Logical Full Abstraction and PCF

John Longley Gordon Plotkin

November 27, 2000

Abstract

We introduce the concept of *logical full abstraction*, generalising the usual equational notion. We consider the language PCF and two extensions with "parallel" operations. The main result is that, for standard interpretations, logical full abstraction is equivalent to equational full abstraction together with universality; the proof involves constructing enumeration operators. We also consider restrictions on logical complexity and on the level of types.

1 Introduction

The study of denotational semantics seeks to provide mathematical descriptions of programming languages by giving denotations of programs in terms of previously understood mathematical structures. For example, if P is a program that takes an input and produces an output, we might take its denotation to be a function from a set of input-values to a set of output-values. The most widely-known approach to denotational semantics is that of traditional domain theory (see e.g., [14]), where the mathematical structures involved are certain kinds of complete partial order (cpo). Other kinds of mathematical structure have also been used successfully—for a selection of different approaches see [1, 11, 13].

One of the principal aims of denotational semantics is to deepen our understanding of the logic of programming languages, and to provide conceptual and mathematical tools for reasoning about programs. A more specific goal is to provide mathematical foundations for "program logics" of a kind that could be used by ordinary programmers.

Denotational semantics can be used to establish relationships between a programming language **L** and a logic **J**. By giving interpretations of both **L** and **J** in some common mathematical structure $\mathcal{M}$, we may be able to show that if certain theorems are provable in the logic then certain properties of programs hold—for example, that if the sentence $P(3) = 5$ is provable in **J** then the program $P(3)$ returns the answer 5. Such a result would show that the logic **J** was indeed useful for proving certain facts about programs in **L**.

In this kind of situation we have a way of directly understanding the meaning of certain simple sentences of **J** (e.g., $P(3) = 5$) as statements about computations in **L**. One is then prompted to ask whether one could extend this to *all* sentences of the logic, and give an interpretation of **J** purely in terms of the language **L** and its evaluation rules, without reference to the structure $\mathcal{M}$. For example, one might interpret quantifiers as ranging over closed programs or terms of appropriate type. We might call this an *operational* interpretation of **J**, in contrast to its *denotational* interpretation in $\mathcal{M}$. Besides the intrinsic interest of such an interpretation, it seems likely that an operational interpretation would be more easily grasped by a non-specialist than a denotational one.

Now, given a logic **J** with both an operational interpretation in terms of **L** and a denotational interpretation in $\mathcal{M}$, it is natural to ask whether these "agree" in the sense that a sentence is true under one interpretation if and only if it is true under the other. In this case, we will say that the interpretation of **L** in $\mathcal{M}$ is *logically fully abstract* (or LFA) for **J**. A logically fully abstract interpretation can be used to show that all sentences provable in **J** express true facts about **L** under the operational interpretation. Note that the familiar notion of (equational) full abstraction can be seen as a special case of logical full abstraction: consider a logic **J** whose only assertions are equations between terms of **L**, and whose operational interpretation is "observational equivalence".

Both the general concept and the name "logical full abstraction" are due to the second author, though the idea was first worked out in the first author's Ph.D. thesis [11]. The idea as we have outlined it above is of course extremely general, as it depends not only on **L** and **J** but also on the kinds of operational and denotational interpretation we have in mind. The aim of the present paper is to illustrate the basic idea by discussing one particular kind of logical full abstraction, in the context of a simple logic for the prototypical functional language PCF (see [14, 3]). We anticipate that the study of other notions of logical full abstraction (whether for PCF or other languages) will prove a very interesting area for further research.

The study of equational full abstraction commonly results in theories of *extensional* objects, often of functions and data structures; these objects have a natural mathematical structure, perhaps of order-theoretic, topological or algebraic character. The study here of logical full abstraction results rather in *intensional* concerns, such as the study of definability and so of computability. These distinctions harken back to Scott's original explicit choice [16, 17] to investigate first extensional theories and only then to consider questions of computability and of the relation with symbolic computation. They also bring to mind the much more recent programme of synthetic domain theory, where one tries to integrate the different approaches by working in, for example, the effective topos [7]. One should also remark that intensional aspects may nonetheless play a role in the study of equational full abstraction—see the study of games in [1, 8].

The rest of the paper is structured as follows. In Section 2 we review the definitions of the three versions of PCF that we will consider. We also define the syntax of a program logic for these languages, and propose a simple operational interpretation of this logic. In Section 3 we introduce a very general notion of denotational interpretation for our languages, and show how such an interpretation gives rise to a denotational interpretation of the logic. We thus obtain a notion of logical full abstraction. In Section 4 we prove the main result of the paper: a standard interpretation is LFA if and only if it is both *equationally fully abstract (EFA)* and *universal* (meaning, roughly, that every element of the model is definable). We end in Section 5 with a few further observations, and mention some open questions and some avenues for future investigation. In particular, we consider restrictions on logical complexity and on the level of types. For example, it follows from our main theorem that a standard interpretation is LFA iff it is Π_2-LFA (i.e., LFA for Π_2-sentences). There are standard interpretations which are EFA, but not Π_1-LFA; it is an open question whether there are any interpretations which are Π_1-LFA but not Π_2-LFA.

2 PCF and its Logic

PCF is an extension of the simply-typed λ-calculus with arithmetic operators and general recursion. It can be regarded as a prototypical "sequential" functional language; an understanding of PCF is thus an important step towards an understanding of modern functional languages such as Haskell, Miranda and ML. We begin by reviewing the syntax and evaluation rules for PCF, and for two extensions, PCF^{+} and PCF^{++}, obtained by adding "parallel" operations. All three of these languages appear essentially in [14]; the formulations here differ in two inessential respects: one is the absence of a Boolean type; the other is the use of a "parallel-or" constant rather than a parallel conditional (for which see [18]).

The *types* of PCF are built up from a single ground type ι (the natural numbers) using the right-associative binary type constructor $\to$; we write $M : \sigma$ to mean "M is a term of type σ". For each type σ we have a countably infinite set of *variables* of type σ, ranged over by $x^\sigma, y^\sigma, z^\sigma, \ldots$; we also have the following collection of *constants*:

$$\begin{array}{rlrl} 0, 1, 2, \ldots & : \iota, & \mathsf{cond} & : \iota \to \iota \to \iota \to \iota, \\ \mathsf{succ}\,,\ \mathsf{pred} & : \iota \to \iota, & \mathsf{Y}_\sigma & : (\sigma \to \sigma) \to \sigma. \end{array}$$

The *terms* of PCF are built up from the variables and constants as usual in the simply-typed λ-calculus:

- if $M : \tau$, then $(\lambda x^\sigma . M) : \sigma \to \tau$;
- if $M : \sigma \to \tau$ and $N : \sigma$, then $(MN) : \tau$.

