follows. If $\alpha, \beta \in DA$, then we say that $\alpha\beta{\downarrow}$ if and only if $ab{\downarrow}$ for all $a \in \alpha$ and $b \in \beta$. In this case, $\alpha\beta$ is defined as $\downarrow\{ab \mid a \in \alpha, b \in \beta\}$, i.e., the downwards closure of $\{ab \mid a \in \alpha, b \in \beta\}$. Restricting DA to the set TA of *non-empty* downwards closed subsets of A also yields a PAP.

The PAP DA will play an important role in the sequel of the paper. The following notation will be convenient when working with DA.

Definition 2.6. For $a \in A$ and $\alpha \in DA$, we write

$$a\alpha :\simeq \downarrow\{a\} \cdot \alpha \simeq \downarrow\{aa' \mid a' \in \alpha\},$$

and a similar definition applies for expressions of the form $a\alpha\beta$, etc.

We proceed to define filters.

Definition 2.7. Let $A = (A, \cdot, \leq)$ be a PAP. A *filter* of A is an non-empty subset $F \subseteq A$ that is:

(i) closed under application, i.e., if $a, b \in F$ and $ab{\downarrow}$, then also $ab \in F$;

(ii) upwards closed, i.e., if $a \leq b$ and $a \in F$, then also $b \in F$.

Example 2.8. If $(A, \leq)$ is a poset with finite meets, then a filter on $(A, \wedge, \leq)$ is a filter in the usual order-theoretic sense.

Example 2.9. (i) If A is a PAP, and F is a filter on A, then F can also be made into a PAP, by restricting both the application map and the order to F. This new PAP will be denoted by $(F, \cdot, \leq)$, or simply by F.

(ii) If A is a PAP, F is a filter on A, and G is a filter on the PAP F, then G is also a filter on A.

Since a filter is defined as a non-empty set with certain closure properties, we can consider the notion of a *generated filter*.

Definition 2.10. Let A be a PAP and let X be a non-empty subset of A. We define $\langle X \rangle$ as the smallest filter on A extending X, and we call this the filter *generated* by X.

In the case of filters on meet-semilattices, one can always generate a filter by first taking all finite meets, and then closing upwards. In the current case, a similar description is available. Befor we can formulate it, we need the notion of a *term*, which will also be central to the definition of PCAs later in this section.

Definition 2.11. Let A be a PAP. The set of *terms* over A is defined recursively as follows:

(i) We assume given a countably infinite set of disinct variables, and these are all terms.

(ii) For every $a \in A$, we assume that we have a *constant symbol* for a, and this is a term. The constant symbol for a is simply denoted by a.

(iii) If t_0 and t_1 are terms, then so is $(t_0 \cdot t_1)$ (but we usually just write $(t_0 t_1)$, and we omit brackets according to our convention).

If $t = t(\vec{x})$ is a term whose variables are among the sequence $\vec{x}$, then we can assign an obvious, possibly undefined, interpretation $t(\vec{a}) \in A$ to an input sequence $\vec{a} \in A$. In this way, every term $t(\vec{x})$ yields a partial function $\lambda\vec{a}.t(\vec{a})\colon A^n \rightharpoonup A$, where n is the length of the sequence $\vec{x}$. We have the following alternative descriptions of generated filters; the proof is easy and omitted.

Lemma 2.12. *Let A be a PAP and $X \subseteq A$ be non-empty. Then*

$$\langle X \rangle = \uparrow\{t(\vec{a}) \mid t(\vec{x}) \text{ a constant-free term}, \vec{a} \in X \text{ and } t(\vec{a})\!\downarrow\},$$

where $\uparrow$ stands for taking the upwards closure.

As promised, we will consider partial applicative preorders equipped with a filter.

Definition 2.13. A *partial applicative structure* (abbreviated PAS) is a quadruple $A = (A, A^{\#}, \cdot, \leq)$, where $(A, \cdot, \leq)$ is a PAP, and $A^{\#}$ is a filter on $(A, \cdot, \leq)$. So explicitly, $A^{\#}$ is a non-empty subset of A satisfying the following axioms:

(B) $A^{\#}$ is closed under application;

(C) $A^{\#}$ is upwards closed.

The PAS A is called *absolute* if $A^{\#} = A$. Moreover, a *filter* on A is a filter F on $(A, \cdot, \leq)$ such that $A^{\#} \subseteq F$.

Example 2.14. Let A be a PAS and let F be a filter on A. Then $(F, A^{\#}, \cdot, \leq)$ is also a PAS. When no confusion can arise, we will denote this PAS simply by F. Of course, $(A, F, \cdot, \leq)$ is also a PAS.

Example 2.15. If F is a filter on the PAP A, then $\{\alpha \in DA \mid \alpha \cap F \neq \emptyset\}$ is a filter on the PAP DA. In particular, if A is a PAS, then we can make DA into a PAS as well by setting $(DA)^{\#} = \{\alpha \in DA \mid \alpha \cap A^{\#} \neq \emptyset\}$. Similar remarks hold for TA.

Thus far, all the structure we have introduced is of an algebraic nature, and does not yet express a notion of computability. Now let us finally introduce the 'computational' component of PCAs.

Definition 2.16. A *partial combinatory algebra* (abbreviated PCA) is a PAS A for which there exist $\mathsf{k}, \mathsf{s} \in A^{\#}$ such that:

(D) $\mathsf{k}ab \leq a$;

(E) $\mathsf{s}ab\!\downarrow$;

(F) $\mathsf{s}abc \preceq ac(bc)$,

for all $a, b, c \in A$.

The elements k and s are usually called *combinators*. Using these combinators, every computation using the application map can be represented by a computable element (i.e., algorithm) from A itself. In order to make this statement precise, we use the terms introduced in Definition 2.11. As we mentioned, every term defines a partial function $A^n \rightharpoonup A$. The key fact about PCAs is that such partial functions are (laxly) computable using an element from A itself.

Proposition 2.17 (Combinatory completeness)**.** *Let A be a PCA. There exists a map that assigns to each term[1] $t = t(\vec{x}, y)$ with at least one variable, an element $\lambda^* \vec{x}, y.t \in A$, satisfying:*

(i) $(\lambda^* \vec{x}, y.t)\vec{a}\downarrow$ (where $\vec{a}$ has the same length as $\vec{x}$);

(ii) $(\lambda^* \vec{x}, y.t)\vec{a}b \preceq t(\vec{a}, b)$;

(iii) if all the constants occurring in t are from $A^{\#}$, then $\lambda^* \vec{x}, y.t \in A^{\#}$ as well.

Proof. Define the element $i \in A^{\#}$ as skk. We will give a slightly more general construction than required for the proposition. For a variable u and a term s, we define a new term $\lambda^* u.s$ with the following properties:

- the free variables of $\lambda^* u.s$ are those of s minus u;

- if $\vec{v}$ are the free variables of $\lambda^* u.s$, then the substitution instance $(\lambda^* u.s)[\vec{b}/\vec{v}]$ is defined for all $\vec{b} \in A$;

- moreover, if $a \in A$, then $(\lambda^* u.s)[\vec{b}/\vec{v}] \cdot a \preceq s[\vec{b}/\vec{v}, a/u]$;

- if all the constants occuring in s are from $A^{\#}$, then the same holds for $\lambda^* u.s$.

We define this new term resursively:

- If s is a constant or a variable distinct from u, then $\lambda^* u.s$ is $\mathsf{k}s$.

- If s is the variable u, then $\lambda^* u.s$ is i.

- If s is $s_0 s_1$, then $\lambda^* u.s$ is $\mathsf{s}(\lambda^* u.s_0)(\lambda^* u.s_1)$.

We leave the verification of the stated properties to the reader.

Now, if $\vec{x} = x_0, \ldots, x_{n-1}$, then we define $\lambda^* \vec{x}, y.t$ as (the interpretation of) the closed term

$$\lambda^* x_0.(\cdots (\lambda^* x_{n-1}.(\lambda^* y.t)) \cdots).$$

The verification of the properties (i), (ii) and (iii) is also left to the reader. Details may also be found in Chapter 1 of [vO08] (which treats the discrete, absolute case, but this can easily be generalized to our case). $\square$

Some useful combinators besides k and s are $i = \mathsf{skk}$ defined above, $\overline{\mathsf{k}} = \mathsf{ki}$, $\mathsf{p} = \lambda^* xyz.zxy$, $\mathsf{p}_0 = \lambda^* x.x\mathsf{k}$ and $\mathsf{p}_1 = \lambda^* x.x\overline{\mathsf{k}}$. Observe that these combinators all belong to $A^{\#}$ and satisfy:

$$ia \leq a, \quad \overline{\mathsf{k}}ab \leq b, \quad \mathsf{p}_0(\mathsf{p}ab) \leq a \quad \text{and} \quad \mathsf{p}_1(\mathsf{p}ab) \leq b.$$

In particular, $\mathsf{p}ab$ is always defined, and we think of this element as (coding) the *pair* (a, b). Accordingly, p is called the pairing combinator, and p_0 and p_1 are known as the unpairing combinators.

Moreover, we can construct *booleans*, i.e. elements $\top, \bot \in A^{\#}$ for which there is a *case operator* $\mathsf{C} \in A^{\#}$ satisfying $\mathsf{C}\top ab \leq a$ and $\mathsf{C}\bot ab \leq b$. Indeed, we may simply take $\top = \mathsf{k}$, $\bot = \overline{\mathsf{k}}$ and $\mathsf{C} = i$. When we are dealing with expressions that are possibly undefined, we need to be a bit more careful. Suppose we have terms $t_0(\vec{x})$, $t_1(\vec{x})$ and $t_2(\vec{x})$, and define the new

---

[1] Strictly speaking: a term along with an ordered sequence of distinct variables containing the variables from the term. A more formal treatment could be given using terms-in-context, but we will not take this trouble here.

term $t := \mathsf{C}t_0t_1t_2$, whose free variables are also among $\vec{x}$. Then this term does not behave as one would expect at first glance. In particular, if $t_0(\vec{a}) \leq \top$ and $t_1(\vec{a})\downarrow$, then it does *not* follow that $t(\vec{a})$ is defined. Indeed, it may happen that $t_2(\vec{a})$ fails to be defined and, since t_2 is a subterm of t, this prevents $t(\vec{a})$ from being defined. We clearly do not want this, since we are not interested in the value (if any) of $t_2(\vec{a})$ when $t_0(\vec{a}) \leq \top$. Therefore, we introduce a *strong case distinction* (we take this terminology from [LN15], Section 3.3.3.). If $t_0(\vec{x})$, $t_1(\vec{x})$ and $t_2(\vec{x})$ are terms, then we define a new term $t'(\vec{x})$ as:

$$\mathsf{C}t_0(\lambda^* y.t_1)(\lambda^* y.t_2)\mathsf{i}.$$

where y is not among the $\vec{x}$. One can easily check that this term has the following property: if $t_0(\vec{a}) \leq \top$, then $t'(\vec{a}) \preceq t_1(\vec{a})$, whereas if $t_0(\vec{a}) \leq \bot$, then $t'(\vec{a}) \preceq t_2(\vec{a})$. We will denote the term t' above by if t_0 then t_1 else t_2. Observe that, if all parameters from t_0, t_1 and t_2 are in $A^{\#}$, then the same holds for if t_0 then t_1 else t_2.

As in ordinary recursion theory, we have *fixpoint operators*. Using the terminology from [LN15], Section 3.3.5, we have a fixpoint operator $\mathsf{y} \in A^{\#}$ and a guarded fixpoint operator $\mathsf{z} \in A^{\#}$ satisfying: $\mathsf{y}a \preceq a(\mathsf{y}a)$, $\mathsf{z}a\downarrow$ and $\mathsf{z}ab \preceq a(\mathsf{z}a)b$. These may be constructed as $\mathsf{y} = uu$ where $u = \lambda^* xy.\mathsf{y}(xxy)$ and $\mathsf{z} = vv$ where $v = \lambda^* xyz.\mathsf{y}(xxy)z$. The fixpoint operator y is generally only useful in total PCAs, since $\mathsf{y}a \preceq a(\mathsf{y}a)$ will always be true if $\mathsf{y}a$ is not defined. The guarded fixpoint operator z has the property that $\mathsf{z}a$ is always defined, and can be used to create self-referential definitions. Explicitly, if $t(x, y)$ is a term, then setting $a := \lambda^* xy.t$ yields an element $T := \mathsf{z}a$ with the property that $Tb \preceq aTb \preceq t(T, b)$ for all $b \in A$. Moreover, if all the parameters from t are in $A^{\#}$, then $T \in A^{\#}$ as well. Obviously, this construction can be generalized to more variables, either by adjusting the definition of z or by using the pairing combinators.

All this justifies the view of PCAs as generalizing computability on the natural numbers. In fact, we can code the natural numbers in a PCA A, by setting recursively $\overline{0} = \mathsf{i}$ and $\overline{n+1} = \mathsf{p}\bot\overline{n}$; observe that all the $\overline{n}$ are in $A^{\#}$. Usually, we will omit the bar and simply write $n \in A^{\#}$, where we really mean its representative $\overline{n}$. Now it is easily checked that the elements $\mathsf{zero} = \mathsf{p}_0$, $\mathsf{suc} = \lambda^* x.\mathsf{p}\bot x$ and $\mathsf{pred} = \lambda^* x.\mathsf{p}_0 x\mathsf{i}(\mathsf{p}_1 x)$ from $A^{\#}$ satisfy: $\mathsf{zero} \cdot 0 \leq \top$, $\mathsf{zero} \cdot (n+1) \leq \bot$, $\mathsf{suc} \cdot n \leq n+1$, $\mathsf{pred} \cdot 0 \leq 0$ and $\mathsf{pred} \cdot (n+1) \leq n$. Moreover, using the guarded fixpoint operator, we may construct a recursor $\mathsf{rec} \in A^{\#}$ such that

$$\mathsf{rec}ab0 \leq a \quad \text{and} \quad \mathsf{rec}ab(n+1) \preceq bn(\mathsf{rec}abn) \text{ for all } n \in \mathbb{N} \text{ and } a, b \in A.$$

Since we have a 'pairing' function given by $\mathsf{p} \in A^{\#}$, we can also code longer tuples in A. More precisely, we can define total functions $j^n \colon A^n \to A$ for $n \geq 0$ by:

- $j^0() = \mathsf{i}$;

- $j^{n+1}(a_0, \ldots, a_n) = \mathsf{p}a_0 \cdot j^n(a_1, \ldots, a_n)$.

Using these functions, we can devise a *coding of finite sequences in A*. If $a_0, \ldots, a_{n-1}$ is a sequence, then we define its code by:

$$[a_0, \ldots, a_{n-1}] := \mathsf{p}n \cdot j^n(a_0, \ldots, a_{n-1}).$$

Observe that $[a_0, \ldots, a_{n-1}]$ is built using the a_i, combinators from $A^{\#}$ and application. In particular, if all the a_i are from $A^{\#}$, then so is the code $[a_0, \ldots, a_{n-1}]$.

Using the combinators above, one can mimick the standard recursion theoretic arguments to show that all elementary operations on sequences are computable in terms of their codes. The following definition introduces a few such computations that we will need in the sequel.

Definition 2.18. If A is a PCA, then $\mathsf{lh}, \mathsf{read}, \mathsf{fst}, \mathsf{concat}, \mathsf{unit}, \mathsf{ext} \in A^{\#}$ are combinators that satisfy:

- $\mathsf{lh} \cdot [a_0, \ldots, a_{n-1}] \leq n$;

- $\mathsf{read} \cdot [a_0, \ldots, a_{n-1}] \cdot i \leq a_i$ if $i < n$;

- $\mathsf{fst} \cdot [a_0, \ldots, a_n] \leq a_0$;

- $\mathsf{concat} \cdot [a_0, \ldots, a_{n-1}] \cdot [b_0, \ldots, b_{m-1}] \leq [a_0, \ldots, a_{n-1}, b_0, \ldots, b_{m-1}]$;

- $\mathsf{unit} \cdot a \leq [a]$;

- $\mathsf{ext} \cdot [a_0, \ldots, a_{n-1}] \cdot a' \leq [a_0, \ldots, a_{n-1}, a']$.

Remark 2.19. Of course, the combinators constructed up to this point are far from unique. But all of them may be constructed using only the elements k and s. When working with a PCA, we will assume that we have made an explicit choice for k and s, and as a result, a choice for all the combinators mentioned above.

Example 2.20. If A is a PCA and F is a filter on A, then $(A, F, \cdot, \leq)$ and $(F, A^{\#}, \cdot, \leq)$ are also PCAs, as can be seen by taking the same combinators k and s.
The following instance of this example will be relevant in the coming sections. If r is an element of A, then we define $F_r = \langle A^{\#} \cup \{r\} \rangle$, i.e., F_r is the least filter on A containing r. We denote the PCA $(A, F_r, \cdot, \leq)$ by $A[r]$.

Example 2.21. If A is a PCA, then so are DA and TA. In both cases, a suitable choice of combinators is $\downarrow\{\mathsf{k}\}, \downarrow\{\mathsf{s}\}$.

