Theoretical Computer Science 206 (1998) 317–329

Theoretical
Computer Science

Unary PCF is decidable

Ralph Loader *,[1]

Paradigm Technology, 79 Boulcott St., Wellington, New Zealand

Received November 1995; revised October 1996
Communicated by G.D. Plotkin

Abstract

We show that *unary PCF*, a very small fragment of Plotkin's PCF [7], has a decidable observational pre-order, and that its fully abstract model is effectively presentable. This is in marked contrast to larger fragments, where corresponding results fail [4]. The techniques used are adaptions of those of Padovani [5], who applied them to the minimal model of the simply typed lambda calculus. © 1998—Elsevier Science B.V. All rights reserved

Keywords: Lambda calculus; PCF; Decidability; Definability

1. Introduction

Plotkin's PCF [7] is a typed lambda calculus, with natural numbers and basic numerical operations, and recursion at arbitrary types. As this calculus can express arbitrary recursive functions, in general, questions about this calculus will obviously tend to be undecidable.

If we restrict the calculus to have only finitely many values, instead of all natural numbers, at ground type, then we obtain a calculus – finitary PCF – for which decidability questions are more sensible. Jung and Stoughton [2] raised the question of whether, or not, the observational pre-order of finitary PCF is decidable, in the hope that a sufficiently good construction of the fully abstract model of PCF would enable this order to be read off in an effective manner. However, in [4] it is shown that ordering is undecidable.

This article is concerned with an even smaller fragment of PCF, namely *unary PCF*, where we have only one value at ground type (an alternative presentation could have multiple values, but no means within the calculus of distinguishing them). Here, in marked contrast to finitary PCF, we show that the observational pre-order of unary

* E-mail: ralph.loader@paradigm.co.nz.

[1] This work was carried out while the author was at Merton College, Oxford.

PCF is decidable. Our proof is not by a semantical construction of the fully abstract-model, the proof technique envisaged by Jung and Stoughton, but rather by a mostly syntactical argument, involving fairly technical manipulations of terms.

The argument is an adaption of one invented by Padovani, who showed [5] that the minimal model of the simply typed lambda calculus has an effective presentation. By deciding the solvability of certain equations in the calculus, we can effectively construct a presentation of the appropriate model of the calculus; and then by working within the model, we can calculate the observational ordering.

The question that this paper answers was asked by Kahrs [3]. He showed that the decidability of the "variable containment problem" reduces to the result of this paper. The variable containment problem is, given some higher order rewriting relation, to decide whether or not rewriting can produce new free variables.

Some notational conventions. We use typewriter style lower case letters $(\mathtt{x}, \mathtt{y}, \mathtt{z}, \ldots)$ to range over variables of lambda calculi. Italic letters $(A,\ B,\ x,\ y\ldots)$, when used as syntactical entities, are used as meta-variables ranging over terms.

We shall use both vectors $A = \langle A^i \rangle_{i \in I}$ and sequences $\bar{t} = t_1 \ldots t_n$. Operations on vectors are generally component-wise, while operations on sequences are repeated: $FA = \langle FA^i \rangle_{i \in I}$, but $g\bar{t} = g t_1 \ldots t_n$. Note that A_j (or $\bar{\mathtt{y}}_j$) denotes the jth vector (or sequence) in a collection of vectors (or sequences), not the jth component of A (or $\bar{y}$).

2. Preliminaries

Definition 1. *Unary PCF* (PCF_1) is the simply typed lambda calculus, with a single ground type o, constants $\top, \bot : o$, and a binary operator $\wedge : o \Rightarrow o \Rightarrow o$. We always use the infix notation $x \wedge y$ for $\wedge$. In addition to the usual conversions of β-reduction and η-expansion, we give two reductions for $\wedge$:

$$\top \wedge x \longrightarrow x \quad \text{and} \quad \bot \wedge x \longrightarrow \bot.$$

This calculus is Church–Rosser and strongly normalising, as readily follows by the techniques of [1]. We define pre-orders $\leqslant_\sigma$ on the closed terms of each type σ as follows. For $a, b : o$, we put $a \leqslant_o b$ if and only if, if a has normal form $\top$, then so does b. At higher types, we make $\leqslant$ a logical relation:

$$f \leqslant_{\sigma \Rightarrow \tau} g \quad \text{iff} \quad f\, a \leqslant_\tau g\, b \text{ whenever } a \leqslant_\sigma b.$$

An application of the logical relations lemma [6, 9] shows that $\leqslant$ is reflexive at each type. An induction over types easily shows that (a) $f \leqslant_{\sigma \Rightarrow \tau} g$ iff, for all closed $r : \sigma$, $f\, r \leqslant g\, r$, and (b) $\leqslant$ is transitive.

The pre-order $\leqslant$ is the *observational pre-order* (although this is not the usual definition). *Observational equivalence*, $\equiv$, is $(\leqslant) \cap (\leqslant)^{-1}$. The *fully abstract* model arises by taking the quotient of $\equiv$.

In this paper, we shall show that the relation $\leqslant$ is decidable, and that the fully abstract model can be presented effectively. We do this by analysing a class of calculi containing unary PCF. There are two equivalence classes of type o, one containing $\top$ and the other containing $\bot$. These form an idempotent commutative monoid with binary operation $\wedge$ and unit $\top$. We generalise this with the definition below:

Definition 2. Let $(A, \wedge, \top)$ be the free idempotent commutative monoid generated by a finite non-empty set $\mathscr{S}$. The *lambda calculus* $\lambda^{\mathscr{S}}$ *generated by* $\mathscr{S}$ has a single ground type o, each member of $\mathscr{S}$ is a constant of ground type, and we have additional constants $\top : o$, $\wedge : o \Rightarrow o \Rightarrow o$. We do not count the constants as variables; hence the term $\kappa \wedge \top$ is closed for any $\kappa \in \mathscr{S}$.

We denote the set of type σ closed terms of $\lambda^{\mathscr{S}}$ by $\lambda^{\mathscr{S}}_\sigma$.