We frequently omit unnecessary parentheses, taking juxtaposition to be left-associative; we also omit type superscripts on variables, when this causes no ambiguity. We identify terms up to change of bound variables (α-conversion); we write $M[N_1/x_1^{\sigma_1}, \ldots, N_n/x_n^{\sigma_n}]$ for capture-avoiding simultaneous substitution (where $N_1 : \sigma_1, \ldots, N_n : \sigma_n$).

An *environment* is a finite non-repetitive list $x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$ of variables (where $n \geq 0$); the empty environment is written $\langle\rangle$. We say that M is a *term of type σ in environment* Γ (and write $\Gamma \vdash M : \sigma$) if $M : \sigma$ and all the free variables of M occur in Γ. If x^σ is a variable not in Γ, we write Γ, x^σ for the environment obtained by appending x^σ to Γ.

The *evaluation rules* for PCF are given by defining a notion of *reduction* (or rewriting) on closed terms. Specifically, we inductively define a binary relation $M \to N$ on closed terms of the same type as follows (here n ranges over the *numerals* $0, 1, 2, \ldots$):

- $(\lambda x^\sigma . M)N \to M[N/x^\sigma]$;
- $\mathsf{succ}\ n \to (n+1)$, $\mathsf{pred}\ (n+1) \to n$, $\mathsf{pred}\ 0 \to 0$, $\mathsf{cond}\ 0NP \to N$, $\mathsf{cond}\ (n+1)NP \to P$, $\mathsf{Y}_\sigma M \to M(\mathsf{Y}_\sigma M)$;
- if $M \to M'$ then $MN \to M'N$;
- if $M \to M' : \iota$ then $\mathsf{succ}\ M \to \mathsf{succ}\ M'$, $\mathsf{pred}\ M \to \mathsf{pred}\ M'$, $\mathsf{cond}\ MNP \to \mathsf{cond}\ M'NP$.

We think of $\to$ as a "one-step reduction relation"; we write $\to^+$ for its transitive closure, and $\to^*$ for its transitive reflexive closure. We say that a term $M : \iota$ *terminates* if $M \to^* n$ for some (necessarily unique) numeral n.

The language defined above is intuitively "sequential"—no two subterms of a term are ever evaluated "in parallel". We now introduce two extensions of PCF including parallel operators. The language PCF$^+$ is defined in the same way as PCF except that we add an extra constant $\mathsf{por} : \iota \to \iota \to \iota$ ("parallel-or"), together with the reduction rules:

- $\mathsf{por}\ 0M \to 0$, $\mathsf{por}\ M0 \to 0$, $\mathsf{por}\ (m+1)(n+1) \to 1$;
- if $M \to M' : \iota$ then $\mathsf{por}\ MN \to \mathsf{por}\ M'N$, $\mathsf{por}\ NM \to \mathsf{por}\ NM'$.

The syntax of PCF^{++} is defined in the same way as PCF$^+$, except that we add a further constant $\mathsf{exists} : (\iota \to \iota) \to \iota$ ("existential quantification"). Its reduction rules are those for PCF$^+$ together with the following, writing Ω_σ for $\mathsf{Y}_\sigma(\lambda x^\sigma . x)$:

- if $Mn \to^+ 0$ for some n, then $\mathsf{exists}\ M \to 0$;

- if $M\Omega_\iota \to^+ m+1$, then exists $M \to 1$.

We say that a one-step reduction $M \to M'$ is *deterministic* if whenever $M \to M''$ then $M' = M''$, and write $M \to_d M'$ for this relationship. Note that whereas for PCF every one-step reduction is deterministic, this is not so for PCF$^+$ and PCF^{++}. Nevertheless, in all these languages *evaluation* is deterministic: if $M \to^* n$ and $M \to^* n'$ then $n = n'$. (In fact the more general *Church-Rosser Property* holds, that if $M \to^* N_i$, for $i = 1, 2$, then for some P, $N_i \to^* P$, for $i = 1, 2$).

We need some standard notions. Suppose $\mathbf{L}$ is one of the three languages PCF, PCF$^+$ or PCF^{++}. A (one-place) *term context* $C[\ \ ]$ of $\mathbf{L}$ is a term of $\mathbf{L}$ with zero or more holes, to be filled by a term of appropriate type. Two terms $M, M' : \sigma$ are *observationally equivalent* (and we write $M \approx M'$) if for all term contexts $C[\ \ ]$ such that $C[M], C[M']$ are closed terms of type ι we have $C[M] \to^* n$ iff $C[M'] \to^* n$. The Context Lemma characterises this equivalence. When M and M' are both closed, the lemma asserts that for $\sigma = \sigma_1 \to \cdots \to \sigma_h \to \iota$, $M \approx M'$ iff for all closed terms $N_1 : \sigma_1, \ldots, N_h : \sigma_h$ and numerals n, $MN_1 \ldots N_h \to^* n$ iff $M'N_1 \ldots N_h \to^* n$ (a more general version for open terms is easily derived). An operational proof of the Context Lemma for PCF can be found in [12, 3] and similar proofs can be obtained for PCF$^+$ and PCF^{++}; for these latter two languages it is also a consequence of the facts that the usual cpo model is adequate and that all finite elements are definable (see Section 3 below for a definition of adequacy). *Operational soundness* (that is, if $M \to^* N$ then $M \approx N$) is a consequence of the Context Lemma, together with the Church-Rosser Property.