Example 2.22. The prototypical example of a (discrete, absolute) PCA is *Kleene's first model* $\mathcal{K}_1$. Its underlying set is $\mathbb{N}$, and $m \cdot n$ is the result, if any, of applying the mth partial recursive function to n.

Definition 2.23. A PCA A is called *semitrivial* if $\top, \bot \in A$ have a common lower bound.

This notion is introduced for the following reason: many constructions in this paper use case distinctions inside a PCA A. Usually, such constructions do not work in a semitrivial PCA, since we cannot distinguish $\top$ and $\bot$. Observe that, if u is a common lower bound of $\top$ and $\bot$, then uab is a common lower bound of a and b, for any $a, b \in A$. So in a semitrivial PCA, every two elements have a common lower bound. On the other hand, if A is not semitrivial, then the numerals for any two distinct $m, n \in \mathbb{N}$ do not have a common lower bound. In particular, every non-semitrivial PCA is infinite.
In the next section, we will introduce the relevant morphisms between PCAs. Before we move to this section, we briefly describe two important categorical constructions on PCAs.

Definition 2.24. Let A be a PCA.

(i) An *assembly* over A is a pair $X = (|X|, E_X)$, where $|X|$ is a set and E_X is a function $|X| \to TA$, i.e., $E_X(x)$ is a non-empty downwards closed subset of A, for all $x \in |X|$.

(ii) A *morphism of assemblies* $X \to Y$ is a function $f : |X| \to |Y|$ for which there exists a $t \in A^{\#}$ such that: for all $x \in |X|$, the set $t \cdot E_X(x)$ (as in Definition 2.6) is defined and a subset of $E_Y(f(x))$. Such a t is called a *tracker* for f.

8

Proposition 2.25. *Assemblies over a PCA A and morphisms between them form a category* $\mathsf{Asm}(A)$, *and this is a quasitopos.*

The proof is a straightforward generalization of the material in Section 1.5 from [vO08], and is omitted. There is an obvious forgetful functor $\Gamma \colon \mathsf{Asm}(A) \to \mathsf{Set}$ sending X to $|X|$ and which is the identity on arrows. In the other direction, there is a functor $\nabla \colon \mathsf{Set} \to \mathsf{Asm}(A)$ given by $\nabla Y = (Y, \lambda y.A)$ and $\nabla f = f$. The functors Γ and ∇ are both regular, and we have $\Gamma \dashv \nabla$ with $\Gamma \nabla \cong \mathrm{id}_{\mathsf{Set}}$.

Even though $\mathsf{Asm}(A)$, being a quasitopos, enjoys nice properties, it is not an exact category. We can make it exact by taking the ex/reg completion, which turns out to be an elementary topos.

Definition 2.26. Let A be a PCA. The *realizability topos* $\mathsf{RT}(A)$ is defined as $\mathsf{Asm}(A)_{\mathrm{ex/reg}}$.

Since $\Gamma \colon \mathsf{Asm}(A) \to \mathsf{Set}$ is regular and Set is exact, this functor may be lifted to a functor $\mathsf{RT}(A) \to \mathsf{Set}$, which will also be denoted by Γ. In the other direction, we have the composition $\mathsf{Set} \xrightarrow{\nabla} \mathsf{Asm}(A) \hookrightarrow \mathsf{RT}(A)$, which will also be denoted by ∇. This yields a geometric inclusion $\Gamma \dashv \nabla \colon \mathsf{Set} \to \mathsf{RT}(A)$, which is equivalent to the inclusion of $\neg\neg$-sheaves of $\mathsf{RT}(A)$.

3 Partial applicative morphisms

In this section, we introduce morphisms between PCAs. Usually, a morphism from a PCA A to a PCA B is a function that assigns to each $a \in A$ a non-empty subset of B. In the ordered setting, this needs to be amended to: a non-empty *downset* of B. The non-emptiness condition is needed to make sure that every computation from A can be 'transferred' along the morphism to B. However, in the relative setting, computations in A are given by elements from $A^{\#}$, rather than A. So one really needs to require the following: for each $a \in A^{\#}$, the associated downset of B contains an element of $B^{\#}$. For a outside $A^{\#}$, the non-emptiness conditions can be omitted, leading to the notion of a *partial applicative morphism*.

Definition 3.1. Let A and B be PCAs.

(i) A *partial applicative morphism* $A \rightharpoonup B$ is a function $f \colon A \to DB$ satisfying the following three requirements:

1. $f(a) \cap B^{\#} \neq \emptyset$ for all $a \in A^{\#}$.
2. There exists a $t \in B^{\#}$ such that: for all $a, a' \in A$, if aa' is defined, then $t \cdot f(a) \cdot f(a')$ is defined as well, and a subset of $f(aa')$. Such a t is called a *tracker* for f.
3. There exists a $u \in B^{\#}$ such that: $u \cdot f(a)$ is defined for all $a \in A$, and $u \cdot f(a) \subseteq f(a')$ whenever $a \leq a'$. We say that f *preserves the order up to* u.

We say that f is *total* if $f(a) \neq \emptyset$ for all $a \in A$, and in this case, we write $f \colon A \to B$.

(ii) If $f, f' \colon A \to DB$ are functions, then we say that $f \leq f'$ if there exists an $s \in B^{\#}$ such that: for all $a \in A$, the set $s \cdot f(a)$ is defined and a subset of $f'(a)$. Such an s is said to *realize* the inequality $f \leq f'$. Moreover, we write $f \simeq f'$ if both $f \leq f'$ and $f' \leq f$.

Proposition 3.2. *PCAs, partial applicative morphisms and inequalities between them form a preorder-enriched category* pPCA. *Restricting the 1-cells to* total *applicative morphisms yields another preorder-enriched category* PCA.

Proof. The identity on a PCA A is given by $\mathrm{id}_A(a) = \,\downarrow\!\{a\}$. Moreover, if $A \xrightarrow{f} B \xrightarrow{g} C$, then their composition gf is defined by $gf(a) = \bigcup_{b \in f(a)} g(b)$. The proof that this yields a preorder-enriched category proceeds as in Section 1.5 of [vO08], with appropriate adjustments. We do mention that pPCA is not a *strict* preorder-enriched category, since $\mathrm{id}_A\, f \simeq f$ in general holds only up to isomorphism (see also Lemma 3.6 below). The other equations for a category do hold strictly.

For the second statement, observe that identities are total, and that total morphisms are closed under composition. $\qquad\square$

The reason why we call our applicative morphisms 'partial' is that we view the fact that $f(a)$ is non-empty as evidence that $f(a)$ is 'actually defined'. Moreover, an element of $f(a) \cap B^{\#}$ counts as *effective* evidence that $f(a)$ is defined. This motivates the following definition.

Definition 3.3. Let $f \colon A \rightharpoonup B$ be a partial applicative morphism. Then we write

$$\mathrm{dom}\, f = \{a \in A \mid f(a) \neq \emptyset\} \quad \text{and} \quad \mathrm{dom}^{\#} f = \{a \in A \mid f(a) \cap B^{\#} \neq \emptyset\}.$$

We see that f is total precisely when $\mathrm{dom}\, f = A$. Since $f \leq f'$ implies that $\mathrm{dom}\, f \subseteq \mathrm{dom}\, f'$, this also means that $\mathsf{PCA}(A, B)$ is upwards closed inside $\mathsf{pPCA}(A, B)$. Moreover, we have the following result.

Lemma 3.4. *Let $f \colon A \rightharpoonup B$ be a partial applicative morphisms. Then $\mathrm{dom}\, f$ and $\mathrm{dom}^{\#} f$ are filters of A.*

Proof. By property 1 of f, we have $A^{\#} \subseteq \mathrm{dom}^{\#} f \subseteq \mathrm{dom}\, f$. If $t \in B^{\#}$ tracks f and $a, a' \in A$ are such that $aa' \downarrow$, then $t \cdot f(a) \cdot f(a') \subseteq f(aa')$. So, if $b \in f(a)$ and $b' \in f(a')$, then $tbb' \in f(aa')$, which shows that $\mathrm{dom}\, f$ is closed under application. Similarly, if $b \in f(a) \cap B^{\#}$ and $b' \in f(a') \cap B^{\#}$, then $tbb' \in f(aa') \cap B^{\#}$, so $\mathrm{dom}^{\#} f$ is closed under application as well. The proof that $\mathrm{dom}\, f$ and $\mathrm{dom}^{\#} f$ are upwards closed proceeds similarly, using property 3 of f. $\qquad\square$

Example 3.5. Let A be a PCA, let $r \in A$ and recall the PCA $A[r]$ from Example 2.20. Then there exists a total applicative morphism $\iota_r \colon A \to A[r]$ that 'acts as the identity', i.e., $\iota_r(a) = \,\downarrow\!\{a\}$. This morphism has the following universal property: if $f \colon A \rightharpoonup B$, then f factors, up to isomorphism, through ι_r if and only if $r \in \mathrm{dom}^{\#} f$. Indeed, f is also a partial applicative morphism $A[r] \rightharpoonup B$ if and only if $F_r \subseteq \mathrm{dom}^{\#} f$; and since $\mathrm{dom}^{\#} f$ is a filter on A, this is true if and only if $r \in \mathrm{dom}^{\#} f$.

When working with partial applicative morphisms, the following lemma ([HvO03], Lemma 3.3) is often useful. It says that every partial applicative morphism is isomorphic to one that preserves the order 'on the nose'.

Lemma 3.6. *Every partial applicative morphism $f \colon A \rightharpoonup B$ is isomorphic to an order-preserving partial applicative morphism f', meaning that $a \leq a'$ implies $f'(a) \subseteq f'(a')$.*

Proof. Define f' as $f\, \mathrm{id}_A$, i.e., $f'(a) = \bigcup_{a' \leq a} f(a')$ for $a \in A$. Proposition 3.2 tells us that f' is a partial applicative morphism which is isomorphic to f. Moreover, it is immediately clear that f' preserves the order on the nose. $\qquad\square$

Remark 3.7. Ordered PCAs were introduced in the paper [HvO03]. Here, the authors consider a category whose objects are ordered (but absolute) PCAs, but whose arrows are *functions* $A \to B$. They show that the assignment $A \mapsto TA$ is part of a monad structure on

this category, whose Kleisli category is (the absolute version of) PCA. For pPCA, a similar treatment can be given using the construction $A \mapsto DA$ instead. In the absolute setting, the PCA DA is not very interesting. Indeed, because DA has a least element $\emptyset$, it is equivalent to the one-element PCA. In the relative setting, on the other hand, this is no longer true, due to the fact that the least element $\emptyset$ of DA is never in $(DA)^{\#}$.

We now proceed to extend the constructions Asm and RT to partial applicative morphisms.

Definition 3.8. Let $f\colon A \rightharpoonup B$ be a partial applicative morphism. We define the functor $\mathsf{Asm}(f)\colon \mathsf{Asm}(A) \to \mathsf{Asm}(B)$ by:

- $|\mathsf{Asm}(f)(X)| = \{x \in |X| \mid E_X(x) \cap \operatorname{dom} f \neq \emptyset\}$;

- $E_{\mathsf{Asm}(f)(X)}(x) = \bigcup_{a \in E_X(x)} f(a)$ for $x \in |\mathsf{Asm}(f)(X)|$;

- $\mathsf{Asm}(f)(g)$ is the restriction of g to $|\mathsf{Asm}(f)(X)|$,

for assemblies $X \in \mathsf{Asm}(A)$ and arrows $g\colon X \to Y$ of $\mathsf{Asm}(A)$.

Since we are working with multiple assemblies simultaneously, we will sometimes write $\Gamma_A \dashv \nabla_A$ for the adjunction between Set and $\mathsf{Asm}(A)$ or $\mathsf{RT}(A)$. A functor $F\colon \mathsf{Asm}(A) \to \mathsf{Asm}(B)$ is called a Γ-functor if $\Gamma_B F \cong \Gamma_A$, and similarly, a ∇-functor if $F \nabla_A \cong \nabla_B$. For functors $\mathsf{RT}(A) \to \mathsf{RT}(B)$, we adopt a similar definition. Now we can state some elementary properties of the functor $\mathsf{Asm}(f)$.

Proposition 3.9. *For every partial applicative morphism f, the functor $\mathsf{Asm}(f)$ is a regular ∇-functor. Moreover, it is a Γ-functor precisely when f is total.*

By the universal property of the ex/reg completion, there is a unique regular functor $\mathsf{RT}(A) \to \mathsf{RT}(B)$ whose restriction to assemblies is $\mathsf{Asm}(f)$. We will denote this functor by $\mathsf{RT}(f)$; this is also a regular ∇-functor. Moreover, $\mathsf{RT}(f)$ is a Γ-functor iff $\mathsf{Asm}(f)$ is a Γ-functor, iff f is total.

Remark 3.10. Longley has shown ([Lon94], Sections 2.2 and 2.3), for the discrete, absolute case, that a left exact functor $F\colon \mathsf{Asm}(A) \to \mathsf{Asm}(B)$ is a ∇-functor if and only if it is a Γ-functor. Morover, the regular ∇-functors are (up to equivalence) precisely the functors of the form $\mathsf{Asm}(f)$. This can be generalized to the ordered case, but the relative case is slightly different. Indeed, while Γ-functors are still always ∇-functors, the converse is not true. In the relative case, the regular ∇-functors $\mathsf{Asm}(A) \to \mathsf{Asm}(B)$ correspond to the partial applicative morphisms $A \rightharpoonup B$, while the regular Γ-functors $\mathsf{Asm}(A) \to \mathsf{Asm}(B)$ correspond to *total* applicative morphisms $A \to B$.

In order to complete the picture, we remark that, if $f' \leq f$, we get a natural transformation $\mu\colon \mathsf{Asm}(f) \Rightarrow \mathsf{Asm}(f')$, where μ_X is the inclusion $|\mathsf{Asm}(f)(X)| \subseteq |\mathsf{Asm}(f')(X)|$ for assemblies X. By the universal property of the ex/reg completion, we also get a natural transformation $\mathsf{RT}(f) \Rightarrow \mathsf{RT}(f')$. This makes Asm and RT into pseudofunctors from pPCA to the 2-category of categories.

Now we discuss some special properties that a partial applicative morphism may have.

Definition 3.11. Let $f\colon A \rightharpoonup B$ be a partial applicative morphism. Then f is called:

(i) *decidable* if there is a $d \in B^{\#}$ (called a *decider* for f) such that $d \cdot f(\top) \subseteq {\downarrow}\{\top\}$ and $d \cdot f(\bot) \subseteq {\downarrow}\{\bot\}$;

(ii) *computationally dense* (abbreviated c.d.) is there is an $n \in B^{\#}$ such that:

$$\forall s \in B^{\#} \exists r \in A^{\#} (n \cdot f(r) \subseteq \downarrow\{s\}); \tag{cd}$$

(iii) *single-valued* if $f(a)$ is a principal downset of B for each $a \in \operatorname{dom} f$;

(iv) *projective* if $f \simeq f'$ for some single-valued $f' \colon A \rightharpoonup B$.

We list some elementary results on decidability and computational density. The proofs are left to the reader.

Proposition 3.12. *Let $A \overset{f}{\rightharpoonup} B \overset{g}{\rightharpoonup} C$ be partial applicative morphisms.*

   (*i*) *if f is c.d., then f is decidable;*

  (*ii*) *if $f' \colon A \rightharpoonup B$ satisfies $f' \le f$ and f is c.d. (resp. decidable), then f' is c.d. (resp. decidable) as well;*

 (*iii*) *if f and g are c.d. (resp. decidable), then gf is c.d. (resp. decidable) as well;*

 (*iv*) *if gf is c.d. (resp. decidable), then g is c.d. (resp. decidable) as well.*

In particular, left adjoints are c.d. and decidable.

Computational density was first introduced in the paper [HvO03]. The definition we gave above is not the original definition from that paper, but rather a simplification due to P. Johnstone (see [Joh13], Lemma 3.2). The main motivation for computational density is the following result, that we state but do not prove.

Theorem 3.13. *Let f be a* total *applicative morphism. Then:*

 (*i*) *f is projective and c.d. iff f has a right adjoint in pPCA;*

 (*ii*) *f is c.d. iff $\mathsf{Asm}(f)$ has a right adjoint, iff $\mathsf{RT}(f)$ has a right adjoint.*

In particular, every total c.d. applicative morphism $A \to B$ gives rise to geometric morphism $\mathsf{RT}(B) \to \mathsf{RT}(A)$.

Remark 3.14. Even though we do not prove Theorem 3.13, we will make some remarks on the proof method. The right adjoint of $\mathsf{Asm}(f)$, if it exists, is certainly a ∇-functor, but it is not necessarily regular. But we do know that it is left exact, and the paper [FvO14] (Theorem 2.2) shows, for the absolute case, that such functors arise from applicative morphisms $TA \to B$. The assumption that f is c.d. can then be used to construct this morphism, which yields the desired right adjoint of $\mathsf{Asm}(f)$. This construction also works for the relative case, except that one may get a *partial* applicative morphism $TA \rightharpoonup B$. This corresponds to the fact that the right adjoint of $\mathsf{Asm}(f)$ may fail to be a Γ-functor. As for (i), we have that f has a right adjoint in pPCA iff the right adjoint of $\mathsf{Asm}(f)$ is regular. This is equivalent to: $\mathsf{Asm}(f)$ preserve projective objects, which is equivalent to f being projective (hence the name).