As the simply typed lambda calculus is strong normalising and Church–Rosser, there is an obvious interpretation of $\lambda^{\mathscr{S}}_o$ in A. Let $\equiv^{\mathscr{S}}$ be the logical relation such that for closed $t_1, t_2 : o$ we have $t_1 \equiv^{\mathscr{S}} t_2$ if and only if t_1 and t_2 have the same interpretation in A, and for $f, g : \sigma \Rightarrow \tau$, we have

$$f \equiv^{\mathscr{S}} g \quad \text{iff} \quad f\,a \equiv^{\mathscr{S}} g\,b \text{ for all } a \equiv b : \sigma.$$

Clearly $\equiv^{\mathscr{S}}$ gives an equivalence relation on the terms of ground type, and it follows immediately that $\equiv^{\mathscr{S}}$ is symmetric at all types. Just as for PCF$_1$, we may use the logical relations lemma to show that $\equiv$ is reflexive and transitive at all types, and thus an equivalence.

We define the model $M^{\mathscr{S}}$ of $\lambda^{\mathscr{S}}$ by letting $M^{\mathscr{S}}_\sigma$ be the quotient of the set $\lambda^{\mathscr{S}}_\sigma$ of closed terms of type σ by the relation $\equiv^{\mathscr{S}}$. This makes $M^{\mathscr{S}}_o$ is isomorphic to the monoid A.

Note that the language of $\lambda^{\{\bot\}}$ is that of PCF$_1$, and $\equiv^{\{\bot\}}$ is just observational equivalence, so that $M^{\{\bot\}}$ is the fully abstract model of PCF$_1$.

If r is a normal member of $\lambda^{\mathscr{S}}_o$, then the equivalence class of r is determined by the set of members of $\mathscr{S}$ that occur in r, and vice versa.

$M^{\mathscr{S}}$ is a term model, and as such has a few sensible properties: the interpretation of a closed term is the equivalence class of that term, and the constants and operations of $\lambda^{\mathscr{S}}$ induce well defined operations on $M^{\mathscr{S}}$.

Definition 3. An *effective presentation* of $M^{\mathscr{S}}$ is a pair of effective functions (m, a), with $m(\tau)$ a finite set for any type τ, and with $(x, y) \mapsto a(\sigma, \tau, x, y)$ a function from $m(\sigma \Rightarrow \tau) \times m(\sigma)$ to $m(\tau)$, such that $M^{\mathscr{S}}$ is isomorphic to m with application a.

For there to be an effective presentation of $M^{\mathscr{S}}$, it is necessary and sufficient for there to be a effective function mapping types σ to the number $|M^{\mathscr{S}}_\sigma|$; necessity is obvious and sufficiency follows from the fact that $\equiv^{\mathscr{S}}$ is co-r.e.

Definition 4. An *interpolation equation* is an equation in the form

$$[X A_1 \ldots A_N = B], \tag{1}$$

where X is a variable of some type $\sigma_1 \Rightarrow \cdots \Rightarrow \sigma_N \Rightarrow o$, each A_i is closed of type σ_i, and B is closed of type o.

A *solution* to an interpolation equation (1) is a closed term which, when substituted for X, makes the equation true up to $\equiv^{\mathscr{S}}$.

An *interpolation system* is a set of interpolation equations all with the same variable:

$$\{[X A_1^i \dots A_N^i = B^i] \mid i \in I\}.$$

It is convenient to use the vector notation $[X \boldsymbol{A}_1 \dots \boldsymbol{A}_N = \boldsymbol{B}]$ for interpolation systems. A *solution* of an interpolation system Φ is a term that is a solution to each member of Φ. Unless otherwise stated, we assume that all interpolation systems are finite.

Lemma 5. 1. *Equivalent (by $\equiv^{\mathscr{S}}$) terms solve exactly the same equations (and systems).*
2. *The solution set of an equation (or system) is unchanged if we replace the A_j^i and B^i by equivalents.*
3. *A member of $M^{\mathscr{S}}$ – i.e., an equivalence class of $\equiv^{\mathscr{S}}$ – is the set of all solutions to some interpolation system Φ. In particular Φ can be taken to be the infinite set of all equations satisfied by a particular member of the equivalence class.*
4. *By limiting the A_j^i and B^i of the system Φ above to a unique choice of representative for each equivalence class of $\equiv^{\mathscr{S}}$, we may take Φ to be finite.*

Proof. 1. Obvious, as is 2.
3. Let r_0 be a closed term of type $\sigma_1 \Rightarrow \cdots \Rightarrow \sigma_N \Rightarrow o$, and let Φ be the set of all interpolation equations satisfied by r_0. If $r \equiv^{\mathscr{S}} r_0$, then r also solves Φ by 1. If r solves Φ, then for any $A_1 : \sigma_1, \dots, A_N : \sigma_N$, we have that $[X A_1 \dots A_N = r_0 A_1 \dots A_N] \in \Phi$, and so $r A_1 \dots A_N \equiv^{\mathscr{S}} r_0 A_1 \dots A_N$; this implies that $r \equiv^{\mathscr{S}} r_0$.
4. Let S_j contain a unique representative of each equivalence class of type σ_j, and let T contain a unique representative of type o.
By 2., the solutions of the system Φ above remains unchanged if we retain only those equations for which the A_j^i and B^i are members of the S_j and T. For the finiteness, we show that $M^{\mathscr{S}}$ is finite at each type, using induction on the type. Clearly, $|M_o^{\mathscr{S}}| = 2^{|\mathscr{S}|}$. A member of $M_{\sigma\Rightarrow\tau}^{\mathscr{S}}$ is uniquely determined by the function it gives from $M_\sigma^{\mathscr{S}}$ to $M_\tau^{\mathscr{S}}$, so that $|M_{\sigma\Rightarrow\tau}^{\mathscr{S}}| \leqslant |M_\tau^{\mathscr{S}}|^{|M_\sigma^{\mathscr{S}}|}$. □

The lemma above shows that there is a close relationship between interpolation systems and the model $M^{\mathscr{S}}$. There is also a close relationship from the point of view of decidability:

Lemma 6. *The model $M^{\mathscr{S}}$ has an effective presentation if and only if there is a procedure for deciding the solvability of interpolation systems (for $\lambda^{\mathscr{S}}$).*

Proof. The "only if" part is obvious by 1. of the previous lemma; given an effective presentation of the model we can search through the presentation for a solution to any given problem.