Now that we have defined the languages of interest, we introduce the syntax of a simple many-sorted program logic $\mathbf{J_L}$, much in the spirit of LCF [5]. We believe that this is the kind of logic that would in principle be useful for specifying and proving properties of programs. The *sorts* of $\mathbf{J_L}$ are the types of $\mathbf{L}$; the expressions of sort σ in $\mathbf{J_L}$ are precisely the terms of type σ in $\mathbf{L}$; and the logical variables of sort σ are just the term variables of type σ. The syntax of the *formulae* of $\mathbf{J_L}$ is as follows (here M, N are expressions of the same type and $P : \iota$):

$$\phi \ ::= \ \ \bot \mid M = N \mid P \Downarrow \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \phi_1 \supset \phi_2 \mid \forall x^\sigma.\phi_1 \mid \exists x^\sigma.\phi_1.$$

Note that $\mathbf{J_L}$ is really a many-sorted *first-order* logic—we have a separate ground sort for each type σ. We identify formulae up to change of bound variables (α-conversion); we write $\phi[N_1/x_1^{\sigma_1}, \ldots, N_n/x_n^{\sigma_n}]$ for capture-avoiding simultaneous substitution into formulae (where $N_1 : \sigma_1, \ldots, N_n : \sigma_n$). We will not be specific here about the axioms and inference rules of $\mathbf{J_L}$, however we essentially have in mind those of classical first-order logic.

Next we give a simple *operational interpretation* of $\mathbf{J_L}$; this gives us a way of translating formulae into statements about computations in $\mathbf{L}$. We define a relation $\models_{op} \phi$ (read "ϕ is operationally true") on *sentences* (i.e., closed formulae) of $\mathbf{J_L}$ as follows:

- $\models_{op} \bot$ doesn't hold;
- $\models_{op} (M = N)$ iff $M \approx N$;
- $\models_{op} (P \Downarrow)$ iff P terminates;
- $\models_{op} \varphi \wedge \psi$ iff $\models_{op} \varphi$ and $\models_{op} \psi$;
- $\models_{op} \varphi \vee \psi$ iff $\models_{op} \varphi$ or $\models_{op} \psi$;
- $\models_{op} \varphi \supset \psi$ iff either $\not\models_{op} \varphi$ or $\models_{op} \psi$;
- $\models_{op} \forall x^{\sigma}.\varphi$ iff $\models_{op} \varphi[M/x]$ for all closed $M : \sigma$;
- $\models_{op} \exists x^{\sigma}.\varphi$ iff $\models_{op} \varphi[M/x]$ for some closed $M : \sigma$.

We extend the relation $\models_{op}$ to all formulae as follows: if ϕ has free variables among $x_1, \ldots, x_n$, then $\models_{op} \phi$ iff $\models_{op} \phi[M_1/x_1, \ldots, M_n/x_n]$ for all closed expressions $M_1, \ldots, M_n$ of appropriate types.

Notice that the notion of operational truth only requires concepts relating to $\mathbf{L}$ itself—we are thus hopeful that this interpretation of the logic would be readily understood by a non-specialist. However, we should point out that operational interpretations of the formulae of $\mathbf{J_L}$ other than the "classical" one we have given are possible—for an alternative (arguably more "computational") interpretation see [11, Section 8.2].

3 Denotational Interpretations of PCF

We now introduce a very general notion of denotational interpretation for our languages; it is convenient to use the language of category theory. Given a category $\mathcal{C}$ with finite products, we interpret types of $\mathbf{L}$ by objects of $\mathcal{C}$, and terms of $\mathbf{L}$ by morphisms of $\mathcal{C}$; we also need a semantic correlate of termination. We therefore say that an *interpretation* $\mathcal{I}$ of $\mathbf{L}$ in $\mathcal{C}$ is given by the following data:

- for each type σ an object $\mathcal{I}[\![\,\sigma\,]\!]$ of $\mathcal{C}$ (and for each environment $\Gamma = x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$ we write $\mathcal{I}[\![\,\Gamma\,]\!]$ for $\mathcal{I}[\![\,\sigma_1\,]\!] \times \cdots \times \mathcal{I}[\![\,\sigma_n\,]\!]$);
- for each $\mathbf{L}$-term $M : \sigma$ in environment Γ a morphism $\mathcal{I}[\![\,\Gamma \vdash M\,]\!]$ from $\mathcal{I}[\![\,\Gamma\,]\!]$ to $\mathcal{I}[\![\,\sigma\,]\!]$ (and if M is closed we write $\mathcal{I}[\![\,M\,]\!]$ for $\mathcal{I}[\![\,\langle\rangle \vdash M\,]\!]$);
- a set $T \subset \mathrm{Hom}(1, \mathcal{I}[\![\,\iota\,]\!])$ (to be thought of as the set of "fully defined" or "terminating" elements of $\mathcal{I}[\![\,\iota\,]\!]$).

We impose two requirements. First, for any environment $\Gamma = x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$ we require that $\mathcal{I}[\![\,\Gamma \vdash x_i^{\sigma_i}\,]\!] : \mathcal{I}[\![\,\Gamma\,]\!] \to \mathcal{I}[\![\,\sigma_i\,]\!]$ be the evident projection, for

$1 \leq i \leq n$. Second, we require that $\mathcal{I}$ be *compositional* in the following sense: if $\Gamma = x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}$, $\Gamma \vdash M:\tau$ and $\Delta \vdash N_i:\sigma_i$, for $1 \leq i \leq n$, then

$$\mathcal{I}[\![\, \Delta \vdash M[N_1/x_1, \ldots, N_n/x_n] \,]\!] \;=\; \mathcal{I}[\![\, \Gamma \vdash M \,]\!] \circ \langle \mathcal{I}[\![\, \Delta \vdash N_1 \,]\!], \ldots, \mathcal{I}[\![\, \Delta \vdash N_n \,]\!] \rangle.$$

That is, tupling and composition in $\mathcal{C}$ corresponds to substitution in $\mathbf{L}$.

This definition of interpretation for $\mathbf{L}$ is extremely weak (we do not require $\mathcal{C}$ to be cartesian-closed, for instance), but it suffices for our purposes. The most familiar concrete example is given by the category of cpos: all three of our languages have a canonical interpretation in this category (see [14]).

The following concepts will play a significant role in the rest of the paper:

- $\mathcal{I}$ is *sound* if $M \to M'$ implies $\mathcal{I}[\![\, M \,]\!] = \mathcal{I}[\![\, M' \,]\!]$;
- $\mathcal{I}$ is *adequate* if for all closed $M:\iota$ we have $\mathcal{I}[\![\, M \,]\!] = \mathcal{I}[\![\, n \,]\!]$ iff $M \to^* n$;
- $\mathcal{I}$ is *equationally fully abstract (EFA)* if for all closed $M, M':\sigma$ we have $\mathcal{I}[\![\, M \,]\!] = \mathcal{I}[\![\, M' \,]\!]$ iff $M \approx M'$;
- $\mathcal{I}$ is *atomically fully abstract (AFA)* if it is EFA and for all closed $M:\iota$ we have $\mathcal{I}[\![\, M \,]\!] \in T$ iff M terminates;
- $\mathcal{I}$ is *universal* if every morphism $f : 1 \to \mathcal{I}[\![\, \sigma \,]\!]$ in $\mathcal{C}$ is *definable* (meaning that there is a closed term $M:\sigma$ such that $\mathcal{I}[\![\, M \,]\!] = f$);
- $\mathcal{I}$ is *standard* if every morphism $1 \to \mathcal{I}[\![\, \iota \,]\!]$ is definable and the set T of fully defined elements is $\{\mathcal{I}[\![\, n \,]\!] \mid n \text{ is a numeral}\}$.