Example 3.15. The morphism ι_r from Example 3.5 is c.d. Indeed, define the element $n := \lambda^* x.xr \in F_r$, where $F_r = \langle A^{\#} \cup \{r\} \rangle$ is as in Example 2.20. If $s \in F_r$, then by Lemma 2.12, there exists a term $t(\vec{x}, y)$ without constants such that $t(\vec{a}, r) \le s$ for certain $\vec{a} \in A^{\#}$. Now consider the element $q = \lambda^* y.t(\vec{a}, y) \in A^{\#}$. We have $nq \preceq qr \preceq t(\vec{a}, r) \le s$, which implies $n \cdot \iota_r(q) \subseteq \downarrow\{s\}$, so n satisfies (cd).

We could also have shown that ι_r is c.d. by exhibiting a right adjoint in PCA. Indeed, consider $h\colon A \to A$ defined by $h(a) = \{b \in A \mid br \leq a\}$. First of all, this is clearly a downset of A, and we have $\mathsf{k}a \in h(a)$ for all $a \in A$, and $\mathsf{k}a \in h(a) \cap A^{\#}$ for all $a \in A^{\#}$. Moreover, h preserves the order on the nose, and finally, h is tracked by $\mathsf{s} \in A^{\#}$. We also have $r \in \mathrm{dom}^{\#} h$, since $\mathsf{i} \in h(r) \cap A^{\#}$, so h is also a morphism $A[r] \to A$, by the universal property of ι_r. It is easy to see that $\mathsf{k} \in A^{\#}$ realizes both $\mathrm{id}_A \leq \iota_r h$ and $\mathrm{id}_{A[r]} \leq h\iota_r$, whereas $\lambda^* x.xr \in F_r$ realizes $\iota_r h \leq \mathrm{id}_{A[r]}$. So we have an adjunction $\iota_r \dashv h$ in PCA with $\iota_r h \simeq \mathrm{id}_{A[r]}$. Applying the construction RT now shows that $\mathsf{RT}(A[r])$ is a *subtopos* of $\mathsf{RT}(A)$. In fact, one can show (e.g., using techniques from [Zoe20]) that $\mathsf{RT}(A[r])$ is an *open* subtopos, given by the subterminal assembly 1_r defined by $|1_r| = \{*\}$ and $E_{1_r}(*) = {\downarrow}\{r\}$.

The remainder of the paper can be seen as an attempt to generalize this example to 'higher orders'.

4 Adjoining a partial function to a PCA

In this section, we generalize a construction from [vO06], which, given a (discrete, absolute) PCA A and a partial function $f\colon A \rightharpoonup A$, 'freely adjoins' f to A. In fact, this can be seen as a higher-order version of the construction of $A[r]$, where we view $A[r]$ as the zeroth-order case (adjoining an element), and the construction in this section as the first-order case. Accordingly, we will denote the resulting PCA by $A[f]$.

Here, of course, we treat the relative, ordered case. Since the order is 'hard-wired' into the PCA, we will not consider all partial functions on A, but only those that cooperate well with the order, i.e., those from $\mathcal{B}A$. The filter is a bit more flexible. For ordinary computability on $\mathbb{N}$, we know that declaring a certain function to be computable (e.g., by making it into an oracle) causes other functions to be computable as well. Because we make a distinction between computable and non-computable *elements* as well, there is also the possibility that more elements become computable, i.e., that the filter becomes larger. More precisely, we will need to close $A^{\#}$ under application of f. Here a set $X \subseteq A$ is closed under application of f if $f(a) \in A$ whenever $a \in X$ and $f(a){\downarrow}$.

Before we proceed to define $A[f]$, let us first give a precise meaning to the computability of a function $f \in \mathcal{B}A$.

Definition 4.1. Let A be a PCA and let $f \in \mathcal{B}A$.

(i) An $r \in A$ is said to *represent* f if $ra \preceq f(a)$ for all $a \in A$. The function f is called *representable* if it represented by an $r \in A$, and *effectively representable* if it is represented by an $r \in A^{\#}$.

(ii) If $g\colon A \rightharpoonup B$ is a partial applicative morphism, then we say that an $s \in B$ *represents* f *w.r.t.* g if $s \cdot g(a)$ is defined and a subset of $g(f(a))$ for all $a \in \mathrm{dom}\, f$. The function f is called *representable w.r.t. g* if it is represented w.r.t. g by an $s \in B$, and *effectively representable w.r.t. g* if it is represented w.r.t. g by an $s \in B^{\#}$.

Observe that (i) is actually a special case of (ii), if we let g be id_A.

Remark 4.2. If $r \in A$, then the partial function $\lambda a.ra$ is always in $\mathcal{B}A$. (Here we mean $\lambda a.ra$ to denote the *partial function* that sends a to ra, if defined; not to be confused with $\lambda^* a.ra \in A$, which represents this function!) This means that the set of all respresentable resp. effectively representable f can also be described as

$$\uparrow\{\lambda a.ra \mid r \in A\} \subseteq \mathcal{B}A \quad \text{resp.} \quad \uparrow\{\lambda a.ra \mid r \in A^{\#}\} \subseteq \mathcal{B}A.$$

13

This will become important in the next section.

We observe that effective representability is transferable along partial applicative morphisms in the following sense: if $A \xrightarrow{g} B \xrightarrow{h} C$ and $f \in \mathcal{B}A$ is effectively representable w.r.t. g, then f is also effectively representable w.r.t. hg. In order to prove this, assume for simplicity that h preserves the order on the nose, and let $t \in C^{\#}$ track h. If $s \in B$ represents f w.r.t. g, then $\lambda^* x.ts_0 x \in C^{\#}$ represents f w.r.t. hg, where $s_0 \in h(s) \cap C^{\#}$. A similar argument shows that representability is transferable along *total* applicative morphisms.

We will now begin the construction of the desired 'free' PCA $A[f]$ is which f is effectively representable. In order for this construction to work, we need to assume that A is not semitrivial. In fact, from this point onwards, we will assume that all PCAs we consider are not semitrivial. The underlying set of $A[f]$ will simply be A itself, and the order on $A[f]$ will simply be the order on A. However, we equip $A[f]$ with a new application operation. Informally, a computation in $A[f]$ will be a computation in A with an oracle for f. That is, the computation can feed a finite number of inputs to f before coming up with the final result. In order to distinguish this new application from the original one, we will write it as $a \odot b$. Of course, this new application will depend on f, so really, we should write $\odot_f$. If there is ambiguity as to which function plays the role of f, we will do so. However, doing so consistently would make expressions concerning $A[f]$ rather illegible, so in this section, we exclusively write $\odot$.

Definition 4.3. Let A be a PCA and let $f \in \mathcal{B}A$. We define the PAP $A[f] = (A, \odot, \leq)$ as follows. For $a, b, c \in A$, we say that $a \odot b = c$ if and only if there exists a (possibly empty) sequence $u_0, \ldots, u_{n-1} \in A$ such that

- for all $i < n$, we have $\mathsf{p}_0(a \cdot [b, u_0, \ldots, u_{i-1}]) \leq \bot$ and $f(\mathsf{p}_1(a \cdot [b, u_0, \ldots, u_{i-1}])) = u_i$;

- $\mathsf{p}_0(a \cdot [b, u_0, \ldots, u_{n-1}]) \leq \top$ and $\mathsf{p}_1(a \cdot [b, u_0, \ldots, u_{n-1}]) = c$.

The sequence $u_0, \ldots, u_{n-1}$ is called a *b-interrogation of f by a*.

Intuitively, the coefficients in the interrogation are the values the oracle returns in the course of the computation of $a \odot b$. At each stage of the computation, the algorithm a is allowed to consult the input b and the values $u_0, \ldots, u_{i-1}$ obtained from the oracle so far. Formally, this means that we let a act on the coded sequence $[b, u_0, \ldots, u_{i-1}]$. We view the result as carrying two pieces of information. The first piece is a (sub)boolean, which tells us whether the computation has gathered enough oracle values to output a result. If not, then the second piece of information is fed to the oracle; if the oracle need not be consulted anymore, then this second piece is the output.

Since A is not semitrivial, there is at most one b-interrogation of f by a, which also means that $a \odot b = c$ for at most one $c \in A$. Observe that $a \odot b$ may fail to be defined in several ways. First of all, one of the applications in A could be undefined. In addition, $\mathsf{p}_0(a \cdot [b, u_0, \ldots, u_{i-1}])$ could fail to be a subboolean or $\mathsf{p}_1(a \cdot [b, u_0, \ldots, u_{i-1}])$ could lie outside the domain of f (i.e., the oracle fails to return a value). Finally, it could happen that the computation keeps feeding inputs to the oracle indefinitely, never coming up with a final output. For example, if $a = \mathsf{k}(\mathsf{p}\bot\bot)$, then $a \odot b$ will always be undefined, even if A itself is a total PCA and f is a total function.

Remark 4.4. In the original definition of $a \odot b$ from [vO06], which is for discrete PCAs, the sequence $u_0, \ldots, u_{n-1}$ should satisfy:

- for all $i < n$, there exists a $v_i \in A$ such that $a \cdot [b, u_0, \ldots, u_{i-1}] = \mathsf{p}\perp v_i$ and $f(v_i) = u_i$;

- there exists a $c \in A$ such that $a \cdot [b, u_0, \ldots, u_{n-1}] = \mathsf{p}\top c$,

in which case $a \odot b = c$. Since we are working with ordered PCAs, however, we cannot hope to get *equalities* between elements from A, since all the available combinators only yield inequalities. We do have the following, which we will use most often when computing $a \odot b$. If there are $\bar{u}_0, \ldots, \bar{u}_{n-1}, c \in A$ such that:

- for all $i < n$, there exists a $v_i \in A$ such that $a \cdot [b, \bar{u}_0, \ldots, \bar{u}_{i-1}] \leq \mathsf{p}\perp v_i$ and $f(v_i) \leq \bar{u}_i$;

- $a \cdot [b, \bar{u}_0, \ldots, \bar{u}_{n-1}] \leq \mathsf{p}\top c$,

then $a \odot b \leq c$. (We write $\bar{u}_i$ rather than u_i because this sequence need not be the actual b-interrogation of f by a.)

Of course, we should show that $A[f]$ is actually a PAP, which is also the point where we need that $f \in \mathcal{B}A$. Suppose we have $a' \leq a$ and $b' \leq b$ such that $a \odot b\downarrow$, and let $u_0, \ldots, u_{n-1}$ be the b-interrogation of f by a. Then by induction, one easily shows that there exist $u'_i \leq u_i$ such that $u'_0, \ldots, u'_{n-1}$ is a b'-interrogation of f by a'; and from this, we get that $a' \odot b'$ is defined and $a' \odot b' \leq a \odot b$, as desired.

In order to complete the definition of $A[f]$ as a PAS, it remains to define $A[f]^\#$. If $A[f]$ is to be the free PCA in which f is effectively representable, there should be a morphism $A \to A[f]$ (cf. the zeroth-order case). The identity on A, i.e., $a \mapsto \downarrow\{a\}$, is an obvious candidate, and in order for this to be a morphism, we must have $A^\# \subseteq A[f]^\#$. We cannot, in general, let $A[f]^\#$ be *equal* to $A^\#$, however, since $A^\#$ could fail to be closed under (defined) $\odot$. The following definition remedies this.

Definition 4.5. Let A be a PCA and let $f \in \mathcal{B}A$. The PAP $A[f]$ is made into a PAS by setting $A[f]^\# := \langle A^\# \rangle$, where the generated filter is taken in the PAP $A[f]$, rather than A, of course.

This, by definition, makes $A[f]$ into a PAS.

Proposition 4.6. *For each PCA A and $f \in \mathcal{B}A$, the quadruple $A[f] = (A, A[f]^\#, \odot, \leq)$ is a PCA.*

Proof. We need to exhibit suitable combinators k_f and s_f for $A[f]$. Recall the combinator $\mathsf{fst} \in A^\#$ from Definition 2.18. For k_f, we can take $\lambda^* x.\mathsf{p}\top(\lambda^* y.\mathsf{p}\top(\mathsf{fst}x)) \in A^\# \subseteq A[f]^\#$. Indeed, if $a, b \in A$, then $\mathsf{k}_f \odot a \leq (\lambda^* y.\mathsf{p}\top(\mathsf{fst}x))[[a]/x]$, and

$$(\mathsf{k}_f \odot a) \cdot [b] \preceq (\mathsf{p}\top(\mathsf{fst}x))[[a]/x, [b]/y] = \mathsf{p}\top(\mathsf{fst}[a]) \leq \mathsf{p}\top a,$$

which means that $\mathsf{k}_f \odot a \odot b \leq a$, as desired. Observe that k_f does not, in fact, depend on f, and that the computation of $\mathsf{k}_f \odot a \odot b$ does not consult the oracle at all.

The definition of s_f (which will also not depend on f) is a little more involved, and writing down an actual definition of s_f would be quite cumbersome. Therefore, we simply explain how to construct it. Using recursion and the elementary operations on sequences and booleans, we can construct an element $S \in A^\#$ such that for all a, b and $u = [u_0, \ldots, u_{n-1}]$ from A, we have:

- if $\forall i \leq n\,(\mathsf{p}_0(x \cdot [u_0, \ldots, u_{i-1}]) \leq \perp)$, then $Sxyu \preceq xu$;

15

- if i is minimal such that $\mathsf{p}_0(x \cdot [u_0, \ldots, u_{i-1}]) \leq \top$, and

$$\forall j \leq n\,(i \leq j \to \mathsf{p}_0(y \cdot [u_0, u_i, \ldots, u_{j-1}]) \leq \bot),$$

then $Sxyu \preceq y \cdot [u_0, u_i, \ldots, u_{n-1}]$;

- if i is minimal such that $\mathsf{p}_0(x \cdot [u_0, \ldots, u_{i-1}]) \leq \top$, and $j \geq i$ is minimal such that $\mathsf{p}_0(y \cdot [u_0, u_i, \ldots, u_{j-1}]) \leq \top$, then

$$Sxyu \preceq \mathsf{p}_1(x \cdot [u_0, \ldots, u_{i-1}]) \cdot [\mathsf{p}_1(y \cdot [u_0, u_i, \ldots, u_{j-1}]), u_j, \ldots, u_{n-1}].$$

We leave it to the reader to check that $Sab \odot c \preceq a \odot c \odot (b \odot c)$ for $a, b, c \in A$. (This is a good exercise in understanding what the definition of S above actually does!) Finally, we can set

$$\mathsf{s}_f = \lambda^* x.\mathsf{p}\top(\lambda^* y.S(\mathsf{fst}x)(\mathsf{fst}y)) \in A^\# \subseteq A[f]^\#.$$

In the same way as we did for k_f, one can verify that $\mathsf{s}_f \odot a \odot b \leq Sab$, so we can conclude that $A[f]$ is a PCA. $\qquad\square$

Before we continue, we introduce two algorithms $t_f, r_f \in A^\# \subseteq A[f]^\#$ that are relevant throughout this section. Set

$$t_f := \lambda^* x.\mathsf{p}\top(\lambda^* y.\mathsf{p}\top(\mathsf{fst}x(\mathsf{fst}y))),$$

where fst is as above. A calculation similar to the ones in the proof above above shows that $t_f \odot a \odot b \preceq a \cdot b$, for all $a, b \in A$. Moreover, set

$$r_f := \lambda^* x.(\text{if } \mathsf{zero}(\mathsf{pred}(\mathsf{lh}x)) \text{ then } \mathsf{p}\bot(\mathsf{fst}x) \text{ else } \mathsf{p}\top(\mathsf{read}x1)). =,$$

where $\mathsf{lh}, \mathsf{fst}, \mathsf{read} \in A^\#$ are as in Definition 2.18. If $a \in A$ is such that $f(a)\downarrow$, then

- $r_f \cdot [a] \leq \mathsf{p}\bot a$ and $f(a)\downarrow$;

- $r_f \cdot [a, f(a)] \leq \mathsf{p}\top f(a)$,

which means that $r_f \odot a \leq f(a)$. So we have $r_f \odot a \preceq f(a)$ for all $a \in A$.
In analogy with Example 3.5, we define $\iota_f \colon A \to A[f]$ by $\iota_f(a) = \downarrow\{a\}$.