Conversely, an effective presentation can be built by recursion on types, by solving interpolation systems. We shall find an $m(\sigma)$ that is a set of equivalence class representatives for $\equiv^{\mathscr{S}}$.

Given σ and $m(\sigma_1),\dots,m(\sigma_N)$ where $\sigma = \sigma_1 \Rightarrow \cdots \Rightarrow \sigma_N \Rightarrow o$, to find $m(\sigma)$ we need only find which set-theoretic functions f from $m(\sigma_1) \times \cdots \times m(\sigma_N)$ to $m(o)$ are definable by closed terms. Given such a function f, let the interpolation system Φ be

$$\{[X A_1 \dots A_N = f(A_1,\dots,A_N)] \mid (A_1,\dots,A_N) \in m(\sigma_1) \times \cdots \times m(\sigma_N)\}.$$

Clearly f is definable if and only if the system Φ has a solution. $\square$

Note that this equivalence is uniform in $\mathscr{S}$ and well-behaved w.r.t. the types in question.

3. Usefulness

The algorithm for deciding solvability for interpolation systems Φ will be a divide-and-conquer one. Given an interpolation system, we will search for terms showing the system is 'self-useful' as defined below. Finding such terms allows us to reduce the solvability of Φ to that of certain proper subsets of Φ.

Definition 7. Let $\mathscr{T}$ be a set of constants disjoint from $\mathscr{S}$. Any substitution mapping $\mathscr{T}$ into $\lambda_o^{\mathscr{S}}$ gives a well defined map from $M^{\mathscr{S}\cup\mathscr{T}}$ to $M^{\mathscr{S}}$. A vector $\boldsymbol{E}$ is said to be *useful* for a vector $\boldsymbol{B} \in (M_o^{\mathscr{S}})^I$ if

- $\boldsymbol{E}$ is a vector in $(M_o^{\mathscr{S}\cup\mathscr{T}})^I$ for some $\mathscr{T}$ disjoint from $\mathscr{S}$.
- For each $\kappa \in \mathscr{T}$, there is $i \in I$ such that (some representative of) E^i does not contain κ. (Equivalently, $E^i \not\equiv E^i \wedge \kappa$.)
- For each i, there is a substitution with domain $\mathscr{T}$ and co-domain $\lambda_o^{\mathscr{S}}$ that maps E^i to B^i.

If $\boldsymbol{B} \in (\lambda_o^{\mathscr{S}})^I$ and $\boldsymbol{E} \in (\lambda_o^{\mathscr{S}\cup\mathscr{T}})^I$, then $\boldsymbol{E}$ is *$\mathscr{S}$-useful* for $\boldsymbol{B}$ if and only if the corresponding vectors in the models are useful. If $\mathscr{S}$ is clear from the context, then we will write *useful* rather than $\mathscr{S}$-useful. (Note that $\mathscr{S}$-usefulness does not depend on the choice of $\mathscr{T}$ such that $\boldsymbol{E} \in (\lambda_o^{\mathscr{S}\cup\mathscr{T}})^I$.)

An interpolation system $[X\, \boldsymbol{A}_1 \dots \boldsymbol{A}_N = \boldsymbol{B}]$ with parameters in $\lambda^{\mathscr{S}}$ is *$\mathscr{S}$-self-useful* if there is a vector $\boldsymbol{E}$, $\mathscr{S}$-useful for $\boldsymbol{B}$, that can be expressed in the form

$$\boldsymbol{A}_{i_1} s_{1,1} \dots s_{1,\alpha_1} \wedge \cdots \wedge \boldsymbol{A}_{i_n} s_{n,1} \dots s_{n,\alpha_n}. \tag{2}$$

Again, we will write *self-useful*, if $\mathscr{S}$ is clear from the context.

Below we give a few basic properties of usefulness that are useful later.

Lemma 8. 1. *Suppose that $\boldsymbol{E}$ is useful for $\boldsymbol{B}$. Then, for each i, the substitution mapping E^i to B^i can be taken to be the substitution mapping each $\kappa \in \mathscr{T}$ to a representative of B^i.*

2. *Let $\Phi = [X\, A_1 \ldots A_N = \boldsymbol{B}]$ be an interpolation system, and $\mathscr{T}$ be some set of constants, with the same cardinality as Φ, and disjoint from $\mathscr{S}$. Then Φ is self-useful if and only if there are $s_{i,j} \in \lambda^{\mathscr{S} \cup \mathscr{T}}$ such that (2) is $\mathscr{S}$-useful for $\boldsymbol{B}$. (Informally, this just says that in the definition of self-usefulness, we may take $\mathscr{T}$ to be some fixed set with the same cardinality as the interpolation system.)*
3. *Let $\mathscr{S}$ and $\mathscr{T}$ be disjoint sets of constants. Suppose that $\boldsymbol{E} \in (M_o^{\mathscr{S} \cup \mathscr{T}})^I$ is useful for $\boldsymbol{B} \in (M_o^{\mathscr{S}})^I$, and that $\boldsymbol{C} \in (M_o^{\mathscr{S}})^I$. Then $\boldsymbol{E} \wedge \boldsymbol{C}$ is useful for $\boldsymbol{B} \wedge \boldsymbol{C}$.*