Our definitions of equational full abstraction and universality are weak in that they involve only *closed* terms—in our general setting it is not possible to deduce the corresponding stronger facts for open terms (and arbitrary morphisms). However, the two notions of full abstraction coincide if the interpretation models β-conversion. Further, the two notions of universality coincide if the interpretation is *cartesian-closed* (by which we mean that the underlying category is cartesian closed, and that the higher-order types, λ-abstraction and application are interpreted accordingly—see [10]; this property implies that the interpretation models $\beta\eta$-conversion). It follows from operational soundness that any EFA interpretation is sound. Note that the usual interpretation in cpos is sound, adequate, standard and cartesian closed; it is AFA for PCF$^+$ and PCF^{++} but not EFA for PCF (see [14]).

Next we show how any interpretation $\mathcal{I}$ of $\mathbf{L}$ gives rise to a *denotational interpretation* of $\mathbf{J_L}$. First some notation: For each type σ let S^σ be $\mathrm{Hom}(1, \mathcal{I}[\![\, \sigma \,]\!])$; we may think of S^σ informally as the set of "elements" of $\mathcal{I}[\![\, \sigma \,]\!]$. For each environment Γ we also let S^Γ be $\mathrm{Hom}(1, \mathcal{I}[\![\, \Gamma \,]\!])$. Then whenever $\Gamma \vdash M:\sigma$ we have the set-theoretic function $\mathcal{I}[\![\, \Gamma \vdash M \,]\!] \circ - : S^\Gamma \to S^\sigma$.

For any formula ϕ with $\mathrm{FV}(\phi) \subseteq \Gamma$, we can now define a subset $[\![\, \phi \,]\!]^\Gamma$ of S^Γ, corresponding intuitively to the set of tuples of elements for which the predicate ϕ holds:

- $z \in [\![\, \bot \,]\!]^{\Gamma}$ never;
- $z \in [\![\, M = N \,]\!]^{\Gamma}$ iff $\mathcal{I}[\![\, \Gamma \vdash M \,]\!] \circ z = \mathcal{I}[\![\, \Gamma \vdash N \,]\!] \circ z$;
- $z \in [\![\, P \Downarrow \,]\!]^{\Gamma}$ iff $\mathcal{I}[\![\, \Gamma \vdash P \,]\!] \circ z \in T$;
- $z \in [\![\, \phi \wedge \psi \,]\!]^{\Gamma}$ iff $z \in [\![\, \phi \,]\!]^{\Gamma}$ and $z \in [\![\, \psi \,]\!]^{\Gamma}$;
- $z \in [\![\, \phi \vee \psi \,]\!]^{\Gamma}$ iff $z \in [\![\, \phi \,]\!]^{\Gamma}$ or $z \in [\![\, \psi \,]\!]^{\Gamma}$;
- $z \in [\![\, \phi \supset \psi \,]\!]^{\Gamma}$ iff $z \notin [\![\, \phi \,]\!]^{\Gamma}$ or $z \in [\![\, \psi \,]\!]^{\Gamma}$;
- $z \in [\![\, \forall x^{\sigma}.\phi \,]\!]^{\Gamma}$ iff $\langle z, w \rangle \in [\![\, \phi \,]\!]^{\Gamma, x^{\sigma}}$ for all $w \in S^{\sigma}$;
- $z \in [\![\, \exists x^{\sigma}.\phi \,]\!]^{\Gamma}$ iff $\langle z, w \rangle \in [\![\, \phi \,]\!]^{\Gamma, x^{\sigma}}$ for some $w \in S^{\sigma}$.

(in the last two cases, we assume—without loss of generality—that x^{σ} does not occur in Γ). We say that ϕ is *denotationally true* under the interpretation $\mathcal{I}$ (and write $\models_{\mathcal{I}} \phi$) if $[\![\, \phi \,]\!]^{\Gamma}$ is the whole of S^{Γ}, where Γ contains all the free variables of ϕ. In particular, if ϕ is closed then $\models_{\mathcal{I}} \phi$ iff $* \in [\![\, \phi \,]\!]$, where $*$ is the unique element of $S^{\langle\rangle}$ and we write $[\![\, \phi \,]\!]$ for $[\![\, \phi \,]\!]^{\langle\rangle}$.

Now that we have given both operational and denotational interpretations of $\mathbf{J}_{\mathbf{L}}$, we have a natural notion of logical full abstraction:

Definition 1 *An interpretation $\mathcal{I}$ of $\mathbf{L}$ is* logically fully abstract (LFA) *if for all sentences ϕ of $\mathbf{J}_{\mathbf{L}}$ we have $\models_{op} \phi$ iff $\models_{\mathcal{I}} \phi$. More generally, if $\mathcal{F}$ is a class of sentences of $\mathbf{J}_{\mathbf{L}}$, we say $\mathcal{I}$ is* LFA for $\mathcal{F}$ *if for all $\phi \in \mathcal{F}$ we have $\models_{op} \phi$ iff $\models_{\mathcal{I}} \phi$.*

Note that if an interpretation is LFA for a class of sentences, then it is also LFA for the Boolean closure of that class. The next lemma shows that such notions as adequacy[1] and eq uational full abstraction can be recovered as special instances of logical full abstraction.

Proposition 2 *Let $\mathcal{I}$ be an interpretation of $\mathbf{L}$. Then*

(i) $\mathcal{I}$ is EFA (respectively AFA) iff it is LFA for all sentences of the form $M = N$ (respectively and $P \Downarrow$);

(ii) $\mathcal{I}$ is adequate iff it is LFA for all sentences of the form $P = n$;

(iii) Suppose that $\mathcal{I}$ is adequate. Then the definable elements of T are those of the form $\mathcal{I}[\![\, n \,]\!]$ iff $\mathcal{I}$ is LFA for all sentences of the form $P \Downarrow$.