Proposition 4.7. *The map ι_f is a total applicative morphism, and f is effectively representable w.r.t. ι_f. Moreover, ι_f has a right adjoint $h \colon A[f] \to A$ satisfying $\iota_f h \simeq \mathrm{id}_{A[f]}$. In particular, ι_f is c.d. and decidable.*

Proof. Since $A^\# \subseteq A[f]^\#$, it is clear that ι_f satisfies the first requirement; and it is also obvious that ι_f preserves the order on the nose. Moreover, t_f is a tracker, so ι_f is indeed a total applicative morphism, and r_f represents f w.r.t. ι_f.
We define the required right adjoint $h \colon A[f] \to A$ by:

$$h(a) := \{b \in A \mid b \odot \mathsf{i} \leq a\},$$

where $\mathsf{i} \in A^\#$ is the identity combinator for A. It is clear that $h(a)$ is a downset, and that h preserves the order on the nose. Now consider the $S \in A^\#$ constructed in the proof of Proposition 4.6. If $b \in h(a)$ and $b' \in h(a')$, then $Sbb' \odot \mathsf{i} \preceq (b \odot \mathsf{i}) \odot (b' \odot \mathsf{i}) \preceq a \odot a'$, so if $a \odot a'\downarrow$, then $Sbb' \in h(a \odot a')$. In other words, S is a tracker for h. Next, we observe that

$\mathsf{p}\top a \in h(a)$ for all $a \in A$, so in particular, we have $A^{\#} \subseteq \mathrm{dom}^{\#}\, h$. Since h preserves the order and has a tracker, we already know that $\mathrm{dom}^{\#}\, h$ is a filter of the partial applicative *preorder* $A[f]$. Combining this with $A^{\#} \subseteq \mathrm{dom}^{\#}\, h$ yields $A[f]^{\#} \subseteq \mathrm{dom}^{\#}\, h$, so h is a total applicative morphism.

By the observation above, we know that $\mathsf{p}\top \in A^{\#}$ realizes $\mathrm{id}_A \leq h\iota_f$. Finally, $\mathrm{id}_{A[f]} \leq \iota_f h$ and $\iota_f h \leq \mathrm{id}_{A[f]}$ are realized by k_f and $\lambda^*x.x \odot \mathsf{i}$, respectively. □

Since ι_f is decidable, we see that the boolean $\bot, \top \in A^{\#}$ of A can also serve as booleans in $A[f]$. In particular, $A[f]$ is also not semitrivial, given that A is not semitrivial.

In analogy with Example 3.15, we see that $\mathsf{RT}(A[f])$ is a subtopos of $\mathsf{RT}(A)$. Before we proceed to establish the universal property of $A[f]$, we give an alternative description of $A[f]^{\#}$.
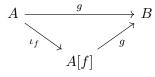
Lemma 4.8. *Let A be a PCA and let $f \in \mathcal{B}A$. Then $A[f]^{\#}$ is the least filter on A which is closed under application of f.*

Proof. We need to show the following: if $A^{\#} \subseteq X \subseteq A$ and X is upwards closed, then X is closed under defined $\odot$ if and only if X is closed under application in A and application of f. First of all, suppose that X is closed under defined $\odot$. If we have $a, b \in X$ such that $ab\!\downarrow$, then $t_f \odot a \odot b \leq ab$, so since $t_f, a, b \in X$, we have that $t_f \odot a \odot b \in X$ as well, hence $ab \in X$. Similarly, if $a \in X$ is such that $f(a)\!\downarrow$, then $r_f \odot a \leq f(a)$ yields that $f(a) \in X$ as well.

Conversely, suppose that X is closed under application in A and application of f. Suppose that we have $a, b \in X$ such that $a \odot b\!\downarrow$, and let $u_0, \ldots, u_{n-1}$ be the b-interrogation of f by a. Then using induction, we may show that $u_i \in X$ for all $i < n$. Indeed, let $i < n$ and suppose that $u_0, \ldots, u_{i-1} \in X$. Then $\mathsf{p}_1(a \cdot [b, u_0, \ldots, u_{i-1}])$, which is an expression built using $a, b, u_0, \ldots, u_{i-1} \in X$, combinators from $A^{\#} \subseteq X$ and application, must be in X as well. Since X is closed under f, it follows that $u_i \in X$ as well, completing the induction. Finally, we see that $a \odot b = \mathsf{p}_1(a \cdot [b, u_0, \ldots, u_{n-1}])$ is in X as well, completing the proof. □

Theorem 4.9. *Let $g\colon A \rightharpoonup B$ be a decidable partial applicative morphism, and let $f \in \mathcal{B}A$. Then g factors, up to isomorphism, through ι_f if and only if f is effectively representable w.r.t. g.*

Proof. The 'only if' statement is clear, since f is effectively representable w.r.t. ι_f, and effective representability transfers along partial applicative morphisms. Conversely, suppose that f is effectively representable w.r.t. g. We need to show that g is also a partial applicative morphism $A[f] \rightharpoonup B$; then we will have that the triangle

$$A \xrightarrow{\quad g \quad} B$$
$$\iota_f \searrow \qquad \nearrow g$$
$$A[f]$$

commutes up to isomorphism. Assume for simplicity that g preserves the order on the nose, let $t \in B^{\#}$ be a tracker of f, let $d \in B^{\#}$ be a decider for g, and let $s \in B^{\#}$ represent f w.r.t. g.

First, we need to show that $A[f]^{\#} \subseteq \mathrm{dom}^{\#}\, g$. We already know that $\mathrm{dom}^{\#}\, g$ is a filter of the PCA A, so according to Lemma 4.8, it suffices to show that $\mathrm{dom}^{\#}\, g$ is closed under application of f. So suppose that $a \in \mathrm{dom}^{\#}\, g$ and that $f(a)\!\downarrow$. Then there exists a $b \in g(a) \cap B^{\#}$, and we get $sb \in g(f(a)) \cap B^{\#}$, so $f(a) \in \mathrm{dom}^{\#}\, g$ as well, as desired. Of course, g still preserves the order when considered as a morphism $A[f] \rightharpoonup B$, so it remains to construct a tracker.

First, recall the combinators $\mathsf{unit}, \mathsf{ext} \in A^{\#}$ from Definition 2.18. For $i = 0, 1$, define $\mathsf{p}'_i = \lambda^* x.tp_i x \in B^{\#}$, where p_i is any element from $g(\mathsf{p}_i) \cap B^{\#}$. This element has the property that $\mathsf{p}'_i \cdot g(a) \preceq g(\mathsf{p}_i a)$ for $a \in A$. Similarly, using an element from $g(\mathsf{unit}) \cap B^{\#}$, we define $\mathsf{unit}' \in B^{\#}$ such that $\mathsf{unit}' \cdot g(a) \subseteq g(\mathsf{unit} \cdot a) \subseteq g([a])$ for all $a \in A$. Moreover, we may define $\mathsf{ext}' \in B^{\#}$ such that $\mathsf{ext}' \cdot g([a_0, \ldots, a_{n-1}]) \cdot g(a') \subseteq g([a_0, \ldots, a_{n-1}, a'])$. Using the fixed point operator in $B^{\#}$, we can construct an element $T \in B^{\#}$ satisfying:

$$Tbv \preceq \text{ if } d(\mathsf{p}'_0(tbv))) \text{ then } \mathsf{p}'_1(tbv) \text{ else } Tb\big(\mathsf{ext}'v(s(\mathsf{p}'_1(tbv)))\big). \tag{1}$$

Suppose that $a, a' \in A$ are such that $a \odot a' \downarrow$, and let $u_0, \ldots, u_{n-1}$ be the a'-interrogation of f by a. First of all, we claim that

$$T \cdot g(a) \cdot g([a', u_0, \ldots, u_{i-1}]) \preceq T \cdot g(a) \cdot g([a', u_0, \ldots, u_i]) \tag{2}$$

for all $i < n$. Suppose that the right hand side of (2) is defined, and consider $b \in g(a)$ and $v \in g([a', u_0, \ldots, u_{i-1}])$. Then we have $tbv \in g(a \cdot [a', u_0, \ldots, u_{i-1}])$, so

$$\mathsf{p}'_0(tbv) \in g(\mathsf{p}_0(a \cdot [a', u_0, \ldots, u_{i-1}])) \subseteq g(\bot).$$

This gives $d(\mathsf{p}'_0(tbv)) \leq \bot$, so we need to evaluate the 'else' clause in (1). We have $\mathsf{p}'_1(tbv) \in g(\mathsf{p}_1(a \cdot [a', u_0, \ldots, u_{i-1}]))$, which gives $s(\mathsf{p}'_1(tbv)) \in g(f(\mathsf{p}_1(a \cdot [a', u_0, \ldots, u_{i-1}]))) = g(u_i)$. Now $\mathsf{ext}'v(s(\mathsf{p}'_1(tbv))) \in g([a', u_0, \ldots, u_i])$, and (1) tells us that $Tbv \leq Tb\big(\mathsf{ext}'v(s(\mathsf{p}'_1(tbv)))\big) \in T \cdot g(a) \cdot g([a', u_0, \ldots, u_i])$, as desired.

Moreover, we have:

$$T \cdot g(a) \cdot g([a', u_0, \ldots, u_{n-1}]) \subseteq g(a \odot a'). \tag{3}$$

Indeed, consider $b \in g(a)$ and $v \in g([a', u_0, \ldots, u_{n-1}])$. Then as above, we find $d(\mathsf{p}'_0(tbv))) \leq \top$, so (1) tells us that $Tbv \leq \mathsf{p}'_1(tbv) \in g(\mathsf{p}_1(a \cdot [a', u_0, \ldots, u_{n-1}])) = g(a \odot a')$. Combining (2) and (3) yields:

$$T \cdot g(a) \cdot g([a']) \subseteq g(a \odot a')$$

whenever $a \odot a' \downarrow$. We conclude that $\lambda^* xy.Tx(\mathsf{unit}'y)$ tracks $g \colon A[f] \rightharpoonup B$, which finishes the proof. $\square$

Since identities are c.d., and c.d. partial applicative morphisms are closed under composition, there is a wide subcategory $\mathsf{pPCA}_{\mathrm{cd}}$ of pPCA consisting of only the c.d. partial applicative morphisms. Similarly, we have the subcategory $\mathsf{pPCA}_{\mathrm{dec}}$ consisting of only the decidable morphisms. We define $\mathsf{PCA}_{\mathrm{cd}}$ and $\mathsf{PCA}_{\mathrm{dec}}$ analogously. The following now easily follows from Theorem 4.9.

Corollary 4.10. *Let A, B be PCAs, let $f \in \mathcal{B}A$ and let $\mathcal{C}$ be any of the preorder-enriched categories $\mathsf{pPCA}_{\mathrm{cd}}, \mathsf{pPCA}_{\mathrm{dec}}, \mathsf{PCA}_{\mathrm{cd}}, \mathsf{PCA}_{\mathrm{dec}}$. Then composition with ι_f:*

$$\mathcal{C}(A[f], B) \to \{g \in \mathcal{C}(A, B) \mid f \text{ is effectively representable w.r.t. } g\}$$
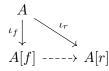
is an equivalence of preorders.

Proof. Theorem 4.9 readily implies that for each of the $\mathcal{C}$, the map above is essentially surjective. Moreover, composition with ι_f reflects the order since ι_f has a right pseudoinverse as shown in Proposition 4.7. $\square$

Example 4.11. The construction of $A[f]$ is a generalization of oracle computations for classical Turing computability. Indeed, if $f \colon \mathbb{N} \rightharpoonup \mathbb{N}$ is a partial function, then $g \colon \mathbb{N} \rightharpoonup \mathbb{N}$ is effectively representable w.r.t. $\iota_f \colon \mathcal{K}_1 \to \mathcal{K}_1[f]$ iff g is Turing computable relative to an oracle for f. See also Corollary 2.3 in [vO06].

Example 4.12. As we mentioned at the beginning, the construction from this section can be seen as a higher-order version of the construction from Example 3.5. On the other hand, the construction of $A[r]$ can be seen as a special case of the construction $A[f]$. Indeed, consider $r \in A$ and denote the constant function with value r, which is an element of $\mathcal{B}A$, by $\hat{r}$. It is easy to see that, for any $g \colon A \rightharpoonup B$, we have that $\hat{r}$ is effectively representable w.r.t. g iff $r \in \mathrm{dom}^{\#} g$. It follows that $A[r]$ and $A[\hat{r}]$ are equivalent PCAs.

Example 4.13. Of course, if $f \in \mathcal{B}A$ is already effectively representable in A itself, then $\iota_f \colon A \to A[f]$ will be an isomorphism of PCAs. Now suppose that f is represented by an element $r \in A$ (but not necessarily $r \in A^{\#}$). Then f is effectively representable w.r.t. $\iota_r \colon A \to A[r]$, so we get a factorization:

$$
\begin{array}{ccc}
 & A & \\
\iota_f \downarrow & \searrow^{\iota_r} & \\
A[f] & \dashrightarrow & A[r]
\end{array}
$$

It is worth observing that the mediating arrow $A[f] \to A[r]$ is *not*, in general, an isomorphism. Indeed, consider, e.g., a PCA with $A^{\#} \neq A$, and take a $b \in A \backslash A^{\#}$. Then the partial function $f \in \mathcal{B}A$ defined by

$$
f(a) = \begin{cases} \mathsf{p}_1 a & \text{if } \mathsf{p}_0 a \leq \top; \\ \text{undefined} & \text{else} \end{cases}
$$

is effectively representable, e.g., by $\mathsf{p}_1 \in A^{\#}$. This means that ι_f is an isomorphism. But it is *also* representable by $r := \lambda^* x.\text{if } \mathsf{p}_0 x \text{ then } \mathsf{p}_1 x \text{ else } b$. Moreover, we have $r \notin A^{\#}$, because $r(\mathsf{p}\bot\bot) \leq b$, so $r \in A^{\#}$ would imply $b \in A^{\#}$. This means that ι_r is *not* an isomorphism, so $A[f] \to A[r]$ cannot be an isomorphism either. So we see that the point here is, really, that a function $f \in \mathcal{B}A$ can have many representers.

5 The PCA of partial functions

In this section, we show how to turn the set $\mathcal{B}A$ from Section 2 into a PCA. As the order, we use the order defined in Definition 2.2(ii). In particular, the empty function is a largest element of $\mathcal{B}A$. It is worth noting that, in contrast with $A[f]$, the order on $\mathcal{B}A$ is *not* discrete even if the order on A is. Indeed, if A is discrete, then $\mathcal{B}A$ consists of all partial functions $A \rightharpoonup A$, and the order is the *reverse* subfunction relation. In this case, the total functions are minimal elements of $\mathcal{B}A$. In the general case, the total functions in $\mathcal{B}A$ form a downwards closed set.

The application on $\mathcal{B}A$ will, in a sense, generalize the $A[f]$ for $f \in \mathcal{B}A$ all at once. As for the construction of $A[f]$, we need the assumption that A is not semitrivial. An important thing to note about the application on $\mathcal{B}A$ is that it will be *total*.

Let us define the application now. For $\alpha, \beta \in \mathcal{A}$ and $a, b \in A$, we say that $\alpha\beta(a) = b$ if and only if there are $u_0, \ldots, u_{n-1}$ such that:

- for all $i < n$, we have $\mathsf{p}_0 \cdot \alpha([a, u_0, \ldots, u_{i-1}]) \leq \bot$ and $\beta(\mathsf{p}_1 \cdot \alpha([a, u_0, \ldots, u_{i-1}])) = u_i$;

- $p_0 \cdot \alpha([a, u_0, \ldots, u_{n-1}]) \leq \top$ and $p_1 \cdot \alpha([b, u_0, \ldots, u_{i-1}]) = b$.

By the assumption that A is nontrivial, the sequence $u_0, \ldots, u_{n-1}$ is unique if it exists, and if it exists it is called the a-interrogation of β by α. The following table compares the definition of $\alpha\beta(a)$ with the definition of $a \odot_f b$ from Definition 4.3.

	$a \odot_f b$	$\alpha\beta(a)$
Interrogator	a	α
Input	b	a
Oracle	f	β

The above defines $\alpha\beta$ as a partial function $A \rightharpoonup A$; we leave it to the reader to check that $\alpha\beta$ is actually in $\mathcal{B}A$, and that this application makes $\mathcal{B}A$ into a PAP. The argument is very similar to the proof that $A[f]$ is a PAP. It should be noted that, even though $\alpha\beta$ is always defined, $\alpha\beta(a)$ could be undefined for the same reasons $a \odot_f b$ could be undefined. Moreover, a remark similar to Remark 4.4 applies: the PCA $\mathcal{B}A$ was introduced in [vO11] for discrete PCAs, where one finds a definition of application involving equalities between elements of A. Again, this version of the definition is not suitable for the ordered case.

For $(\mathcal{B}A)^{\#}$, we take the set of all effectively representable functions from $\mathcal{B}A$. By Remark 4.2, this is equivalent to saying:

$$(\mathcal{B}A)^{\#} = \uparrow\{\lambda a.ra \mid r \in A^{\#}\}.$$

This set is clearly upwards closed, so we need to check that it also closed under the application defined above. This requires a bit more work. Suppose that we have $\rho, \sigma \in (\mathcal{B}A)^{\#}$, and take $r, s \in A^{\#}$ such that $ra \preceq \rho(a)$ and $sa \preceq \sigma(a)$. Using the fixpoint operator, we may find a $T \in A^{\#}$ such that

$$Txyu \preceq \text{ if } p_0(xu) \text{ then } p_1(xu) \text{ else } Txy(\text{ext} \cdot u \cdot (y(p_1 u))),$$

where $\text{ext} \in A^{\#}$ is as in Definition 2.18. Now it is not hard to check that $Trs \cdot [a] \preceq \rho\sigma(a)$, which means that $\rho\sigma$ is represented by $\lambda^* x.Trs(\text{unit} \cdot x) \in A^{\#}$. We can conclude that $\rho\sigma \in A^{\#}$ as well, so $\mathcal{B}A$ is a PAS.