Proof. 1. Consider normal representatives b^i and e^i of B^i and E^i. If no $\kappa \in \mathscr{T}$ occurs in e^i, then $b^i \equiv e^i$, and there is nothing to do.
Otherwise, b^i is a finite conjunction of members of $\mathscr{S} \cup \{\top\}$, and e^i is a finite conjunction of members of $\mathscr{S} \cup \mathscr{T} \cup \{\top\}$, and further, any member of $\mathscr{S}$ occurring in e^i also occurs in b^i. The result is now obvious, considering the idempotence of $\wedge$.
2. Suppose that an interpolation system is self-useful, with the useful vector (2) using constants in the set $\mathscr{T}_0$. Let $\mathscr{T} = \{\mu^i \mid i \in I\}$ be some set of constants indexed by i. For each $\kappa \in \mathscr{T}_0$, choose $i_\kappa \in I$ such that κ does not occur in the normal form of the i_κth component of (2). Using 1. above, it is clear that substituting μ^{i_κ} for each κ in (2) gives us another useful vector.
3. The appropriate substitutions are unchanged. Clearly, no $\kappa \in \mathscr{T}$ can occur in C^i, so if κ does not occur in some representative of E^i, then it does not occur in some representative of $C^i \wedge E^i$. $\square$

Lemma 9. *Usefulness is transitive in the following senses:*
1. *If $\boldsymbol{B} \in (M_o^{\mathscr{S}})^I$, $\boldsymbol{C} \in (M_o^{\mathscr{S} \cup \mathscr{T}})^I$ and $\boldsymbol{D} \in (M_o^{\mathscr{S} \cup \mathscr{T} \cup \mathscr{U}})^I$ are such that $\boldsymbol{C}$ is useful for $\boldsymbol{B}$ and $\boldsymbol{D}$ is useful for $\boldsymbol{C}$, then $\boldsymbol{D}$ is useful for $\boldsymbol{B}$.*
2. *If $\boldsymbol{B} \in (\lambda_o^{\mathscr{S}})^I$, $\boldsymbol{C} \in (\lambda_o^{\mathscr{S} \cup \mathscr{T}})^I$ and $\boldsymbol{D} \in (\lambda_o^{\mathscr{S} \cup \mathscr{T} \cup \mathscr{U}})^I$ are such that $\boldsymbol{C}$ is $\mathscr{S}$-useful for $\boldsymbol{B}$, and $\boldsymbol{D}$ is $(\mathscr{S} \cup \mathscr{T})$-useful for $\boldsymbol{C}$, then $\boldsymbol{D}$ is $\mathscr{S}$-useful for $\boldsymbol{B}$.*
3. *If $A_j \in (\lambda_{\sigma_j}^{\mathscr{S}})^I$ and $\boldsymbol{C} \in (\lambda_o^{\mathscr{S} \cup \mathscr{T}})^I$ are such that $[X\, A_1 \ldots A_N = \boldsymbol{C}]$ is $\mathscr{S} \cup \mathscr{T}$-self-useful, and $\boldsymbol{C}$ is $\mathscr{S}$-useful for $\boldsymbol{B} \in (\lambda_o^{\mathscr{S}})^I$, then $[X\, A_1 \ldots A_N = \boldsymbol{B}]$ is $\mathscr{S}$-self-useful.*

where $\mathscr{S}$, $\mathscr{T}$ and $\mathscr{U}$ are disjoint.

Proof. We show 1., from which 2. and 3. then follow. Appropriate substitutions are constructed by composition: let $\sigma : \mathscr{T} \to \lambda_o^{\mathscr{S}}$ be such that $\sigma C^i = B^i$ and let $\tau : \mathscr{U} \to \lambda_o^{\mathscr{S} \cup \mathscr{T}}$ be such that $\tau D^i = C^i$. Define $\rho : \mathscr{T} \cup \mathscr{U} \to \lambda_o^{\mathscr{S}}$ by

$$\rho\kappa = \sigma\kappa$$

for $\kappa \in \mathscr{T}$, and

$$\rho\kappa = \sigma\tau\kappa$$

for $\kappa \in \mathscr{U}$. For any term r of $\lambda^{\mathscr{S} \cup \mathscr{T} \cup \mathscr{U}}$, we have $\rho r = \sigma\tau r$, so that in particular, $\rho D^i = B^i$.

For $\kappa \in \mathscr{U}$, there is $i \in I$ such that κ does not occur in the normal members of D^i, since $\boldsymbol{D}$ is useful for $\boldsymbol{C}$. For $\kappa \in \mathscr{T}$, there is $i \in I$ such that κ does not occur in the normal members of C^i, and so κ cannot occur in the normal members of D^i either.

□

The correctness of the algorithm below depends upon the following proposition, whose proof is deferred:

Proposition 10. *For every $\mathscr{S}$, every solvable interpolation system in $\lambda^{\mathscr{S}}$ is $\mathscr{S}$-self-useful.*

Algorithm. *We present an algorithm for building effective presentations and deciding solvability of interpolation systems, simultaneously verifying its correctness.*

The algorithm uses a double recursion. The outer recursion is over types σ. The inner recursion will be over the size of an interpolation system of type σ.

Suppose that for any $\mathscr{S}$ we can construct an effective presentation of $M^{\mathscr{S}}$ at proper sub-types of $\sigma = \sigma_1 \Rightarrow \cdots \Rightarrow \sigma_N \Rightarrow o$. To construct an effective presentation of $M^{\mathscr{S}}_{\sigma}$, it suffices to decide the solvability of certain interpolation systems of type σ, as in the proof of Lemma 6.

We do this by recursion over the size of an interpolation system:

$$\Phi = \left\{ [X A^i_1 \ldots A^i_N = B^i] \mid i \in I \right\}.$$

If Φ is empty, then any term of the appropriate type will suffice.

Take a set $\mathscr{T}$ of new constants with the same cardinality as Φ. Using the outer recursion, we construct an effective presentation of $M^{\mathscr{S} \cup \mathscr{T}}$ at sub-types of σ.