Proof (i) is immediate. For (ii), it suffices to note that, by the Context Lemma, $P \approx n$ iff $P \to^{*} n$ for closed $P : \iota$. For (iii), note that the condition on T is equivalent to the statement that for all closed $P : \iota$, $\models_{\mathcal{I}} P \Downarrow$ iff $\models_{\mathcal{I}} P = n$ for some n and also that the statement that $\mathcal{I}$ is LFA for all sentences of the form $P \Downarrow$ holds is equivalent to the statement that for all closed $P : \iota$, $\models_{\mathcal{I}} P \Downarrow$ iff $\models_{op} P \Downarrow$. But it is an easy consequence of adequacy and the Context Lemma that for all closed $P : \iota$, $\models_{\mathcal{I}} P = n$ for some n iff $\models_{op} P \Downarrow$. $\square$

[1]This observation is due to Eugenio Moggi.

The above lemma makes it clear that any EFA interpretation is adequate. Further, such an interpretation is AFA iff the definable elements of T are those of the form $\mathcal{I}[\![\, n \,]\!]$. In particular, a standard EFA interpretation is AFA.

4 A characterization of LFA Interpretations

In this section we prove the following theorem characterizing LFA interpretations. This is the main result of the paper.

Theorem 3 *Let $\mathcal{I}$ be a standard interpretation of* $\mathbf{L}$. *Then it is LFA for* $\mathbf{J_L}$ *iff it is both EFA and universal.*

The standardness condition is a mild requirement that seems to hold in all natural interpretations. (We will see below that this condition is in fact necessary for the conclusion.) Our theorem can be used to show that LFA interpretations exist, and that particular interpretations are LFA for particular languages; it also guides us in the search for LFA interpretations.

The right-to-left implication in the theorem is fairly straightforward:

Lemma 4 *Let $M : \sigma$ be closed, $z \in S^{\Gamma}$. Then $\langle z, \mathcal{I}[\![\, M \,]\!]\rangle \in [\![\, \phi \,]\!]^{\Gamma, x^{\sigma}}$ iff $z \in [\![\, \phi[M/x] \,]\!]^{\Gamma}$.*

Proof First note that if $\Gamma, x^{\sigma} \vdash N : \tau$ then, by the requirements placed on interpretations, $\mathcal{I}[\![\, \Gamma, x^{\sigma} \vdash N \,]\!] \circ \langle z, \mathcal{I}[\![\, M \,]\!]\rangle = \mathcal{I}[\![\, \Gamma \vdash N[M/x^{\sigma}] \,]\!] \circ z$. The proof of the lemma is now a routine induction on ϕ. □

Proposition 5 *Suppose $\mathcal{I}$ is a standard, EFA and universal interpretation of* $\mathbf{L}$. *Then it is LFA.*

Proof We first show by induction that $\models_{op} \phi$ iff $\models_{\mathcal{I}} \phi$ for all sentences ϕ of $\mathbf{J_L}$. For the sentence $\perp$ this is trivial. For sentences of the form $M = N$ or $P \Downarrow$ it is given by Proposition 2(i) and the fact that a standard EFA interpretation is AFA. The cases for the connectives $\wedge, \vee, \supset$ are all trivial.

For sentences $\forall x^{\sigma}.\phi$, suppose first that $\models_{\mathcal{I}} \forall x^{\sigma}.\phi$. Then given any closed $M : \sigma$ we have $\langle *, \mathcal{I}[\![\, M \,]\!]\rangle \in [\![\, \phi \,]\!]^{x^{\sigma}}$ by definition of $[\![\, \forall x^{\sigma}.\phi \,]\!]$, hence, by Lemma 4, $* \in [\![\, \phi[M/x] \,]\!]$ and so $\models_{op} \phi[M/x]$ by the induction hypothesis. Thus $\models_{op} \forall x^{\sigma}.\phi$. Conversely, suppose $\models_{op} \forall x^{\sigma}.\phi$. For any $w \in S^{\sigma}$, by universality we have $w = \mathcal{I}[\![\, M \,]\!]$ for some closed $M : \sigma$. But we have that $\models_{op} \phi[M/x]$, and so $* \in [\![\, \phi[M/x] \,]\!]$ by the induction hypothesis. Hence by Lemma 4 $\langle *, w\rangle \in [\![\, \phi \,]\!]^{x^{\sigma}}$. Thus $[\![\, \phi \,]\!]^{x^{\sigma}} = S^{x^{\sigma}}$, so $\models_{\mathcal{I}} \forall x^{\sigma}.\phi$. The argument for $\exists$ is similar.

This completes the proof for sentences. To see that the result extends to all formulae, just observe that if $\forall \vec{x}.\phi$ is the universal closure of ϕ then $\models_{op} \phi$ iff $\models_{op} \forall \vec{x}.\phi$, and $\models_{\mathcal{I}} \phi$ iff $\models_{\mathcal{I}} \forall \vec{x}.\phi$. □

For the converse direction, we know from Proposition 2(i) that every LFA interpretation is AFA. Thus it only remains to show that every standard LFA interpretation is universal. To show this, we will for each type σ construct a closed term $E^\sigma : \iota \to \sigma$ (called an *enumerator* for type σ) such that for all closed $M : \sigma$ there exists n such that $E^\sigma n \approx M$. It is easy to see that this suffices: we have $\models_{op} \forall x^\sigma.\exists y^\iota.E^\sigma y = x$, and so if $\mathcal{I}$ is LFA then also $\models_{\mathcal{I}} \forall x^\sigma.\exists y^\iota.E^\sigma y = x$. That is, for all $x \in S^\sigma$ there exists $y \in S^\iota$ such that $\mathcal{I}[\![\, z^\iota \vdash E^\sigma z^\iota \,]\!] \circ y = x$. But $\mathcal{I}$ is standard so $y = \mathcal{I}[\![\, N \,]\!]$ say, hence $x = \mathcal{I}[\![\, E^\sigma N \,]\!]$. Thus $\mathcal{I}$ is universal.