Showing that $\mathcal{B}A$ is a PCA is rather involved, and we will not provide all the details here. In fact, the k-combinator is still easy: using operations on sequences, one may construct an $r \in A^{\#}$ such that:

- $r \cdot [[[a]]] \leq p\perp a$;

- $r \cdot [[[a], b]] \leq p\top(p\top(p\top b))$.

Then $\kappa \in (\mathcal{B}A)^{\#}$ defined by $\kappa(a) \simeq ra$ will satisfy $\kappa\alpha\beta(a) \preceq \alpha(a)$, hence $\kappa\alpha\beta \leq \alpha$. The s-combinator $\sigma \in (\mathcal{B}A)^{\#}$ can be constructed in the following way. For $\alpha, \beta \in \mathcal{B}A$, the function $\sigma\alpha\beta$ should be an interrogator that, when provided with oracle γ, works as follows. Suppose that $\alpha\gamma(\beta\gamma)(a)$ is defined with interrogation sequence $u_0, \ldots, u_{n-1}$. On input $[a]$, it first simulates the computation of $\alpha\gamma([a])$. When finished, it checks whether $p_0 \cdot (\alpha\gamma([a]))$ holds. If so, it can output $\alpha\gamma([a])$. If not, it proceeds to simulate, again using the oracle γ, the computation $\beta\gamma(p_1 \cdot \alpha\gamma([a]))$, thus finding u_0. Then it proceeds to simulate the computation of $\alpha\gamma([a, u_0])$. It keeps going back and forth between simulating a computation instances of $\alpha\gamma$ and $\beta\gamma$, reconstruction the entire sequence $u_0, \ldots, u_{n-1}$. Then it finally finds that $p_0 \cdot \alpha\gamma([a, u_0, \ldots, u_{n-1}])$ holds, and it can give the correct output. This means that the task is to construct a σ that, when fed oracles for α and β, produces such an interrogator $\sigma\alpha\beta$.

Details on how to do this may be found in [vO11]; it does not show that $\boldsymbol{\sigma}$ is an effective computation, but the constructions involved are clearly effective computations in A. See also the following remark, however.

Remark 5.1. The paper [vO11] treats $\mathcal{B}A$ as an absolute, discrete PCA. In this paper, on the other hand, we want to consider $\mathcal{B}A$ as a relative PCA, with $(\mathcal{B}A)^{\#}$ consisting of the effectively representable functions. By doing so, the applicative morphism i in Proposition 5.2 below becomes compuationally dense, so we have a geometric morphism $\mathsf{RT}(\mathcal{B}A) \to \mathsf{RT}(A)$, which is even a surjection (Proposition 5.7). Moreover, it allows us to apply the construction from Example 3.5 to $\mathcal{B}A$ in a nontrivial way, which is essential for the new topos-theoretic construction of $\mathsf{RT}(A[f])$ described below.

The move to relativity also *forces* us to view $\mathcal{B}A$ as an ordered PCA. Indeed, we do not have that $(\mathcal{B}A, (\mathcal{B}A)^{\#}, \cdot, =)$ is always a PCA, even if A itself is discrete. What we have, is an effectively representable $\boldsymbol{\sigma}$ such that: if $\alpha\gamma(\beta\gamma)(a)$ is defined, then $\boldsymbol{\sigma}\alpha\beta\gamma(a)$ is also defined with (in the discrete case) the same value. However, if $\mathcal{B}A$ is equipped with the discrete order, then $\alpha\gamma(\beta\gamma)$ and $\boldsymbol{\sigma}\alpha\beta\gamma$ should be the *same* function. And this is not automatically true. Indeed, one of the reasons $\alpha\gamma(\beta\gamma)$ could turn out undefined, is that a certain intermediate result that should be a boolean, is not in fact a boolean. When simulating computations, $\boldsymbol{\sigma}$ feeds such expressions to the if-then-else operator, and this may very well yield an unintended result. One could remedy this by setting all such unintended results to undefined, or to an output that sends the computation of $\boldsymbol{\sigma}\alpha\beta\gamma(a)$ into an infinite loop, but one cannot hope, in general, that this keeps $\boldsymbol{\sigma}$ effective. It may *happen* to be possible, of course. The most prominent example is Kleene's first model $\mathcal{K}_1$, which allows a nice coding of booleans for which this problem does not arise.

The main focus of [vO11] is actually not $\mathcal{B}A$, but a related PCA $\mathcal{K}_2A$ whose elements are the *total* functions $A \to A$. The application is defined in the same way, except that $\alpha\beta$ is only defined if $\alpha\beta(a)$ is defined for all $a \in A$. Then the problem described above does not arise, since all the functions involved are total. But there is another problem, namely that the $\boldsymbol{\sigma}$ described above will clearly be *partial*. In order to make $\boldsymbol{\sigma}$ into an element of $\mathcal{K}_2A$, we need to extend it to a total function, and once again, this can not necessarily be done in an effective way. Kleene's first model again forms an exception; indeed, it is well known that Kleene's *second* model, which is $\mathcal{K}_2(\mathcal{K}_1)$, has a relative version.

For $a \in A$, let $\hat{a} \in \mathcal{B}A$ denote the constant function with value a.

Proposition 5.2. *There is a total c.d. applicative morphism* $i \colon A \to \mathcal{B}A$, *defined by* $i(a) = {\downarrow}\{\hat{a}\}$.

Proof. If $a \in A^{\#}$, then $\hat{a}$ is clearly represented by $\mathsf{k}a \in A^{\#}$, so $\hat{a} \in (\mathcal{B}A)^{\#}$, meaning that i satisfies the first requirement. A similar argument shows that i is total. Moreover, i clearly preserves the order on the nose, so in order to show it is an applicative morphism, it remains to construct a tracker. Using the elementary operations on sequences, we may contruct a $t \in A^{\#}$ satisfying:

- $t \cdot [[x]] \leq \mathsf{p}\top(\mathsf{p}\bot\mathsf{i})$;

- $t \cdot [[x, y]] \leq \mathsf{p}\bot\mathsf{i}$;

- $t \cdot [[x, y], z] \preceq \mathsf{p}\top(\mathsf{p}\top(zy))$.

21

If $\tau \in (\mathcal{B}A)^{\#}$ is defined by $\tau(a) \simeq ta$, then it is straightforward to check that $\tau\hat{a}\hat{b}(c) \preceq ab$ for all $a, b, c \in A$, so τ is a tracker for i.

For computational density, we take an $n \in A^{\#}$ satisfying:

- $n \cdot [x] \leq \mathsf{p}\bot\mathsf{i}$;

- $n \cdot [x, y] \preceq \mathsf{p}\top(yx)$.

If $\nu \in (\mathcal{B}A)^{\#}$ is defined by $\nu(a) \simeq na$, then it is easy to check that $\nu\hat{r}(a) \preceq ra$ for all $a, r \in A$. So, if $\rho \in (\mathcal{B}A)^{\#}$ is represented by $r \in A^{\#}$, then $\nu\hat{r} \leq \rho$, so $\nu \cdot i(r) \subseteq {\downarrow}\{\rho\}$, showing that ν satisfies (cd). $\qquad\square$

Since i is c.d., it is decidable as well, which means that $\widehat{\top}, \widehat{\bot}$ can serve as booleans in $\mathcal{B}A$. In particular, $\mathcal{B}A$ is not semitrivial, given that A is not semitrivial.

The fact that i is c.d. is the main reason for viewing $\mathcal{B}A$ as a relative PCA. If we consider $\mathcal{B}A$ as an absolute PCA, then the corresponding result does not hold for cardinality reasons. Indeed, one might say that 'there are more functions than respresenters'. The computational density of i means that we get a geometric morphism $\mathsf{RT}(\mathcal{B}A) \to \mathsf{RT}(A)$. In fact, the right adjoint of i must already exist at the level of PCAs, since i is also total and projective. We will construct this right adjoint explicitly.

As in [vO11], the PCA $\mathcal{B}A$ has the property that every element of $\mathcal{B}A$ is representable w.r.t. $i\colon A \to \mathcal{B}A$.

Proposition 5.3. *Let A be a PCA. Then every $\alpha \in \mathcal{B}A$ is representable w.r.t. $i\colon A \to \mathcal{B}A$.*

Proof. Construct an $r \in A^{\#}$ satisfying:

- $r \cdot [[x]] \leq \mathsf{p}\top(\mathsf{p}\bot\mathsf{i})$;

- $r \cdot [[x, y]] \leq \mathsf{p}\bot y$;

- $r \cdot [[x, y], z] \leq \mathsf{p}\top(\mathsf{p}\top z)$,

and let $\rho \in (\mathcal{B}A)^{\#}$ be defined by $\rho(a) \simeq ra$. Then it is easily checked that $\rho\alpha\hat{a} \preceq \widehat{\alpha(a)}$ for all $\alpha \in \mathcal{B}A$ and $a \in A$. It follows that $\rho\alpha$ represents α w.r.t. i, for all $\alpha \in \mathcal{B}A$. $\qquad\square$

The following result also appears as Proposition 5.1 in [vO11]. Since we have introduced the concept of a *partial* applicative morphism, we can formulate the result here in a nicer way.

Theorem 5.4. *Let $g\colon A \rightharpoonup B$ be a decidable partial applicative morphism. Then there exists a largest partial applicative morphism $h\colon \mathcal{B}A \rightharpoonup B$ such that $hi \simeq g$, and it is explicitly defined by:*

$$h(\alpha) = \{b \in B \mid b \text{ represents } \alpha \text{ w.r.t. } g\} = \{b \in B \mid \forall a \in A(\alpha(a){\downarrow} \to b \cdot g(a) \subseteq g(\alpha(a)))\}.$$

Proof. It is easy to see that $h(\alpha)$ is a downset of B. For simplicity, we will assume that g preserves the order on the nose; then it follows that h preserves the order on the nose as well. Moreover, if $\alpha \in (\mathcal{B}A)^{\#}$, then α is effectively representable in A, which implies that α is also effectively representable w.r.t. g, which means that $h(\alpha) \cap B^{\#}$ is non-empty. So in order to show that h is a partial applicative morphism it remains to construct a tracker.

The construction of a tracker will be very similar to the construction of a tracker in the proof of Theorem 4.9. Let $t \in B^{\#}$ be a tracker of g and let $d \in B^{\#}$ be a decider of g. Moreover,

take $\mathsf{ext}', \mathsf{p}_0', \mathsf{p}_1', \mathsf{unit}' \in B^{\#}$ as in the proof of Theorem 4.9. Using the fixpoint operator in B, we may construct a $U \in B^{\#}$ such that

$$Uxyv \preceq \text{ if } d(\mathsf{p}_0'(xv)) \text{ then } \mathsf{p}_1'(xv) \text{ else } Uxy(\mathsf{ext}'v(y(\mathsf{p}_1'(xv)))).$$

Using the familiar kind of argument, one can show that $T \cdot h(\alpha) \cdot h(\beta) \cdot g([a]) \subseteq g(\alpha\beta(a))$, whenever $\alpha, \beta \in \mathcal{B}(A)$ and $a \in A$ are such that $\alpha\beta(a)\downarrow$. It follows that $\lambda^* xyz.Uxy(\mathsf{unit}'z) \in B^{\#}$ tracks h.

For $a \in A$, we have:

$$hi(a) = h(\hat{a}) = \{b \in B \mid \forall a' \in A\,(b \cdot g(a') \subseteq g(a))\}.$$

Now it is easy to see that $\mathsf{k} \in B^{\#}$ realizes $g \leq hi$, and if $j \in g(\mathsf{i}) \cap B^{\#}$, then $\lambda^* x.xj \in B^{\#}$ realizes $hi \leq g$. So we indeed have $hi \simeq g$.

In order to show that h is the *largest* partial appplicative morphism such that $hi \simeq g$, suppose we have another $h' \colon \mathcal{B}A \rightharpoonup B$ such that $h'i \simeq g$, and assume that h' preserves the order on the nose. Let $t' \in B^{\#}$ be a tracker of h', and let $r, s \in B^{\#}$ realize $h'i \leq g$ resp. $g \leq h'i$. Moreover, consider the function $\rho \in (\mathcal{B}A)^{\#}$ from the proof of Proposition 5.3; it has the property that $\rho\alpha\hat{a} \preceq \widehat{\alpha(a)}$ for all $\alpha \in \mathcal{B}A$ and $a \in A$. We pick a $q \in h'(\rho) \cap B^{\#}$. Now it is easy to see that, if $b \in g(a)$, $c \in h'(\alpha)$ and $\alpha(a)\downarrow$, we have $r(t'(t'qc)(sb)) \in g(\alpha(a))$. This implies that $\lambda^* xy.r(t'(t'qx)(sy)) \in B^{\#}$ realizes $h' \leq h$, as desired. □

Observe that the h constructed above will automatically be decidable as well, since $hi \simeq g$.

Corollary 5.5. *Let* $g \colon A \rightharpoonup B$ *be a decidable partial applicative morphism. Then*

$$\{\alpha \in \mathcal{B}A \mid \alpha \text{ is representable w.r.t. } g\} \quad \text{and}$$
$$\{\alpha \in \mathcal{B}A \mid \alpha \text{ is effectively representable w.r.t. } g\}$$

are filters of $\mathcal{B}A$.

Proof. If we let $h \colon \mathcal{B}A \rightharpoonup B$ be as in Theorem 5.4, then these sets are $\mathrm{dom}\,h$ and $\mathrm{dom}^{\#}\,h$, respectively. □

Example 5.6. If we apply Theorem 5.4 to i itself, then we see that

$$h(\alpha) = \{\beta \in \mathcal{B}A \mid \beta \text{ represents } \alpha \text{ w.r.t. } i\}$$

defines the largest $h \colon \mathcal{B}A \to \mathcal{B}A$ such that $hi \simeq i$. We claim that h is isomorpic to the identity. By Theorem 5.4, we already know that $\mathrm{id}_{\mathcal{B}A} \leq h$. For the converse inequality, construct an $r \in A^{\#}$ such that:

- $r \cdot [x] \leq \mathsf{p}{\perp}[\mathsf{i}]$;

- $r \cdot [x, u_0 \ldots, u_i] \leq \text{ if } \mathsf{p}_0 u_i \text{ then } \mathsf{p}{\top}(\mathsf{p}_1 u_i) \text{ else } \mathsf{p}{\perp}[\mathsf{i}, \underbrace{x, \ldots, x}_{i+1 \text{ times}}]$, for $i \geq 0$.

If $\rho \in (\mathcal{B}A)^{\#}$ is defined by $\rho(a) \simeq ra$, then $\rho\beta(a) \leq \beta\hat{a}(\mathsf{i})$ for all $a \in A$ and $\beta \in \mathcal{B}A$. In particular, if $\beta \in h(\alpha)$, then $\rho\beta \leq \alpha$, so ρ realizes $h \leq \mathrm{id}_{\mathcal{B}A}$. In a slogan, we could say that *elements* of $\mathcal{B}A$ and their *representers* can be used interchangably in $\mathcal{B}A$.

If $f \in \mathcal{B}A$, then we can consider $\iota_f \colon \mathcal{B}A \to \mathcal{B}A[f]$ as in Example 3.5. A completely similar argument shows that ι_f is the largest $h \colon \mathcal{B}A \to \mathcal{B}A[f]$ such that $hi \simeq \iota_f i$, so the slogan

23

remains true for this case. If we add a *function* (on $\mathcal{B}A$) to $\mathcal{B}A$, however, then this principle will break down. It is clear that the argument above will not work, because adding a function involves changing the application. In the next section, we will give an explicit counterexample (Example 6.16).

Another example of Theorem 5.4 is given by the following result. Again, we stress that such a result does not hold if $\mathcal{B}A$ is taken to be an absolute PCA.

Proposition 5.7. *The applicative morphism* $i\colon A \to \mathcal{B}A$ *has a right adjoint* $h\colon \mathcal{B}A \rightharpoonup A$ *satisfying* $hi \simeq \mathrm{id}_A$. *In particular, there is a geometric surjection* $\mathsf{RT}(\mathcal{B}A) \twoheadrightarrow \mathsf{RT}(A)$.

Proof. By Theorem 5.4, there is a (largest) partial applicative morphism $h\colon \mathcal{B}A \to A$ such that $hi \simeq \mathrm{id}_A$. We also have $ihi \simeq i$, and by Example 5.6, this implies $ih \leq \mathrm{id}_{\mathcal{B}A}$. $\square$

As a corollary, we see that, while i makes every element of $\mathcal{B}A$ representable, it adds no new *effectively* representable functions.