Now search for $s_{i,j}$ making (2) a vector useful for $\boldsymbol{B}$. Using our effective presentation of $M^{\mathscr{S} \cup \mathscr{T}}$ at sub-types of σ, we can restrict our search to a finite set of terms; this also places a bound on the n of (2) if we assume that the conjuncts are distinct.

If such $s_{i,j}$ are not found, then by Proposition 10 and 2. of Lemma 8, Φ is not solvable, and we are done. If the search is successful, then for each $\kappa \in \mathscr{T}$, let I_κ be the set of those $i \in I$ such that κ occurs in the normal form of the ith component of (2). As $|I_\kappa| < |I|$, we can use the inner recursion to decide the solvability of each $\Phi_\kappa = \left\{ [X A^i_1 \ldots A^i_N = B^i] \mid i \in I_\kappa \right\}$. We claim that Φ has a solution if and only if each Φ_κ has a solution. The 'only if' is trivial as $\Phi_\kappa \subset \Phi$. For the 'if' part, if each Φ_κ has solution $\lambda \mathrm{x}_1 \ldots \mathrm{x}_N . t_\kappa$, let t be the term

$$\lambda \mathrm{x}_1 \ldots \mathrm{x}_N . (\mathrm{x}_{i_1} s_{1,1} \ldots s_{1,\alpha_1} \wedge \cdots \wedge \mathrm{x}_{i_n} s_{n,1} \ldots s_{n,\alpha_n})[t_\kappa / \kappa]_{\kappa \in \mathscr{T}}.$$

If $i \in I_\kappa$, then $t_\kappa[A^i_1/\mathrm{x}_1 \ldots A^i_N/\mathrm{x}_N] \equiv B^i$, so that for each $i \in I$, we have that $t A^i_1 \ldots A^i_N$ is equivalent to the result of substituting B^i for each $\kappa \in \mathscr{T}$ in the ith component of (2). By 1. of Lemma 8, the result of this substitution is equivalent to B^i as required. □

The rest of this paper is concerned with the proof of Proposition 10.

4. Transferring terms

The algorithm we just gave constructs terms that are 'transferring' as defined below. We shall make use of terms in this form in the arguments that follow.

Definition 11. An η-expanded term $t = \lambda \mathrm{x}_1 \ldots \mathrm{x}_N . t_0$ is *transferring* if, for every sub-term of t in the form $\mathrm{x}_i s_1 \ldots s_\beta$, the free variables of the terms $s_1 \ldots s_\beta$ are among $\mathrm{x}_1 \ldots \mathrm{x}_n$.

Note that the condition that the term be η-expanded is vital; the term $\lambda \mathrm{x}. \mathrm{x}(\mathrm{x}(\lambda \mathrm{y}. \mathrm{y}))$ of type $((o \Rightarrow o) \Rightarrow o \Rightarrow o) \Rightarrow o \Rightarrow o$ satisfies the condition above, but its η-expansion $\lambda \mathrm{xu}. \mathrm{x}(\lambda \mathrm{v}. \mathrm{x}(\lambda \mathrm{y}. \mathrm{y})\mathrm{v})\mathrm{u}$ is not transferring as it has the sub-term $\mathrm{x}(\lambda \mathrm{y}.\mathrm{y})\mathrm{v}$, which has v free, but v is not among the outer-most bound variables, which are x and u.

Below, we give the fact that our algorithm constructs transferring terms as a lemma.

Lemma 12. *Suppose that t is a term such that every interpolation system it solves is self-useful. Then every interpolation system solved by t has a transferring solution.*

Proof. The construction used in the algorithm

$$\lambda \overline{\mathrm{x}}. r,\ \lambda \overline{\mathrm{x}}. s \ \mapsto\ \lambda \overline{\mathrm{x}}. r[s/\kappa],$$

preserves the property of being transferring. Let Φ be an interpolation system solved by t. Following the divide and conquer strategy (i.e., the inner recursion) of the algorithm, we get that the interpolation system Φ has a transferring solution. (Note that the outer recursion of the algorithm is not needed here – the sets $M_\sigma^{\mathscr{S}}$ exist and are finite, and for this proof they do not need to be given effectively.) $\square$

The following condition is used in the constructions that follow; it allows us to distinguish which sub-terms of some term are vital to the behaviour of that term.

Definition 13. Let s be a closed term of type o, and let t be a sub-term (occurrence) in s.

Form s' by replacing t with a new constant κ. The normal form of s' is a conjunction, where the conjuncts are either members of $\mathscr{S} \cup \{\top\}$ or terms with head κ.

We say that t is *active* in s if at least one of the conjuncts has head κ. (This is equivalent to s' depending on κ, when interpreted in $M^{\mathscr{S}}$.)

Lemma 14. 1. *An occurrence $t_1 t_2$ is active in s if and only if t_1 is active in s.*
2. *If t is a closed, ground, sub-term of s, then a sub-term u of t is active in s if and only if u is active in t and t is active in s.*
3. *When closed $t : o$ is active in $s[t]$, we have $s[t] \equiv t \wedge s[\top]$.*
4. *Let $s : o$ be a closed term, and form s' by replacing some ground sub-term in s by s. Then $s \equiv s'$.*

Proof. 1. and 2. are obvious. For 3., let $\kappa : o$ be a new constant; it suffices to show that $s[\kappa] \equiv \kappa \wedge s[\top]$. The normal form of $s[\kappa]$ is in the form $a_1 \wedge \cdots \wedge a_n$. As κ is active in $s[\kappa]$, κ is among the a_i, and the result follows.

We may slightly reformulate 4. as $s[t] \equiv s[\lambda \overline{\mathtt{y}}. s[t]]$, where t and $s[t]$ are closed. Let $\mathtt{X}$ be a new variable of the same type as t, and let $a_1 \wedge \cdots \wedge a_n$ be the normal form of $s[\mathtt{X}]$. If $\mathtt{X}$ does not occur in any of the a_i, then

$$s[t] \equiv a_1 \wedge \cdots \wedge a_n \equiv s[\lambda \overline{\mathtt{y}}. s[t]].$$

If $\mathtt{X}$ does occur in some of the a_i, then it occurs as the head-variable. The result now follows from the idempotency of $\wedge$. $\square$

5. Solvability implies usefulness

The following lemma is the technical heart of our proof of proposition 10.