The fact that such enumerators exist is of some interest in its own right. In the case of PCF^{++}, suitable terms E^σ are already defined in [14], but for PCF and PCF^{+} we need a different technique. The method we use is, essentially, to construct a "simulator" for the relevant language within itself. A very similar method has recently (and independently) been employed by Abramsky *et al.* [1] to prove a definability result for an interpretation of sequential PCF based on *games*; a similar result has been proved by Hyland and Ong [8]. This yields an alternate semantic proof of the existence of enumerators for PCF, analogous to that in [14] for PCF^{++}. It is worth remarking that there is no corresponding definability result for PCF^{+}. It may well be that there can be none; it is not at all clear, however, how to even formulate a precise statement to that effect. In what follows, **L** stands for either PCF or PCF^{+}, and $\lceil - \rceil$ is some effective Gödel-numbering of **L**-terms as natural numbers.

Proposition 6 *For each type σ there exists a closed term $E^\sigma : \iota \to \sigma$ of* **L** *such that $E^\sigma \lceil M \rceil \approx M$ for all closed terms $M : \sigma$ of* **L**.

We now fix $\sigma = \sigma_1 \to \cdots \to \sigma_h \to \iota$ ($h \geq 0$) and consider the construction of E^σ; we will not be completely explicit as all we require is its existence. The basic idea is that—given closed terms $M : \sigma$ and $N_j : \sigma_j$, for $j = 1, \ldots, h$—the computation of $E^\sigma \lceil M \rceil N_1 \ldots N_h$ will simulate the reduction of $M N_1 \ldots N_h$ via the Gödel-numbering. The problem here is that we do not have access to the Gödel-numbers of the N_j, but only to the terms themselves; so, in fact, we symbolically reduce (via Gödel-numbers) the term $M x_1 \ldots x_h$, where the x_j are variables used to stand for the N_j. This results in a further problem when, in the course of the symbolic reduction, we come to a term of the form $x_j M_1 \ldots M_a$. In this case we do not reduce, but rather interpret, "passing" suitable simulations of $M_1, \ldots, M_a$ to N_j.

For the idea to work, it turns out that we need variables not only of the types σ_j, but also of all their subtypes. Let us define the relation $\prec$ between types to be the transitive relation generated from all instances of $\gamma_j \prec (\gamma_1 \to \cdots \to \gamma_m \to \iota)$, for $j = 1, \ldots, m$. Let $\tau_1, \ldots, \tau_k$ be the (possibly repetitive) enumeration of all types $\tau \prec \sigma$ in *breadth-first* order (regarding

σ as a binary tree). Notice that for each i with $1 \leq i \leq k$ there exist $p_i \leq q_i$ such that $\tau_i = \tau_{p_i+1} \to \cdots \to \tau_{q_i} \to \iota$; we drop the subscripts on p and q when they are clear from the context. Note also that $\tau_i = \sigma_i$ for $1 \leq i \leq h$.

To simulate $MN_1 \ldots N_h$ we may need arbitrarily many variables of each type τ_i. We thus suppose that we have a countably infinite supply of variables $x_i^j : \tau_i$ for each i, and that the mapping $(i, j) \mapsto \lceil x_i^j \rceil$ is recursive; we say a term is *σ-open* if its free variables are among the x_i^j. To "store" the "values" of these variables we will use closed terms $F_i : \iota \to \tau_i$, where $F_i\, j$ stores the value of x_i^j. Since at any given stage only finitely many variables will be in use, we need a way to introduce new variables. For each i we let push_i be $\lambda v^{\tau_i} f^{\iota \to \tau_i} j^{\iota} \lambda y_{p+1}^{\tau_{p+1}} \ldots y_q^{\tau_q}. \mathsf{cond}\ j\ (v y_{p+1} \ldots y_q)\ (f(\mathsf{pred}\ j) y_{p+1} \ldots y_q)$. The effect of $\mathsf{push}_i\, VF$ is to store the value given by a closed term V in the "register" x_i^0, and the previous value of x_i^j in the register x_i^{j+1}; we have $\mathsf{push}_i\, VF0 \approx V$ and $\mathsf{push}_i\, VF(j+1) \approx Fj$. To compensate for the use of push_i, we also need an operation $\uparrow_i$ on terms of $\mathbf{L}$ that "bumps up" the indices on the appropriate variables; the term $N{\uparrow_i}$ is defined to be the term obtained by the capture-avoiding simultaneous substitution of V_i^j for x_i^j for every x_i^j occurring freely in N. Clearly the mapping $\lceil N \rceil \mapsto \lceil N{\uparrow_i} \rceil$ is partial recursive.

So, to define E^σ we construct an "ι-simulator" $S : \rho$ (where ρ is the type $(\iota \to \tau_1) \to \cdots \to (\iota \to \tau_k) \to \iota \to \iota$) such that if $N : \iota$ is a σ-open term then $SF_1 \ldots F_k \lceil N \rceil$ simulates N, taking x_i^j to stand for $F_i\, j$. Following the idea outlined above, S will perform repeated *one-step* reductions, but terms of the form $x_i^j N_{p+1} \ldots N_q$ are interpreted by passing simulations of the arguments $N_{p+1}, \ldots, N_q$ to $F_i\, j : \tau_i$. For this, we need "τ_i-simulators" S_i of type $\rho_i = (\iota \to \tau_1) \to \cdots \to (\iota \to \tau_k) \to \iota \to \tau_i$; these are defined from the ι-simulator $S : \rho$ by the terms $\Theta_i : \rho \to \rho_i$ given in Lemma 7, below. Formally, S is obtained as a fixed-point of the term $\Phi : \rho \to \rho$ given in Lemma 8, below; its definition makes use of the Θ_i, and the terms $R : \rho$ used there and in Lemma 7 are to be thought of as "approximants" to S.

We need some special notation. First, for any vector $\vec{F}$ of terms $F_1, \ldots, F_k$ and term M we write $M\vec{F}$ for $(\ldots(MF_1)\ldots F_k)$. Second, for any such vector we abbreviate $F_1, \ldots, F_p, (\mathsf{push}_{p+1} V_{p+1} F_{p+1}), \ldots, (\mathsf{push}_q V_q F_q), F_{q+1}, \ldots, F_k$ to $\mathsf{push}_{p,q}(V_{p+1}, \ldots, V_q; \vec{F})$, where $1 \leq p \leq q \leq k$. Third, for any term M and $1 \leq p \leq q \leq k$ we write $M{\uparrow_{p,q}}$ for $(\ldots(M{\uparrow_{p+1}})\ldots{\uparrow_q})$. Finally, the notation $C[|N_1|, \ldots, |N_a|] \to_d^+ C'[|N_1'|, \ldots, |N_{a'}'|]$ means that given PCF terms $U_1, \ldots, U_a$ such that $U_j \to^* N_j$ $(1 \leq j \leq a)$ there exist PCF terms $U_1', \ldots, U_{a'}'$ such that $U_{j'}' \to^* N_{j'}'$ $(1 \leq j' \leq a')$ and $C[U_1, \ldots, U_a] \to_d^+ C'[U_1', \ldots, U_{a'}']$; notice that the U_j and $U_{j'}'$ must be closed. (Here $C[\ \ , \ldots, \ \ ]$ and $C'[\ \ , \ldots, \ \ ]$ are "multi-place" contexts and $\to_d$ is the deterministic one-step reduction relation defined in Section 2.) This is useful because the *call-by-name* evaluation mechanism of $\mathbf{L}$ means that we cannot force subterms such as the