Corollary 5.8. *An element of* $\mathcal{B}A$ *is effectively representable w.r.t. i iff it is effectively representable in A itself.*

Proof. This is immediate from the existence of a left inverse for i in pPCA, along with the fact that *effective* representability transfers along partial applicative morphisms. $\square$
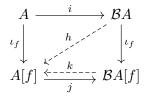
We close this section with a novel topos-theoretic interpretation of the construction from the previous section, which freely adjoins a partial function to a PCA. For $f \in \mathcal{B}A$, we can adjoin f as a function to A, yielding $\iota_f\colon A \to A[f]$, but we can also adjoin it as an *element* to $\mathcal{B}A$, which gives $\iota_f\colon \mathcal{B}A \to \mathcal{B}A[f]$. In the proof of Proposition 5.3, we constructed a $\rho \in (\mathcal{B}A)^{\#}$ such that $\rho\alpha$ is defined and represents α w.r.t. i, for all $\alpha \in \mathcal{B}A$. From this, we easily deduce that $\rho f \in (\mathcal{B}A[f])^{\#}$ represents f w.r.t. $\iota_f i$. This means we get a factorisation:

$$
\begin{array}{ccc}
A & \xrightarrow{\ \ i\ \ } & \mathcal{B}A \\
{\scriptstyle \iota_f}\downarrow & & \downarrow{\scriptstyle \iota_f} \\
A[f] & \dashrightarrow{\ j\ } & \mathcal{B}A[f]
\end{array}
$$

where j acts as i does, i.e., $j(a) = \downarrow\{\hat{a}\}$.
According to Theorem 5.4,

$$h(\alpha) = \{a \in A \mid a \text{ represents } \alpha \text{ w.r.t. } \iota_f\colon A \to A[f]\}$$

defines a partial applicative morphism $h\colon \mathcal{B}A \rightharpoonup A[f]$ such that $hi \simeq \iota_f$. By construction, f is effectively representable w.r.t. ι_f, so $f \in \mathrm{dom}^{\#}\, h$. This means that h factors through $\iota_f\colon \mathcal{B}A \to \mathcal{B}A[f]$ as $h \simeq k\iota_f$ for some $k\colon \mathcal{B}A[f] \rightharpoonup A[f]$.

$$
\begin{array}{ccc}
A & \xrightarrow{\ \ i\ \ } & \mathcal{B}A \\
{\scriptstyle \iota_f}\downarrow & {\scriptstyle h} & \downarrow{\scriptstyle \iota_f} \\
A[f] & \underset{j}{\overset{k}{\leftrightarrows}} & \mathcal{B}A[f]
\end{array}
$$

Now we have

$$kj\iota_f \simeq k\iota_f i \simeq hi \simeq \iota_f,$$

24

so by Corollary 4.10, we have $kj \simeq \mathrm{id}_{A[f]}$. On the other hand,

$$jk\iota_f i \simeq jhi \simeq j\iota_f \simeq \iota_f i.$$

By Example 5.6, this implies $jk\iota_f \leq \iota_f$, which yields $jk \leq \mathrm{id}_{\mathcal{B}A[f]}$. We conclude that $j \dashv k$ with $kj \simeq \mathrm{id}_{A[f]}$.

Recall from Example 3.15 that $\mathsf{RT}(\mathcal{B}A[f])$ is equivalent to the slice of $\mathsf{RT}(\mathcal{B}A)$ over the subterminal assembly 1_f, which is given by $|1_f| = \{*\}$ and $E_{1_f}(*) = \downarrow\{f\}$. So at the level of toposes, we get the following diagram:

$$
\begin{array}{ccc}
\mathsf{RT}(\mathcal{B}A)/1_f & \hookrightarrow & \mathsf{RT}(\mathcal{B}A) \\
\downarrow & & \downarrow \\
\mathsf{RT}(A[f]) & \hookrightarrow & \mathsf{RT}(A)
\end{array}
$$

Intuitively, this diagram may be explained as follows. First, we cover $\mathsf{RT}(A)$ by the topos $\mathsf{RT}(\mathcal{B}A)$, where the truth values are sets of *functions* rather than sets of elements of A. Then, we make f computable (or alternatively: true) at *that* level by taking the slice subtopos $\mathsf{RT}(\mathcal{B}A)/1_f$. Finally, $\mathsf{RT}(A[f])$ is retrieved as the geometric surjection-inclusion factorisation of the composition $\mathsf{RT}(\mathcal{B}A)/1_f \hookrightarrow \mathsf{RT}(\mathcal{B}A) \twoheadrightarrow \mathsf{RT}(A)$.

Another application of the construction of j and k above is the following nice characteization of the elements in $\mathcal{B}A$ that are effectively representable w.r.t. $\iota_f : A \to A[f]$. Since j has a left inverse, we have that $\alpha \in \mathcal{B}A$ is effectively representable w.r.t. ι_f iff α is effectively representable w.r.t. $j\iota_f \simeq \iota_f i$. By Example 5.6, we know that $\alpha \in \mathcal{B}A$ and representers of α w.r.t. $\iota_f i$ can be effectively translated into one another, so α is representable w.r.t. $\iota_f i$ iff $\alpha \in (\mathcal{B}A[f])^{\#} = \langle(\mathcal{B}A)^{\#} \cup \{f\}\rangle$. We can conclude that

$$\{\alpha \in \mathcal{B}A \mid \alpha \text{ is effectively representable w.r.t. } \iota_f : A \to A[f]\} = \langle(\mathcal{B}A)^{\#} \cup \{f\}\rangle.$$

In other words, the set of all elements of $\mathcal{B}A$ that are 'forced' to become effectively representable if f is effectively representable, can simply be obtained by adding f to the filter in the PCA $\mathcal{B}A$.

6 Adjoining a type-2 functional

In the previous section, we have explained how to freely adjoin zeroth- and first-order functions to a PCA A. This prompts the question whether a similar construction is available for *second-order* functions. The answer to this question is yes, as the paper [FvO16] shows for the absolute discrete case. In this section, we generalize this construction to *some* relative ordered PCAs; it will in fact turn out that the order needs to be nice in a specific sense for the construction to work. There are two reasons for generalizing the material from [FvO16]. First of all, the construction ties in quite nicely with Theorem 5.4 above in a way that had not been observed before. Second, in the next section we will apply this construction to $\mathcal{B}A$ (which is always an ordered PCA), in order to see what one can obtain for the *third*-order case.

Let us first introduce the restriction on PCAs we need in order for the construction to work.

Definition 6.1. Let A be a PCA.

(i) A *chain* in A is a non-empty subset $X \subseteq A$ that is totally ordered by $\leq$.

(ii) A is called *chain-complete* if every chain $X \subseteq A$ has a greatest lower bound in A, which we will denote by $\bigwedge X$.

(iii) If A is chain-complete, then a partial applicative morphism $g\colon A \rightharpoonup B$ is called *chain-continuous* if, for every chain $X \subseteq A$, we have $g\left(\bigwedge X\right) = \bigcap_{a \in X} g(a)$.

We stress some important aspects of this definition.

Remark 6.2. (i) Since we will not usually be interested in the greatest lower bound of $\emptyset$ (i.e., the top element), chains are non-empty by definition.

(ii) If $g\colon A \rightharpoonup B$ is chain-continuous, then it must preserve the order on the nose. Indeed, if $a' \leq a$, then $X = \{a', a\}$ is a chain with $\bigwedge X = a'$. It follows that $g(a') = g(a') \cap g(a)$, i.e., $g(a') \subseteq g(a)$.

(iii) Of course, Definition 6.1 works just as well if A and B are merely posets, and g is any function $A \rightarrow DB$. But we will only use these notions for PCAs and partial applicative morphisms.

Example 6.3. Every discrete PCA A is chain-complete, since the only chains are singletons. For the same reason, every partial applicative morphism $g\colon A \rightharpoonup B$ is trivially chain-continuous. This shows that our construction will subsume the discrete case.

Example 6.4. Suppose that A is chain-complete, and that $g\colon A \rightarrow B$ is total and single-valued, i.e., we can write $g(a) = \downarrow\{g_0(a)\}$ for a certain *function* $A \rightarrow B$. Then g is chain-continuous if and only if, for each chain $X \subseteq A$, the greatest lower bound of $g_0(X) \subseteq B$ exists and is equal to $g_0(\bigwedge X)$.
From this, it is easy to deduce the following. Suppose that A and B are chain-complete, and that $A \xrightarrow{g} B \xrightarrow{h} C$ are chain-continuous with g projective. Then hg is chain-continuous as well. This does *not* seem to be true if we do not require that g is projective.

The following two lemmata discuss the compatibility between the notions introduced above and the construction $\mathcal{B}A$.

Lemma 6.5. *Let A be a chain-complete PCA. Then $\mathcal{B}A$ is also chain-complete, and $i\colon A \rightarrow \mathcal{B}A$ is chain-continuous.*

Proof. Let $\{\alpha_i \mid i \in I\}$ be a chain in $\mathcal{B}A$. We define its greatest lower bound $\alpha = \bigwedge\{\alpha_i \mid i \in I\}$ as follows. First of all, we set:

$$\operatorname{dom}\alpha = \bigcup_{i \in I} \operatorname{dom}\alpha_i = \{a \in A \mid \exists i \in I\,(\alpha_i(a))\downarrow\}.$$

For $a \in \operatorname{dom}\alpha$, the set $X_a = \{\alpha_i(a) \mid i \in I \text{ and } a \in \operatorname{dom}\alpha_i\}$ is a chain in A. This means we can define $\alpha(a) = \bigwedge X_a$. We leave it to the reader to show that $\alpha \in \mathcal{B}A$, and that α is indeed the greatest lower bound of the α_i.
For the final statement, we observe that, for a chain $X \subseteq A$, we have $\bigwedge_{a \in X} \hat{a} = \widehat{\bigwedge X}$, which suffices by Example 6.4. $\qquad\square$

Lemma 6.6. *Consider a decidable partial applicative morphism $g\colon A \rightharpoonup B$, where A is chain-complete. Let $h\colon \mathcal{B}A \rightharpoonup B$ be the largest partial applicative morphism such that $hi \simeq g$ as in Theorem 5.4. If g is chain-continuous, then h is chain-continuous as well.*

Proof. Suppose that g is chain-continuous. We recall that, since g preserves the order on the nose, h does so as well. If $\{\alpha_i \mid i \in I\}$ is a chain in $\mathcal{B}A$ and $\alpha = \bigwedge_{i \in I} \alpha_i$, then this already implies that $h(\alpha) \subseteq \bigcap_{i \in I} h(\alpha_i)$. So it remains to show the converse inclusion, i.e., if $b \in B$ represents all the α_i w.r.t. g, then b also represents α w.r.t. g.

So suppose that $b \in B$ represents α_i w.r.t. g, for all $i \in I$, and consider an $a \in \operatorname{dom} \alpha$. Then $a \in \operatorname{dom} \alpha_i$ for some i, which implies that $b \cdot g(a)$ is defined. Moreover, for all $i \in I$ such that $a \in \operatorname{dom} \alpha_i$, we have $b \cdot g(a) \subseteq g(\alpha_i(a))$. This yields:

$$b \cdot g(a) \subseteq \bigcap_{\substack{i \in I \\ a \in \operatorname{dom} \alpha_i}} g(\alpha_i(a)) = g\left(\bigwedge\{\alpha_i(a) \mid i \in I \text{ and } a \in \operatorname{dom} \alpha_i\}\right) = g(\alpha(a)),$$

as desired. □

The main point of introducing chain-completeness is that we can perform 'fixpoint constructions' in $\mathcal{B}A$, which is crucial for generalizing the contruction from [FvO16]. This construction is described in the following proposition.

Proposition 6.7. *Let A be a chain-complete PCA and let $F \in \mathcal{B}\mathcal{B}A$ be a total function. Then F has a largest fixpoint in $\mathcal{B}A$.*

Remark 6.8. It may seem strange that we construct a *largest* fixpoint, since recursion theory is usually concerned with *smallest* fixpoints. However, we must keep in mind that 'largest' should be read w.r.t. the ordering on $\mathcal{B}A$ as in Definition 2.2. If A is discrete, then this is the *reverse* subfunction relation, so what we identify as the largest fixpoint would usually indeed be called the smallest fixpoint.

Proof of Proposition 6.7. A total function $F \in \mathcal{B}\mathcal{B}A$ is simply an order-preserving function $\mathcal{B}A \to \mathcal{B}A$. We define, recursively, an ordinal-indexed sequence of element f_γ of $\mathcal{B}A$, as follows:

- $f_0 = \emptyset$;

- $f_{\gamma+1} = F(f_\gamma)$;

- $f_\lambda = \bigwedge_{\kappa < \lambda} f_\kappa$ if $\lambda > 0$ is a limit ordinal.

Using transfinite induction and the fact that F is order-preserving, one may show that $f_\gamma \geq f_\delta$ for $\gamma \leq \delta$, and that the sequence is well-defined. By cardinality considerations, the sequence must stabalize at some point, i.e., there exists an ordinal ζ such that $F(f_\zeta) = f_\zeta$. Then f_ζ is a fixpoint of f. Moreover, if $f' \in \mathcal{B}A$ is an element satisfying $f' \leq F(f')$, then by transfinite induction, it easily follows that $f' \leq f_\gamma$ for every ordinal γ, and in particular, $f' \leq f_\zeta$. We conclude that f_ζ is the largest fixpoint of F. □

Proposition 6.9. *Consider a decidable partial applicative morphism $g \colon A \rightharpoonup B$, such that A is chain-complete and g is chain-continuous. Let $F \in \mathcal{B}\mathcal{B}A$ be a total function, and let $h \colon \mathcal{B}A \rightharpoonup B$ be the largest partial applicative morphism such that $hi \simeq g$. If F is effectively representable w.r.t. h, then the largest fixpoint of F is effectively representable w.r.t. g.*

Proof. Let $\mathsf{z} \in B^\#$ be the guarded fixpoint operator. We will show that, if $r \in B$ represents F w.r.t. h, then $\mathsf{z}r$ (which is always defined) represents the largest fixpoint of F w.r.t. g. Clearly, this implies the proposition, for we have $\mathsf{z}r \in B^\#$ if $r \in B^\#$.

So suppose that $r \in B$ represents F w.r.t. h. Define the sequence f_γ as in the proof of Proposition 6.7, so that the largest fixpoint of F is f_ζ for some ordinal ζ. We will show, using transfinite induction, that $\mathsf{z}r \in h(f_\gamma)$ for all ordinals γ. In particular, we will have $\mathsf{z}r \in h(f_\zeta)$, which means that $\mathsf{z}r$ represents f_ζ w.r.t. g, as desired.

First of all, we have $h(f_0) = h(\emptyset) = B$, so the base case is trivial. Now suppose that $\mathsf{z}r \in h(f_\gamma)$ for a certain ordinal γ. Then $r(\mathsf{z}r) \in h(F(f_\gamma)) = h(f_{\gamma+1})$, i.e., $r(\mathsf{z}r)$ represents $f_{\gamma+1}$ w.r.t. g. Now, since $\mathsf{z}rb \preceq r(\mathsf{z}r)b$ for all $b \in B$, it follows that $\mathsf{z}r$ also represents $f_{\gamma+1}$ w.r.t. g, i.e., $\mathsf{z}r \in h(f_{\gamma+1})$. Finally, Lemma 6.6 tells us that h is chain-continuous, from which the limit case immediatelt follows. This completes the induction. $\qquad\square$

Now let us introduce the type-2 functionals that we want to adjoin to a PCA A.

Definition 6.10. Let A be a PCA.

(i) The set $\mathcal{B}_2 A$ is defined as the set of all partial functions $\mathcal{B}A \rightharpoonup A$ such that $\alpha \leq \beta$ implies $F(\alpha) \preceq F(\beta)$ for all $\alpha, \beta \in \mathcal{B}A$.

(ii) For $F, G \in \mathcal{B}_2 A$, we say that $F \leq G$ if $F(\alpha) \preceq G(\alpha)$ for all $\alpha \in \mathcal{B}A$.

Remark 6.11. Observe that, even for discrete PCAs A, the set $\mathcal{B}_2 A$ does not consist of all partial functions $\mathcal{B}A \rightharpoonup A$. Indeed, in the discrete case, we have that $F \in \mathcal{B}_2 A$ if and only if the following holds: whenever α is a subfunction of β and $F(\alpha)\downarrow$, we have that $F(\beta)$ is defined as well and equal to $F(\alpha)$. One may view this as an 'extensionality' requirement: if $F(\alpha)$ is defined, then this must be based solely on the values specified by α. Note, however, that we do not require that $F(\alpha)\downarrow$ implies that the value of $F(\alpha)$ is already determined by *finitely many* values of α. In other words, it is possible that F 'consults' infinitely many values of α. The paper [FvO16] shows how to adjoin partial functions $A^A \rightharpoonup A$, which are always in $\mathcal{B}_2 A$ (if A is discrete).

Definition 6.12. Let A be a PCA and let $F \in \mathcal{B}_2 A$.

(i) We say that $r \in A$ *represents* F if: whenever $\alpha \in \operatorname{dom} F$ and $a \in A$ represents α, we have $ra \leq F(\alpha)$. We say that F is representable (resp. effectively representable) if F is represented by some $r \in A$ (resp. $r \in A^\#$).