Lemma 15. *Suppose that*

$$A_0\ (\lambda \overline{\mathtt{y}}_1 . t_1[A_1 \ldots A_N]) \ \ldots\ (\lambda \overline{\mathtt{y}}_\alpha . t_\alpha[A_1 \ldots A_N]) \ \equiv^{\mathscr{S}} \ \boldsymbol{B} \tag{3}$$

and each term $\lambda \mathtt{x}_1 \ldots \mathtt{x}_N \overline{\mathtt{y}_i}. t_i[\mathtt{x}_1 \ldots \mathtt{x}_N]$ is transferring. Then the system $[X\, A_0 \ldots A_N = \boldsymbol{B}]$ is $\mathscr{S}$-self-useful.

Proof. We show by induction on natural numbers Γ, that, for any $\mathscr{S}$, any transferring $t_1 \ldots t_\alpha$ and any $\boldsymbol{B} \in (\lambda_o^{\mathscr{S}})^I$, *if* (3) holds, and there are at most Γ occurrences of $\mathtt{x}_1 \ldots \mathtt{x}_N$ in the terms $t_i[\mathtt{x}_1 \ldots \mathtt{x}_N]$, *then* $[X\, A_0 \ldots A_N = \boldsymbol{B}]$ is $\mathscr{S}$-self-useful.

If there are no $\overline{\mathtt{x}}$ occurrences, then the LHS of (3) is in the form (2), and $\boldsymbol{B}$ is useful for $\boldsymbol{B}$, so the result holds.

Otherwise, the $t_i[\mathtt{x}_1 \ldots \mathtt{x}_N]$ can be written in the form

$$t_i'[\mathtt{x}_j\, (\lambda \overline{\mathtt{z}_1}. s_1) \ldots (\lambda \overline{\mathtt{z}_\beta}. s_\beta), \mathtt{x}_1 \ldots \mathtt{x}_N],$$

where there is exactly one $\mathtt{v}$ occurrence in $t_l'[\mathtt{v}, \overline{\mathtt{x}}]$, for some l, and no $\mathtt{v}$ occurrences in $t_i'[\mathtt{v}, \overline{\mathtt{x}}]$ for $i \neq l$. As the $t_i[\overline{\mathtt{x}}]$ are transferring, no variables other than the $\overline{\mathtt{x}}$, $\overline{\mathtt{y}_l}$ and $\overline{\mathtt{z}_p}$ occur freely in the s_p. By considering the right-most $\overline{\mathtt{x}}$ occurrence in $t_l[\overline{\mathtt{x}}]$, we can have that in fact only $\overline{\mathtt{y}_l}$ and $\overline{\mathtt{z}_p}$ occur freely in the s_p.

Let $t_i''[\overline{\mathtt{x}}] = t_i'[\kappa, \overline{\mathtt{x}}]$ for a new constant $\kappa : o$. If κ is not active in some component of

$$A_0\ (\lambda \overline{\mathtt{y}_1} . t_1''[A_1 \ldots A_N]) \ \ldots\ (\lambda \overline{\mathtt{y}_\alpha} . t_\alpha''[A_1 \ldots A_N]), \tag{4}$$

then, using 4. of Lemma 14, this vector is $\mathscr{S}$-useful for $\boldsymbol{B}$. Letting $\boldsymbol{B}''$ be (4), we may apply the induction hypothesis to show that

$$[X\, A_0 \ldots A_N = \boldsymbol{B}'']$$

is $(\mathscr{S}\cup\{\kappa\})$-self-useful. Applying the transitivity of usefulness, we get that $[X\,A_0\ldots A_N = \boldsymbol{B}]$ is $\mathscr{S}$-self-useful.

Alternatively, κ is active in each component of (4). We handle this case of the induction step in three stages; stage 1 shows how some cases of the induction statement reduce to the induction hypothesis. Stage 2 shows how more cases of the induction statement reduce to an instance of stage 1 (with Γ unchanged), and stage 3 shows that the general case of the induction statement reduces to an instance of stage 2 (with Γ unchanged).

Stage 1. Suppose that there are no $\overline{\mathrm{y}_l}$ occurrences in the s_p. Then each component of $A_j\,(\lambda\overline{\mathrm{z}_1}.s_1)\ldots(\lambda\overline{\mathrm{z}_\beta}.s_\beta)$ is a closed term, and active in the corresponding component of the lhs of (3). Thus, using 3. of Lemma 14, the lhs of (3) is equivalent to the conjunction of

$$A_j\,(\lambda\overline{\mathrm{z}_1}.s_1)\ldots(\lambda\overline{\mathrm{z}_\beta}.s_\beta),$$

and

$$\boldsymbol{B}^\bullet = A_0\ \left(\lambda\overline{\mathrm{y}_1}.t_1^\bullet[A_1\ldots A_N]\right)\ \ldots\ \left(\lambda\overline{\mathrm{y}_\alpha}.t_\alpha^\bullet[A_1\ldots A_N]\right),$$

where $t_i^\bullet[\overline{\mathrm{x}}] = t_i'[\top,\overline{\mathrm{x}}]$. The $t_i^\bullet[\overline{\mathrm{x}}]$ have (in total) strictly fewer $\overline{\mathrm{x}}$ occurrences that the $t_i[\overline{\mathrm{x}}]$, so we may apply the induction hypothesis to the second conjunct above to show that

$$[X\,A_0\ldots A_N = \boldsymbol{B}^\bullet]$$

is $\mathscr{S}$-self-useful. Applying 3. of Lemma 8, we obtain the result.