$U'_{j'}$ to be evaluated when we would like. The consideration of deterministic reduction (and hence of PCF terms) is only needed for Lemma 11 below. Note that transitivity holds: if $C[|N_1|,\ldots,|N_a|] \to_d^+ C'[|N'_1|,\ldots,|N'_{a'}|] \to_d^+ C''[|N''_1|,\ldots,|N''_{a''}|]$ then $C[|N_1|,\ldots,|N_a|] \to_d^+ C''[|N''_1|,\ldots,|N''_{a''}|]$.

Lemma 7 *There exist closed* $\mathbf{L}$*-terms* $\Theta_i : \rho \to \rho_i$ *for* $1 \le i \le k$*, such that for all* σ*-open* $N : \tau_i$ *and closed* $R:\rho, F_1:\iota\to\tau_1,\ldots,F_k:\iota\to\tau_k$ *we have*

$$\Theta_i R\vec{F}\ |\lceil N\rceil| \quad \to_d^+ \quad \lambda y_{p+1}^{\tau_{p+1}}\ldots y_q^{\tau_q}.\ R\,\mathsf{push}_{p,q}(y_{p+1},\ldots,y_q;\vec{F})\ |\lceil (N{\uparrow}_{p,q})x^0_{p+1}\ldots x^0_q\rceil|\ .$$

Proof Define Θ_i to be the term

$$\lambda r^\rho f_1^{\iota\to\tau_1}\ldots f_k^{\iota\to\tau_k} z^\iota y_{p+1}^{\tau_{p+1}}\ldots y_q^{\tau_q}.\ r\ \mathsf{push}_{p,q}(y_{p+1},\ldots,y_q;\vec{f})(Gz)$$

where $G : \iota \to \iota$ is a closed PCF term such that for any σ-open $N : \tau_i$, $G\lceil N\rceil \to^+ \lceil (N{\uparrow}_{p,q})x^0_{p+1}\ldots x^0_q\rceil$. □

We now consider the term Φ. Notice that the clauses given below cover all syntactic shapes for terms of type ι. (The clause marked † applies only to PCF$^+$.)

Lemma 8 *There exists a closed* $\mathbf{L}$*-term* $\Phi : \rho \to \rho$ *such that for all closed terms* $R:\rho, F_1:\iota\to\tau_1,\ldots,F_k:\iota\to\tau_k$ *we have*

$$\begin{array}{lll}
& \Phi R\vec{F}\,|\lceil n\rceil| & \to_d^+ n;\\
& \Phi R\vec{F}\,|\lceil \mathsf{succ}\ M\rceil| & \to_d^+ \mathsf{succ}\ (R\vec{F}\,|\lceil M\rceil|);\\
& \Phi R\vec{F}\,|\lceil \mathsf{pred}\ M\rceil| & \to_d^+ \mathsf{pred}\ (R\vec{F}\,|\lceil M\rceil|);\\
& \Phi R\vec{F}\,|\lceil \mathsf{cond}\ LMN\rceil| & \to_d^+ \mathsf{cond}\ (R\vec{F}\,|\lceil L\rceil|)(R\vec{F}\,|\lceil M\rceil|)(R\vec{F}\,|\lceil N\rceil|);\\
& \Phi R\vec{F}\,|\lceil \mathsf{Y}_\tau N_1 N_2\ldots N_a\rceil| & \to_d^+ R\vec{F}\,|\lceil N_1(\mathsf{Y}_\tau N_1)N_2\ldots N_a\rceil| \quad (a\ge 2);\\
& \Phi R\vec{F}\,|\lceil (\lambda z^\tau.M)N_1\ldots N_a\rceil| & \to_d^+ R\vec{F}\,|\lceil M[N_1/z]N_2\ldots N_a\rceil| \quad (a\ge 1);\\
\dagger & \Phi R\vec{F}\,|\lceil \mathsf{por}\ MN\rceil| & \to_d^+ \mathsf{por}\ (R\vec{F}\,|\lceil M\rceil|)(R\vec{F}\,|\lceil N\rceil|);\\
& \Phi R\vec{F}\ |\lceil x_i^j N_{p+1}\ldots N_q\rceil| & \to_d^+ F_i\,j\ (\Theta_{p+1}R\vec{F}\ |\lceil N_{p+1}\rceil|)\ldots(\Theta_q R\vec{F}\ |\lceil N_q\rceil|).
\end{array}$$

Proof (Hint) We construct Φ via a "case split" with at most $(k+7)$ cases. The need for the consideration of PCF terms arises here, in order to ensure deterministic reduction. □

Note that terms of the forms $\mathsf{succ}\ M$, $\mathsf{pred}\ M$, $\mathsf{cond}\ LMN$ or $\mathsf{por}\ MN$ are interpreted rather than symbolically reduced; this is needed to handle terms such as $\mathsf{succ}\ x_i^j N_{p+1}\ldots N_q$ where the operator must be interpreted since an argument is. It is also worth noting that there is no way to interpret terms such as $\mathsf{Y}_\tau N_1 N_2\ldots N_a$ or $(\lambda z^\tau.M)N_1\ldots N_a$ as the type τ there is arbitrary and our method enables us to deal only with a finite number of given types (here the τ_i).