(ii) If $g \colon A \rightharpoonup B$ is a partial applicative morphism, then we say that $s \in B$ *represents* F w.r.t. g if: whenever $\alpha \in \operatorname{dom} F$ and $b \in B$ represents α w.r.t. g, we have $sb \in g(F(\alpha))$. We say that F is representable (resp. effectively representable) w.r.t. g if F is represented w.r.t. g by some $s \in B$ (resp. $s \in B^\#$).

Again, the first item is a special case of the second item, by taking $g = \operatorname{id}_A$.

Remark 6.13. The effective representability of second-order functionals is *not* transferable along partial applicative morphisms. Below, we present an instructive example of this phenenomenon. It is also the reason why Theorem 6.15 below cannot be restated as a 'universal property' in the same vein as Corollary 4.10. In this sense, the representability of second-order functionals is categorically ill-behaved when compared to the first-order case.

Example 6.14. Consider Kleene's first model $\mathcal{K}_1$ and let $F \colon \mathbb{N}^\mathbb{N} \to \mathbb{N}$ be defined by:

$$F(f) = \begin{cases} 0 & \text{if } f \text{ is recursive;} \\ 1 & \text{otherwise.} \end{cases}$$

Since the domain of F consists only of total functions, which form a discrete subset of $\mathcal{B}\mathcal{K}_1$, we automatically have $F \in \mathcal{B}_2\mathcal{K}_1$. Moreover, F is effectively representable in $\mathcal{K}_1$. Indeed, the only representable total functions in $\mathcal{K}_1$ are, by definition, the recursive functions, and F is constant on those. However, F is not even representable w.r.t. $i\colon \mathcal{K}_1 \to \mathcal{B}\mathcal{K}_1$, let alone effectively so. Indeed, the representability of F w.r.t. i would imply that the function $\hat{F} \in \mathcal{B}\mathcal{B}\mathcal{K}_1$ given by:

$$\hat{F}(f) = \widehat{F(f)} = \begin{cases} \hat{0} & \text{if } f \text{ is recursive;} \\ \hat{1} & \text{otherwise.} \end{cases}$$

is representable in $\mathcal{B}\mathcal{K}_1$. For, given $f \in \mathcal{B}\mathcal{K}_1$, we can first effectively find a represener of f w.r.t. i, and then use the represener for F w.r.t. i to obtain $\widehat{F(f)} = \hat{F}(f)$. However, $\hat{F}$ cannot be representable in $\mathcal{B}\mathcal{K}_1$. Indeed, suppose that it is represented by $\rho \in \mathcal{B}\mathcal{K}_1$. Then

$$F(f) = \hat{F}(f)(0) = \rho f(0)$$

for all $f\colon \mathbb{N} \to \mathbb{N}$. In particular, $\rho\hat{0}(0) = 0$. This computation consults the oracle $\hat{0}$ only finitely many times, so there exists an N such that $F(f) = \rho f(0) = 0$ for *all* $f\colon \mathbb{N} \to \mathbb{N}$ with $f(n) = 0$ for all $n < N$. This is clearly a contradiction, since this includes non-recursive f. (More generally, the point here is that a represener ρ can consult only finitely many values of an $f\colon \mathbb{N} \to \mathbb{N}$ to determine $\rho f(0)$, but this does not suffice to determine whether f is recursive.)

Now we state and prove the main result of this section.

Theorem 6.15. (*Cf. [FvO16], Theorem 3.1.*) *Let A be a chain-complete PCA and let $F \in \mathcal{B}_2A$. Then there exists an $f \in \mathcal{B}A$ such that:*

(*i*) *F is effectively representable w.r.t. $\iota_f\colon A \to A[f]$;*

(*ii*) *if $g\colon A \rightharpoonup B$ is decidable and chain-continuous, and F is effectively representable w.r.t. g, then g factors, up to isomorphism, through ι_f.*

Proof. Define the total function $\tilde{F} \in \mathcal{B}\mathcal{B}A$ by:

$$\tilde{F}(\alpha)(a) \simeq F(\lambda a'.a \odot_\alpha a').$$

We leave it to the reader to show that $\tilde{F}$ is well-defined and an element of $\mathcal{B}\mathcal{B}A$. Let $f \in \mathcal{B}A$ be the largest fixpoint of $\tilde{F}$, whose existence was asserted in Proposition 6.7.

(i). Suppose that $a \in A$ represents $\alpha \in \operatorname{dom}F$ with respect to ι_f. Then we have that $\lambda a'.a \odot_f a' \leq \alpha$, which implies that $f(a) \simeq F(\lambda a'.a \odot_f a')$ is defined and $f(a) \leq F(\alpha)$. So any represener of f w.r.t. ι_f also represents F w.r.t. ι_f. Since f is, by construction, effectively representable w.r.t. ι_f, it follows that F is effectively representable w.r.t. ι_f as well.

(ii). Let $h\colon \mathcal{B}A \rightharpoonup B$ be the largest partial applicative morphism such that $hi \simeq g$. In order to show that g factors through ι_f, it suffices to show that f is effectively representable w.r.t. g. According to Proposition 6.9, this follows if we show that $\tilde{F}$ is effectively representable w.r.t. h.

Let $s \in B^{\#}$ be such that $s \cdot h(\alpha) \subseteq g(F(\alpha))$ for all $\alpha \in \operatorname{dom}F$. By employing the usual kind of fixpoint argument, we may construct an $r \in B^{\#}$ such that $r \cdot h(\alpha) \cdot g(a) \cdot g(a') \subseteq g(a \odot_\alpha a')$ whenever $a \odot_\alpha a'\downarrow$. Now we claim that $t := \lambda^* xy.s(\lambda^* z.rxyz) \in B^{\#}$ represents $\tilde{F}$ w.r.t. h. In order to prove this, consider $\alpha \in \mathcal{B}A$ and $a \in A$ such that $\tilde{F}(\alpha)(a)$ is defined, and take $b \in h(\alpha)$ and $c \in g(a)$. Moreover, consider $a' \in A$ such that $a \odot_\alpha a'$ is defined, and

29

$c' \in g(a')$. Then we have that $((\lambda^* z.rxyz)[b/x, c/y]) \cdot c' \preceq rbcc' \in g(a \odot_\alpha a')$, which means that $(\lambda^* z.rxyz)[b/x, c/y]$ represents $\lambda a'.a \odot_\alpha a'$. This implies that

$$tbc \preceq s \cdot ((\lambda^* z.rxyz)[b/x, c/y]) \in g(F(\lambda a'.a \odot_\alpha a')) = g\left(\tilde{F}(\alpha)(a)\right).$$

We can conclude that tb represents $\tilde{F}(\alpha)$ w.r.t. g, in other words, that $tb \in h(\tilde{F}(\alpha))$, as desired. □

If A is chain-complete and $F \in \mathcal{B}_2 A$, then we will denote $A[f]$ and ι_f constructed above by $A[F]$ and ι_F, respectively.

We close this section with the counterexample announced at the end of Example 5.6. It shows that, once we add an oracle to $\mathcal{B}A$, elements of $\mathcal{B}A$ and their representers can start to behave very differently. This is a serious obstruction to studying higher-order computability on A by means of $\mathcal{B}A$.

Example 6.16. As in Example 6.14, we let A be Kleene's first model $\mathcal{K}_1$. Consider the *Kleene functional* $E: \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$ defined by:

$$E(f) = \begin{cases} 0 & \text{if } \forall n \in \mathbb{N}(f(n) = 0); \\ 1 & \text{if } \exists n \in \mathbb{N}(f(n) > 0), \end{cases}$$

which is in $\mathcal{B}_2 \mathcal{K}_1$ since its domain consists of total functions. Also, as in Example 6.14, we define $\hat{E} \in \mathcal{B}\mathcal{B}\mathcal{K}_1$ by $\hat{E}(f) = \widehat{E(f)}$ for $f: \mathbb{N} \to \mathbb{N}$. We will now proceed to show the following things about the composition

$$\mathcal{K}_1 \xrightarrow{i} \mathcal{B}\mathcal{K}_1 \xrightarrow{\iota_{\hat{E}}} \mathcal{B}\mathcal{K}_1[\hat{E}].$$

1. Every total function $\mathbb{N} \to \mathbb{N}$ which is representable w.r.t. $\iota_{\hat{E}} \circ i$ is arithmetical.

2. The composition $\iota_{\hat{E}} \circ i$ does *not* factor through $\iota_E: \mathcal{K}_1 \to \mathcal{K}_1[E]$.

3. The second-order functional E is *not* effectively representable w.r.t. $\iota_{\hat{E}} \circ i$.

4. The morphism $\iota_{\hat{E}}$ is *not* the largest partial applicative morphism $h: \mathcal{B}\mathcal{K}_1 \rightharpoonup \mathcal{B}\mathcal{K}_1[\hat{E}]$ such that $hi \simeq \iota_{\hat{E}} \circ i$.

For the first claim, we first observe the following: since the range of $\hat{E}$ is simply $\{\hat{0}, \hat{1}\} \subseteq (\mathcal{B}\mathcal{K}_1)^\#$, we have $\left(\mathcal{B}\mathcal{K}_1[\hat{E}]\right)^\# = (\mathcal{B}\mathcal{K}_1)^\#$, which is just the set of partial recursive functions. Now, if $\alpha, \beta, \gamma \in \mathcal{B}\mathcal{K}_1$ are partial functions, then the relation '$\alpha\beta = \gamma$' can be expressed arithmetically in terms of the graphs of α, β and γ, as is immediate from the definition of application in $\mathcal{B}\mathcal{K}_1$. Moreover, if $\alpha \in \mathcal{B}\mathcal{K}_1$ and $i \in \mathbb{N}$, then $\hat{E}(\alpha) = \hat{i}$ can be expressed arithmetically in terms of i and the graph of α. And we know that all the combinators in $\mathcal{B}\mathcal{K}_1$ are partial recursive functions, and therefore have arithmetically expressible graphs. Finally, we assume for simplicity that the booleans in $(\mathcal{B}\mathcal{K}_1)^\#$ are simply $\hat{0}$ and $\hat{1}$, which we can, because $i: \mathcal{K}_1 \to \mathcal{B}\mathcal{K}_1$ is decidable. Now suppose that $f: \mathbb{N} \to \mathbb{N}$ is represented w.r.t. $\iota_{\hat{E}} \circ i$ by $\rho \in \left(\mathcal{B}\mathcal{K}_1[\hat{E}]\right)^\# = (\mathcal{B}\mathcal{K}_1)^\#$. Then ρ is partial recursive, and since f is total, we have $f(a) = b$ iff $\rho \odot_{\hat{E}} \hat{a} = \hat{b}$. We see that $\rho \odot_{\hat{E}} \hat{a} = \hat{b}$ holds iff there exists a coded sequence $u = [u_0, \ldots, u_{n-1}]$ of *natural numbers*, such that:

- for all $i < n$, we have $\mathsf{p}_0(\rho \cdot [\hat{a}, \hat{u}_0, \ldots, \hat{u}_{i-1}]) = \hat{0}$ and $\hat{E}(\mathsf{p}_1(\rho \cdot [\hat{a}, \hat{u}_0, \ldots, \hat{u}_{i-1}])) = \hat{u}_i$, and:

- $\mathsf{p}_0(\rho \cdot [\hat{a}, \hat{u}_0, \ldots, \hat{u}_{n-1}]) = \hat{1}$ and $\mathsf{p}_1(\rho \cdot [\hat{a}, \hat{u}_0, \ldots, \hat{u}_{n-1}]) = \hat{b}$.

By the remarks above, this is an arithmetical relation in terms of a and b, so we see that '$f(a) = b$' is arithmetical, as desired.

The paper [FvO16] shows that, when Theorem 6.15 is applied to Kleene's first model, the result is equivalent to Kleene's original notion of computability w.r.t. a higher-order functional. In particular, the partial functions $\mathbb{N} \rightharpoonup \mathbb{N}$ which are effectively representable w.r.t. ι_E are precisely the *hyper*arithmetical functions ([FvO16], Corollary 4.1). Now, if $\iota_{\hat{E}} \circ i$ were to factor through ι_E, then every hyperarithmetical functions would be effectively representable w.r.t. $\iota_{\hat{E}} \circ i$ as well. But claim 1 tells us that this is not the case, since there are certainly total hyperarithmetical functions which are not arithmetical (e.g., the characteristic function of a hyperarithemtical set which is not arithmetical).

Claim 3 immediately follows from claim 2 by Theorem 6.15.

Finally, in order to prove claim 4, suppose for the sake of contradiction that $\sigma \in (\mathcal{BK}_1)^\#$ realizes the inequality $h \leq \iota_{\hat{E}}$, where $h(\alpha) = \{\beta \in \mathcal{BK}_1 \mid \beta \text{ represents } \alpha \text{ w.r.t. } \iota_{\hat{E}} \circ i\}$. Moreover, let $\rho \in (\mathcal{BK}_1)^\#$ represent $\hat{E}$ w.r.t. $\iota_{\hat{E}}$. Then it easily follows that

$$\lambda^* x. \rho \odot_{\hat{E}} (\sigma \odot_{\hat{E}} x) \in (\mathcal{BK}_1)^\#$$

represents E w.r.t. $\iota_{\hat{E}} \circ i$, which is not the case by claim 3.

7 The third-order case

In this final section, we investigate what can be achieved in the case of *third*-order functionals. First, let us introduce the objects considered in this section.

Definition 7.1. Let A be a PCA.

(i) The set $\mathcal{B}_3 A$ consists of all partial functions $\Phi \colon \mathcal{B}_2 A \rightharpoonup A$ such that $\Phi(F) \preceq \Phi(G)$ whenever $F \leq G$.

(ii) We say that $r \in A$ *represents* $\Phi \in \mathcal{B}_3 A$ if: whenever $F \in \operatorname{dom} \Phi$ and $a \in A$ represents F, we have $ra \leq \Phi(F)$. We say that Φ is representable (resp. effectively representable) if Φ is represented by some $r \in A$ (resp. $r \in A^\#$).

(iii) If $g \colon A \rightharpoonup B$ is a partial applicative morphism, then we say that $s \in B$ *represents* Φ w.r.t. g if: whenever $F \in \operatorname{dom} \Phi$ and $b \in B$ represents F w.r.t. g, we have $sb \in g(\Phi(F))$. We say that Φ is representable (resp. effectively representable) w.r.t. g if Φ is represented w.r.t. g by some $s \in B$ (resp. $s \in B^\#$).

As the authors of [FvO16] mention as well, there is a fundamental obstacle when studying the representability of third-order functionals. A PCA A can only 'talk about' first-order functions by means of their representers. So as far as A is concerned, only representable functions really exist. For representing second-order functionals F, this is not a problem. On the contrary, is makes the task easier: if a first-order function α does not have a representer, then we do not have to worry about α when constructing a representer for F. In other words, from the point of view of A, the domain of F may look smaller than it actually is.

For a third-order functional Φ, on the other hand, the second-order functionals serve as *inputs*, and A may lose information contained in these inputs. More precisely, there may be two distinct $F, G \in \mathcal{B}_2 A$ such that $\Phi(F)$ and $\Phi(G)$ do not have a common lower bound, but which are the same from the point of view of A. Let us give an example of this phenemenon.

Example 7.2. Again, we let A be Kleene's first model $\mathcal{K}_1$. Consider the third-order functional $\Phi \in \mathcal{B}_3 \mathcal{K}_1$ defined by $\operatorname{dom} \Phi = \{F \in \mathcal{B}_2 \mathcal{K}_1 \mid \mathbb{N}^{\mathbb{N}} \subseteq \operatorname{dom} F\}$ and

$$\Phi(F) = \begin{cases} 0 & \text{if } \forall f \in \mathbb{N}^{\mathbb{N}} (F(f) = 0); \\ 1 & \text{if } \exists f \in \mathbb{N}^{\mathbb{N}} (F(f) > 0). \end{cases}$$

We leave it to the reader to check that Φ is actually in $\mathcal{B}_3 \mathcal{K}_1$. Recall from Example 6.14 the functional $F \colon \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$ defined by

$$F(f) = \begin{cases} 0 & \text{if } f \text{ is recursive}; \\ 1 & \text{otherwise.} \end{cases}$$

Consider also the function $G \in \mathcal{B}_2 \mathcal{K}_1$, which is 0 on total functions, and undefined on non-total functions. Then we clearly have $\Phi(F) = 1 \neq 0 = \Phi(G)$. However, from the point of view of $\mathcal{K}_1$, the functionals F and G are equal, and both are represented by an index for the constant 0 function.

This does not yet exclude the possibility that, as for the second-order case, we can construct a partial function $g \colon \mathbb{N} \rightharpoonup \mathbb{N}$ such that Φ becomes representable w.r.t. $\iota_g \colon \mathcal{K}_1 \to \mathcal{K}_1[g]$. However, we can adjust the example above to show that this is impossible as well. Define $F_g \colon \mathbb{N}^{\mathbb{N}} \to \mathbb{N}$ by:

$$F_g(f) = \begin{cases} 0 & \text{if } f \text{ is representable w.r.t. } \iota_g; \\ 1 & \text{otherwise.} \end{cases}$$

Once again, we have $\Phi(F_g) = 1 \neq 0 = \Phi(G)$, but $A[g]$ cannot distinguish between F_g and G. This means that Φ is not representable (let alone *effectively* representable) w.r.t. to ι_g for *any* partial function g.