Stage 2. Now suppose that there are some $\overline{\mathrm{y}_l}$ occurrences in the s_p. Let r_κ ($\kappa\in\mathscr{T}$, with $\mathscr{T}$ a set of new constants) enumerate the type o sub-terms of the s_p which have one of the $\overline{\mathrm{y}_l}$ as a head variable. Then we may write each s_p as $s_p^*[r_\kappa/\kappa]_{\kappa\in\mathscr{T}}$, with s_p^* containing only the $\overline{\mathrm{z}_p}$ free. As $t_l[\overline{\mathrm{x}}]$ is transferring, the r_κ have free variables among the $\overline{\mathrm{y}_l}$.

Consider the normal form $\boldsymbol{E}$ of

$$A_j\,(\lambda\overline{\mathrm{z}_1}.s_1^*)\ \ldots\ (\lambda\overline{\mathrm{z}_\beta}.s_\beta^*).$$

If some of the $\kappa\in\mathscr{T}$ occur in every component of $\boldsymbol{E}$, then we will use stage 3; for now, we suppose that none of the $\kappa\in\mathscr{T}$ occur in every component of $\boldsymbol{E}$. Let

$$t_i^*[\mathrm{x}_1\ldots\mathrm{x}_N] = t_i'[\mathrm{x}_j\,(\lambda\overline{\mathrm{z}_1}.s_1^*)\ldots(\lambda\overline{\mathrm{z}_\beta}.s_\beta^*),\mathrm{x}_1\ldots\mathrm{x}_N],$$

so that

$$\boldsymbol{B}^* = A_0\ \left(\lambda\overline{\mathrm{y}_1}.t_1^*[A_1\ldots A_N]\right)\ \ldots\ \left(\lambda\overline{\mathrm{y}_\alpha}.t_\alpha^*[A_1\ldots A_N]\right)$$

is useful for $\boldsymbol{B}$ by 4. of Lemma 14, and the s_p^* do not contain the $\overline{\mathrm{y}}_p$. We can now use stage 1, to see that

$$[X\,A_0\ldots A_N = \boldsymbol{B}^*]$$

is $(\mathscr{S}\cup\mathscr{T})$-self-useful, so by the transitivity of usefulness, we derive the induction statement.

Stage 3. For the general case, suppose that $\kappa_1 \ldots \kappa_m$ occur in every component of $\boldsymbol{E}$. Let $r'_{\kappa_1} = \cdots = r'_{\kappa_m} = \top$ and $r'_\kappa = r_\kappa$ for $\kappa \notin \{\kappa_1 \ldots \kappa_m\}$. By m applications of 3. of Lemma 14, we have that

$$\lambda\overline{\mathrm{y}_l}\big(\kappa_1 \wedge \cdots \wedge \kappa_m \wedge \, \big(\boldsymbol{A}_j\,(\lambda\overline{\mathrm{z}_1}.s_1^*)\,\ldots\,(\lambda\overline{\mathrm{z}_\beta}.s_\beta^*)\big)\big)[\top/\kappa_1 \ldots \top/\kappa_m]$$
$$\equiv^{\mathscr{S}}\ \lambda\overline{\mathrm{y}_l}.\,\big(\boldsymbol{A}_j\,(\lambda\overline{\mathrm{z}_1}.s_1^*)\,\ldots\,(\lambda\overline{\mathrm{z}_\beta}.s_\beta^*)\big).$$

This implies that

$$\lambda\overline{\mathrm{y}_l}.\,t_l^{**}[\boldsymbol{A}_1 \ldots \boldsymbol{A}_N] \equiv^{\mathscr{S}} \lambda\overline{\mathrm{y}_l}.\,t'_l\big[\mathrm{x}_j\,(\lambda\overline{\mathrm{z}_1}.s_1)\,\ldots\,(\lambda\overline{\mathrm{z}_\beta}.s_\beta), \boldsymbol{A}_1 \ldots \boldsymbol{A}_N\big]$$
$$= \lambda\overline{\mathrm{y}_l}.\,t_l[\boldsymbol{A}_1 \ldots \boldsymbol{A}_N],$$

where $t_l^{**}[\mathrm{x}_1 \ldots \mathrm{x}_N]$ is

$$t'_l\big[r_{\kappa_1} \wedge \cdots \wedge r_{\kappa_m} \wedge \big(\mathrm{x}_j\,(\lambda\overline{\mathrm{z}_1}.s_1^*)\,\ldots\,(\lambda\overline{\mathrm{z}_\beta}.s_\beta^*)\big)[r'_\kappa/\kappa]_{\kappa\in\mathscr{T}}, \mathrm{x}_1 \ldots \mathrm{x}_N\big]$$

which enables us to reduce to stage 2. This completes the induction step. □

Corollary 16. *Suppose an interpolation system* $[X\,\boldsymbol{A}_1 \ldots \boldsymbol{A}_N = \boldsymbol{B}]$ *has a solution* t *in the form*

$$\lambda\mathrm{x}_1 \ldots \mathrm{x}_N.\,\mathrm{x}_i\,(\lambda\overline{\mathrm{y}_1}.\,t_1)\,\ldots\,(\lambda\overline{\mathrm{y}_\alpha}.\,t_\alpha),$$

with each term $\lambda\mathrm{x}_1 \ldots \mathrm{x}_N\,\overline{\mathrm{y}_l}.\,t_l$ *transferring. Then the interpolation system has a transferring solution.*

Proof. By the lemma above, any interpolation system solved by t is self-useful, so by Lemma 12, there is a transferring solution to any interpolation system solved by t. □

Lemma 17. *For every finite* $\mathscr{S}$, *every equivalence class of* $\equiv^{\mathscr{S}}$ *contains a transferring term.*

Proof. We show by induction on the size of a β-normal η-long term s, that s is equivalent to a transferring term.

If s is a closure of a conjunction, we apply the induction hypothesis to the closures of the conjuncts.