This example shows that a construction as in the previous section, where $A[F]$ was of the form $A[f]$ for an $f \in \mathcal{B}A$, simply cannot work in the third-order case. On the other hand, the example above would clearly be blocked if we move to a PCA in which every (partial) function is representable. Fortunately, we have such a PCA, namely $\mathcal{B}A$, which allows us to prove the following 'lax' result about the third-order case.

Theorem 7.3. *Let A be a chain-complete PCA and let $\Phi \in \mathcal{B}_3 A$. Then there exists a c.d. total applicative morphism $\iota_\Phi \colon A \to A[\Phi]$ such that:*

(i) *Φ is effectively representable w.r.t. ι_Φ;*

(ii) *if $g \colon A \rightharpoonup B$ is decidable and chain-continuous, and Φ is effectively representable w.r.t. g, then there exists a largest $h \colon A[\Phi] \rightharpoonup B$ such that $h\iota_\Phi \simeq g$.*

Proof. Define $\tilde{\Phi} \in \mathcal{B}_2 \mathcal{B}A$ by

$$\tilde{\Phi}(F)(a) \simeq \Phi(\lambda\alpha.F(\alpha)(\mathsf{i})) \quad \text{for } F \in \mathcal{B}\mathcal{B}A \text{ and } a \in A.$$

Observe that if $\tilde{\Phi}(F)$ is defined, then it is a constant function. We leave it to the reader to check that $\tilde{\Phi}$ is actually in $\mathcal{B}_2\mathcal{B}A$. As before, if $F \in \mathcal{B}_2 A$, then we define $\hat{F} \in \mathcal{B}\mathcal{B}A$ by $\hat{F}(\alpha) \simeq \widehat{F(\alpha)}$. Since

$$\tilde{\Phi}(\hat{F})(a) \simeq \Phi(\lambda\alpha.\hat{F}(\alpha)(\mathsf{i})) \simeq \Phi(\lambda\alpha.F(\alpha)) \simeq \Phi(F),$$

we have $\tilde{\Phi}(\hat{F}) \simeq \widehat{\Phi(F)}$ for all $F \in \mathcal{B}_2 A$.

We will show that $\iota_\Phi \colon A \to A[\Phi]$ can be taken to be the composition:

$$A \xrightarrow{\quad i \quad} \mathcal{B}A \xrightarrow{\quad \iota_{\tilde{\Phi}} \quad} \mathcal{B}A[\tilde{\Phi}]$$

For the sake of readbility, we will just write ι for $\iota_{\tilde{\Phi}}$, and we write $\odot$ for the application in $\mathcal{B}A[\tilde{\Phi}]$.

(i) By construction, $\tilde{\Phi}$ is representable w.r.t. ι by means of a $\rho \in (\mathcal{B}A[\tilde{\Phi}])^{\#}$. By Theorem 5.4, we also know that there exists a $\sigma \in (\mathcal{B}A[\tilde{\Phi}])^{\#}$ such that $\sigma \odot \alpha$ represents α w.r.t. $\iota \circ i$ for all $\alpha \in \mathcal{B}A$. We will show that

$$\tau = \lambda^*x.\rho \odot (\lambda^*y.x \odot (\sigma \odot y)) \in (\mathcal{B}A[\tilde{\Phi}])^{\#}$$

represents Φ w.r.t. $\iota \circ i$. So let $F \in \mathcal{B}_2 A$ be such that $\Phi(F)$ is defined, and suppose that $\beta \in \mathcal{B}A$ represents F w.r.t. $\iota \circ i$. First of all, we claim that $(\lambda^*y.x \odot (\sigma \odot y))[\beta/x]$ represents $\hat{F}$ w.r.t. ι. In order to show this, let $\alpha \in \mathcal{B}A$ be such that $\hat{F}(\alpha)$ is defined. Then:

$$(\lambda^*y.x \odot (\sigma \odot y))[\beta/x] \odot \alpha \preceq \beta \odot (\sigma \odot \alpha) \preceq \widehat{F(\alpha)} = \hat{F}(\alpha),$$

since $\sigma \odot \alpha$ represents α w.r.t. $\iota \circ i$ and β represents F w.r.t. $\iota \circ i$. This proves the claim, and it follows that

$$\tau \odot \beta \preceq \rho \odot ((\lambda^*y.x \odot (\sigma \odot y))[\beta/x]) \preceq \tilde{\Phi}(\hat{F}) = \widehat{\Phi(F)},$$

as desired.

(ii) Suppose that $s \in B^{\#}$ represents Φ w.r.t. g, and consider $h \colon \mathcal{B}A \rightharpoonup B$ defined by $h(\alpha) = \{b \in B \mid b \text{ represents } \alpha \text{ w.r.t. } g\}$. Since h is also decidable and chain-continuous, it suffices to show that $\tilde{\Phi} \in \mathcal{B}_2\mathcal{B}A$ is representable w.r.t. h. Then Theorem 6.15 tells us that h is also a morphism $\mathcal{B}A[\tilde{\Phi}] \rightharpoonup B$, and Theorem 5.4 implies that this is the largest partial applicative morphism by means of which g factors through $\iota \circ i$.

We will show that

$$t = \lambda^*x.\mathsf{k}(s(\lambda^*y.xyj)) \in B^{\#}$$

represents $\tilde{\Phi}$ w.r.t. h, where j is any element from $g(\mathsf{i}) \cap B^{\#}$. So let $F \in \mathcal{B}\mathcal{B}A$ be such that $\tilde{\Phi}(F)$ is defined, and suppose that $b \in B$ represents F w.r.t. h. First of all, we claim that $(\lambda^*y.xyj)[b/x]$ represents $\lambda\alpha.F(\alpha)(\mathsf{i})$ as a second-order functional w.r.t. g. In order to prove the claim, let $\alpha \in \mathcal{B}A$ be such that $F(\alpha)(\mathsf{i})$ is defined, and let $c \in B$ represent α w.r.t. g. Then $c \in h(\alpha)$, which means that bc is defined and in $h(F(\alpha))$, i.e., bc represents $F(\alpha)$ w.r.t. g. Since $j \in g(\mathsf{i})$, this yields that $((\lambda^*y.xyj)[b/x]) c \preceq bcj$ is defined and an element of $g(F(\alpha)(\mathsf{i}))$, which proves the claim. Now we find that $s((\lambda^*y.xyj)[b/x]) \in \Phi(\lambda\alpha.F(\alpha)(\mathsf{i}))$, so it follows that $tb \preceq \mathsf{k}(s((\lambda^*y.xyj)[b/x]))$ is defined and represents $\tilde{\Phi}(F)$. In other words, we have $tb \in h(\tilde{\Phi}(F))$, as desired. $\square$

In Section 6, we investigated the possibility of making an $F \in \mathcal{B}_2 A$ effectively representable by adjoining $\hat{F} \in \mathcal{B}\mathcal{B}A$ to $\mathcal{B}A$. It turned out that this will not work in general, the Kleene

functional $E \in \mathcal{B}_2\mathcal{K}_1$ being a counterexample. Therefore, it may seem strange that a similar strategy *does* work for the third-order case! Let us explain why this is so. In the second-order case, the task was to construct, given a representer of $\hat{F}$, a representer of F. Now, a representer of F eats representers of $\alpha \in \mathcal{B}A$, whereas a representer of $\hat{F}$ wants to eat α itself. So the task really is to effectively find, given a representer of α, the function α itself so that it can be fed to the representer of $\hat{F}$. But the problem is exactly that, once we add an *oracle* to $\mathcal{B}A$, this is no longer possible in general. On the other hand, the *converse* construction obviously does work, i.e., given a representer of F, we can construct a representer of $\hat{F}$ (see also Example 6.14). Since we constructed $\tilde{\Phi}$ in such a way that $\tilde{\Phi}(\hat{F}) \simeq \widehat{\Phi(F)}$, this is precisely what we need to construct a representer for Φ, given a representer for $\tilde{\Phi}$. We can also put this as follows: in the business of representing Φ, the representers of second-order functionals $F \in \mathcal{B}_2A$ are not the things to be constructed, but the things that are *given*. Unfortunately, this also reveals that the current strategy can probably not be pushed beyond the third-order case, because in the fourth-order case, things will be 'the wrong way around' again.

We close the paper with an interesting corollary of Theorem 7.3. Suppose we have a decidable partial applicative morphism $g \colon A \rightharpoonup B$ and an $f \in \mathcal{B}A$ which is effectively representable w.r.t. g. Then we know that, besides f and the functions that were already effectively representable in A, other functions must become effectively representable w.r.t. g as well. So a natural question to ask here is: what is the minimal set of elements of $\mathcal{B}A$ that must become effectively representable w.r.t. a partial applicative morphism as soon as f is effectively representable? We can reformulate the question as follows. Let $\mathrm{ER}(f)$ denote the class of all decidable partial applicative morphism $g \colon A \rightharpoonup B$ with respect to which f is effectively representable. Then we are interested in the set:

$$\mathrm{FOE}(f) := \bigcap_{g \in \mathrm{ER}(f)} \{\alpha \in \mathcal{B}A \mid \alpha \text{ is effectively representable w.r.t. } g\},$$

where FOE should be read as 'first-order effect'. The construction from Section 4 tells us what this set is: indeed, we know that $\iota_f \colon A \to A[f]$ is in $\mathrm{ER}(f)$ and that every element of $\mathrm{ER}(f)$ factors through ι_f. This means that $\mathrm{FOE}(f)$ is simply the set of $\alpha \in \mathcal{B}A$ that are effectively representable w.r.t. ι_f, which we know to be $\langle(\mathcal{B}A)^{\#} \cup \{f\}\rangle$.

As we know, the representability of second-order functionals does not behave as nicely from the point of view of pPCA. But we can still ask, given an $F \in \mathcal{B}_2A$, which *first-order* functions are forced to become effectively representable if F is effectively representable. For simplicity, we will assume that A is discrete, so that we do not have to worry about chain-completeness and chain-continuity. As above, we let $\mathrm{ER}(F)$ denote the class of all decidable partial applicative morphisms $g \colon A \rightharpoonup B$ such that F is effectively representable w.r.t. g, and we write

$$\mathrm{FOE}(F) := \bigcap_{g \in \mathrm{ER}(F)} \{\alpha \in \mathcal{B}A \mid \alpha \text{ is effectively representable w.r.t. } g\},$$

for the 'first-order effect' of F. Then $\mathrm{FOE}(F)$ is also of the form $\langle(\mathcal{B}A)^{\#} \cup \{f\}\rangle$ for some $f \in \mathcal{B}A$, namely, the f as in Theorem 6.15.

For the third-order case, we can pose the analogous question. Let A be a discrete PCA and consider $\Phi \in \mathcal{B}_3A$. Let $\mathrm{ER}(\Phi)$ denote the class of all decidable partial applicative morphisms $g \colon A \rightharpoonup B$ such that Φ is effectively representable w.r.t. g, and we write

$$\mathrm{FOE}(\Phi) := \bigcap_{g \in \mathrm{ER}(\Phi)} \{\alpha \in \mathcal{B}A \mid \alpha \text{ is effectively representable w.r.t. } g\},$$

for the 'first-order effect' of Φ. Then Theorem 7.3 tells us that $\mathrm{FOE}(\Phi)$ consists of all $\alpha \in \mathcal{B}A$ that are effectively representable w.r.t. ι_Φ. We can now ask: is this set also of the form $\langle(\mathcal{B}A)^\# \cup \{f\}\rangle$ for some $f \in \mathcal{B}A$? In order to show that the answer is 'yes', we first need the following definition.
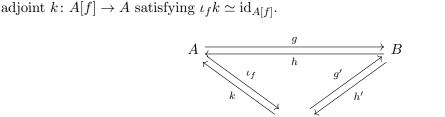
Definition 7.4. Let A be a discrete PCA and let $g\colon A \rightharpoonup B$ be a partial applicative morphism. Then g is called *discrete* if $g(a) \cap g(a') = \emptyset$ for every two *distinct* $a, a' \in A$.

The following result is based on Theorem 2.12 of [FvO14].

Lemma 7.5. *Let A be a discrete PCA and let $g\colon A \to B$ be a* total *applicative morphism which is discrete, projective and c.d. Then there exists an $f \in \mathcal{B}A$ such that:*

$$\{\alpha \in \mathcal{B}A \mid \alpha \text{ is effectively representable w.r.t. } g\} = \langle(\mathcal{B}A)^\# \cup \{f\}\rangle.$$

Proof. Since g is total, projective and c.d., Theorem 3.13(i) tells us that g has a right adjoint $h\colon B \rightharpoonup A$ in pPCA. We claim that the partial applicative morphism $hg\colon A \rightharpoonup A$ is discrete. Suppose we have $a, a' \in A$ such that $hg(a) \cap hg(a')$ is nonempty. Since g preserves the order up to a realizer, it follows that $ghg(a) \cap ghg(a')$ is also nonempty. Since $g \dashv h$, we have $ghg \simeq g$, so this implies that $g(a) \cap g(a')$ is nonempty, so $a = a'$ by the discreteness of g. Now define the partial function $f\colon A \rightharpoonup A$ by: $f(a) = a'$ if and only if $a \in hg(a')$, which is well-defined by the discreteness of hg. Then $g(a) \subseteq ghg(f(a))$, so if $s \in B^\#$ realizes $ghg \leq g$, then s also represents f w.r.t. g. This means that g factors through ι_f by means of a $g'\colon A[f] \to B$, which is defined simply by $g'(a) = g(a)$ for $a \in A$. Now g' is total, projective and c.d. as well, so it has a right adjoint $h'\colon B \rightharpoonup A$. Moreover, we recall that ι_f has a right adjoint $k\colon A[f] \to A$ satisfying $\iota_f k \simeq \mathrm{id}_{A[f]}$.

Since $g \simeq g'\iota_f$, we also have $h \simeq kh'$, hence $\iota_f h \simeq \iota_f k h' \simeq h'$. This means we can assume without loss of generality that $h'(b) = h(b)$ for all $b \in B$. In particular, $h'g'(a) = hg(a)$ for all $a \in A$. But now it is clear that any representer $r \in (A[f])^\#$ of f will also realize the inquality $h'g' \leq \mathrm{id}_{A[f]}$. Combining this with $g' \dashv h'$ yields $g'h' \simeq \mathrm{id}_{A[f]}$. This implies that $\alpha \in \mathcal{B}A$ is effectively representable w.r.t. $g \simeq g'\iota_f$ if and only if α is effectively representable w.r.t. ι_f; if and only if $\alpha \in \langle(\mathcal{B}A)^\# \cup \{f\}\rangle$. $\square$

Remark 7.6. Observe that the proof of Lemma 7.5 also implies that the image topos of the geometric morphism $\mathsf{RT}(B) \to \mathsf{RT}(A)$ induced by g is also a realizability topos, namely $\mathsf{RT}(A[f])$.

Corollary 7.7. *Let A be a discrete PCA and let $\Phi \in \mathcal{B}_3 A$. Then there exists an $f \in \mathcal{B}A$ such that $\mathrm{FOE}(\Phi) = \langle(\mathcal{B}A)^\# \cup \{f\}\rangle$.*

Proof. This follows from Lemma 7.5 if we can show that ι_Φ is total, discrete, projective and c.d. All of these are easy to check. $\square$

References

[FvO14] E. Faber and J. van Oosten. More on geometric morphisms between realizability toposes. *Theory and Applications of Categories*, 29(30):874–95, 2014.

[FvO16] E. Faber and J. van Oosten. Effective operations of type 2 in pcas. *Computability*, 5(2):127–45, 2016.

[HvO03] P. Hofstra and J. van Oosten. Ordered partial combinatory algebras. *Math. Proc. Camb. Phil. Soc.*, 134(3):445–463, 2003.

[Joh13] P. T. Johnstone. Geometric morphisms of realizability toposes. *Theory and Applications of Categories*, 28(9):241–249, 2013.

[LN15]   J. Longley and D. Normann. *Higher-Order Computability*. Springer-Verlag Berlin Heidelberg, 2015.

[Lon94] J. Longley. *Realizability Toposes and Language Semantics*. PhD thesis, University of Edinburgh, 1994.

[vO06]   J. van Oosten. A general form of relative recursion. *Notre Dame Journal of Formal Logic*, 47(3):311–18, 2006.

[vO08]   J. van Oosten. *Realizability: An Introduction to its Categorical Side*, volume 152 of *Studies in Logic and the Foundations of Mathematics*. Elsevier, 2008.

[vO11]   J. van Oosten. Partial combinatory algebras of functions. *Notre Dame Journal of Formal Logic*, 52(4):431–48, 2011.

[Zoe20] J. Zoethout. Internal partial combinatory algebras and their slices. *Theory and Applications of Categories*, 35(52):1907–1952, 2020.