Suppose that s is $\lambda\mathrm{x}_1 \ldots \mathrm{x}_n.\,\mathrm{x}_i\,(\lambda\overline{\mathrm{y}_1}.s_1)\,\ldots\,(\lambda\overline{\mathrm{y}_\alpha}.s_\alpha)$. By the induction hypothesis, there are transferring terms $\lambda\overline{\mathrm{x}}\,\overline{\mathrm{y}_j}.\,t_j$ equivalent to $\lambda\overline{\mathrm{x}}\,\overline{\mathrm{y}_j}.\,s_j$, so that s is equivalent to

$$\lambda\mathrm{x}_1 \ldots \mathrm{x}_n.\,\mathrm{x}_i\,(\lambda\overline{\mathrm{y}_1}.\,t_1)\,\ldots\,(\lambda\overline{\mathrm{y}_\alpha}.\,t_\alpha).$$

There is an interpolation system Φ whose solution-set is the equivalence class of s. As the term above solves Φ, we can apply Corollary 16 to show that Φ has a transferring solution. □

Theorem 18. *The solvability of interpolation systems is decidable, so there is an effective presentation of the fully abstract model of* PCF$_1$ $= \lambda^{\{\perp\}}$, *and the observational pre-order of* PCF$_1$ *is decidable.*

Proof. We give a proof of Proposition 10, as this suffices to show that the algorithm of Section 3 does in fact work. Let $\Phi = [X\, A_1 \ldots A_N = \boldsymbol{B}]$ be an interpolation system. We wish to show that if Φ has a solution, then Φ is self-useful.

Suppose that Φ has a solution in the form $\lambda\overline{\mathrm{x}}.\,\mathrm{x}_i\,(\lambda\overline{\mathrm{y}_1}.\,t_1)\,\ldots\,(\lambda\overline{\mathrm{y}_\alpha}.\,t_\alpha)$. By Lemma 17, we can assume that the terms $\lambda\overline{\mathrm{x}}\,\overline{\mathrm{y}_j}.\,t_j$ are transferring, and thus apply Lemma 15, so that Φ is self-useful.

In general, Φ may have a solution in the form $\lambda\mathrm{x}_1 \ldots \mathrm{x}_n.(s_1 \wedge \cdots \wedge s_\alpha)$. Let $\boldsymbol{B}_i$ be $(\lambda\mathrm{x}_1 \ldots \mathrm{x}_N.\,s_i)\,A_1 \,\ldots\, A_N$. Applying the paragraph above to each of the systems $[X\, A_1 \ldots A_N = \boldsymbol{B}_i]$, we can find $\mathscr{T}_i$ and $\boldsymbol{E}_i \in (M_o^{\mathscr{S}\cup\mathscr{T}_i})^I$ useful for $\boldsymbol{B}_i$ that can be expressed as a conjunction of the form (2). WLOG, we can take the $\mathscr{T}_i$ to be disjoint, so that $\boldsymbol{E}_1 \wedge \cdots \wedge \boldsymbol{E}_\alpha$ is useful for $\boldsymbol{B}_1 \wedge \cdots \wedge \boldsymbol{B}_\alpha \equiv \boldsymbol{B}$, and Φ is self-useful. □

6. Conclusion

We have shown that the observational order, of a very restricted fragment of PCF, is decidable. Such issues were raised by Jung and Stoughton [2], as a test for semantical techniques used in relation to the full abstraction problem for PCF. They originally asked the question for finitary PCF; however, that fragment turns out to have an undecidable observational pre-order [4].

Currently known constructions of the fully abstract model of PCF do not appear to lead to a proof of the decidability result here. One should not be too worried about this – the connection between the fully abstract models of finitary PCF and PCF is much tighter than that between the models of unary PCF and PCF. The proof of decidability given here works because unary PCF is a special case of the typed lambda calculus over certain algebras, not because it is a fragment of PCF.

Having said that, a semantical proof of the result here, avoiding the rather horrible syntactical manipulations, would be interesting, and perhaps would shed more light on the result. Additionally, the techniques used in this paper originated with Padovani's work on the still open problem of the decidability of higher order matching. Maybe new proofs of the known decidability results would enable us to make progress on higher order matching.

Acknowledgements

Thanks are due to an anonymous referee, who provided detailed criticisms of an earlier version of the paper.

Postscript

As I was making the final corrections to this paper, I received a manuscript from Schmidt-Schauss [8] giving a much simpler proof of the result here. It proceeds by using a strictness analysis of terms to construct sets of combinators that can be used to build up the required presentation of the model.

References

[1] V. Breazu-Tannen, J. Gallier. Polymorphic rewriting conserves algebraic confluence, Inform. and Comput. 114 (1994) 1–29.
[2] A. Jung, A. Stoughton, Studying the fully abstract model of PCF within its continuous function model, in: M. Bezem, J.F. Groote, (Eds.), Typed Lambda Calculi and Applications, Lecture Notes in Computer Science, vol. 664 Springer, Berlin, 1993, pp. 232–257.
[3] S. Kahrs, The variable containment problem, in: Higher-Order Algebra, Logic and Term Rewriting, Lecture Notes in Computer Science, vol. 1074, Springer, Berlin, 1995, pp. 109–123.
[4] R. Loader, Finitary PCF is undecidable, Theoret. Comput. Sci., submitted.
[5] V. Padovani, Decidability of all minimal models, in: M. Coppo et al., (Eds.), Proc. BRA Types Workshop, Torino, 1995, Lecture Notes in Computer Science, vol. 1158, Springer, Berlin, to appear.
[6] G. Plotkin, λ-definability and logical relations, Memorandum SAI – RM – 4, School of Artificial Intelligence, University of Edinburgh, 1973.
[7] G. Plotkin, LCF considered as a programming language, Theoret. Comput. Sci. 5 (1977) 223–255.
[8] M. Schmidt-Schauss, Decidability of behavioral equivalence in unary PCF, Theoret. Comput. Sci., to appear.
[9] R. Statman, Logical relations and the typed λ-calculus, Inform. and Control 65 (1985) 85–97.