We now define $S = \mathsf{Y}_\rho\Phi$ and $S_i = \Theta_i S$, for $i = 1, \ldots, k$. Note that:

$$\begin{array}{ll} S\vec{F}\ |\lceil n\rceil| & \to_d^+ n; \\ S\vec{F}\ |\lceil \mathsf{succ}\ M\rceil| & \to_d^+ \mathsf{succ}\ (S\vec{F}\ |\lceil M\rceil|); \end{array}$$

etc. and that:

$$S_i\vec{F}\ |\lceil N\rceil| \ \to_d^+\ \lambda y_{p+1}^{\tau_{p+1}} \ldots y_q^{\tau_q}.\, S\mathsf{push}_{p,q}(y_{p+1}, \ldots, y_q; \vec{F})\ |\lceil (N{\uparrow}_{p,q})x_{p+1}^0 \ldots x_q^0\rceil|\,.$$

Finally, we take $E^\sigma : \iota \to \sigma$ to be a closed term such that

$$E^\sigma\lceil M\rceil \ \to_d^+\ \lambda y_1^{\sigma_1} \ldots y_h^{\sigma_h}.\, S\mathsf{push}_{0,h}(y_1, \ldots, y_h; \vec{\Omega})\ |\lceil M x_1^0 \ldots x_h^0\rceil|$$

where $\vec{\Omega}$ is $\Omega_{\iota\to\tau_1}, \ldots, \Omega_{\iota\to\tau_k}$.

We need to prove that $E^\sigma\lceil M\rceil \approx M$ for all closed $M : \sigma$. By the Context Lemma, it is enough to show that for all closed $N_1 : \sigma_1, \ldots, N_h : \sigma_h$ we have $E^\sigma\lceil M\rceil N_1 \ldots N_h \to^* n$ iff $M N_1 \ldots N_h \to^* n$. Clearly the following lemma suffices:

Lemma 9 *Suppose $F_i\, j \approx V_i^j : \tau_i$ for each i, j, where the F_i and V_i^j are closed. Then for all σ-open $N : \iota$ and PCF terms U such that $U \to^* \lceil N\rceil$ we have $S\vec{F}U \to^* n$ iff $N[V_i^j/x_i^j] \to^* n$.*

The notation $N[V_i^j/x_i^j]$ denotes the term obtained from N by the simultaneous substitution of V_i^j for x_i^j, for every x_i^j occurring freely in N.

The lemma is proved by relating the possible reduction sequences of $N[V_i^j/x_i^j]$ with those of its simulation $S\vec{F}\lceil N\rceil$. Define "encoding" relations $\rhd$ between terms of type ι, $\rhd_i$ between terms of type τ_i, for $i = 1, \ldots, k$, and $\unrhd$ between terms of the same type as follows:

- if $U \to^* \lceil N\rceil$ where U is a PCF term and $N : \iota$ is a σ-open term and if $F_i j \approx V_i^j : \tau_i$ for each i, j, where the F_i and V_i^j are closed, then $S\vec{F}U \rhd N[V_i^j/x_i^j]$;
- if $U \to^* \lceil N\rceil$ where U is a PCF term and $N : \tau_i$ is a σ-open term and if $F_i j \approx V_i^j : \tau_i$ for each i, j, where the F_i and V_i^j are closed, then $S_i\vec{F}U \rhd_i N[V_i^j/x_i^j]$;
- if $P_s \rhd Q_s$ or $P_s \rhd_i Q_s$ (for some $1 \le i \le k$) for $1 \le s \le r$, then, for any r-place context $C[\ \ , \ldots, \ \ ]$, $C[P_1, \ldots, P_r] \unrhd C[Q_1, \ldots, Q_r]$.

Note that $\unrhd$ is reflexive; note too that $\unrhd$ is closed under substitution in the sense that if $P \unrhd Q$ and $P' \unrhd Q' : \tau$ then $P[P'/z^\tau] \unrhd Q[Q'/z^\tau]$. We write $C[|\lceil N_1\rceil|, \ldots, |\lceil N_a\rceil|] \unrhd Q$ (where $a \ge 0$), to mean that for any PCF terms $U_1, \ldots, U_a$ such that $U_j \to^* \lceil N_j\rceil$ $(j = 1, \ldots, a)$ we have $C[U_1, \ldots, U_a] \unrhd Q$.

Lemma 9 is an immediate consequence of the next two lemmas.

Lemma 10 *If $P\!:\!\iota$ is closed, $P \trianglerighteq Q$ and $Q \rightarrow^* n$ then $P \rightarrow^* n$*

Proof The proof proceeds by induction on the length l of a shortest reduction sequence from Q to n. If this is 0, then either P is n or else $P \rhd n$, and so, by the remarks after Lemma 8, $P \rightarrow_d^+ n$. For $l > 0$, fixing P and Q, we first note that there is an r-place context $C[\ \ ,\ldots,\ \ ]$ ($r \geq 0$) and there are P_s and Q_s ($1 \leq s \leq r$) such that $Q = C[Q_1,\ldots,Q_r]$, $P = C[P_1,\ldots,P_r]$ and for $1 \leq s \leq r$, $P_s \rhd Q_s$ or $P_s \rhd_i Q_s$, for some $1 \leq i \leq k$.

There are three main cases. In the first two $C[\ \ ,\ldots,\ \ ]$ has the form $[\ \ ]C_1[\ \ ,\ldots,\ \ ]\ldots C_t[\ \ ,\ldots,\ \ ]$—that is, there is a context hole in "head position"; the third is where there is not. In the first case, the proof proceeds either by reducing the length of the shortest reduction sequence (and applying the induction hypothesis) or else by reducing to the third case, with the same reduction sequence. The second case reduces to the first, with the same reduction sequence. In the third case, the length is always reduced.

So let us suppose there is indeed a context hole in head position. The first case is where $C[\vec{Q}] = Q_s = Q$ and $C[\vec{P}] = P_s = P$, for some $1 \leq s \leq r$, and $P \rhd Q$ (and $t = 0$). The second case is where $C[\vec{Q}] = Q_s C_1[\vec{Q}]\ldots C_t[\vec{Q}]$ and $C[\vec{P}] = P_s C_1[\vec{P}]\ldots C_t[\vec{P}]$, for some $1 \leq s \leq r$, and $P_s \rhd_i Q_s$, for some $1 \leq i \leq k$ (and $t = q - p$).

Let us now consider the first case. Here $P = S\vec{F}U$ and $Q = N[V_i^j/x_i^j]$ where $N\!:\!\iota$ is σ-open, where $F_i j \approx V_i^j\!:\!\tau_i$ for each i, j, where the F_i and V_i^j are closed, and where U is a PCF term such that $U \rightarrow^* \lceil N \rceil$. The proof now proceeds according to the form of N; it is here that the workings of the simulator are seen.

First, let us suppose $N = \mathsf{por}\ N_1 N_2$. Then, by the remarks after Lemma 8 we have $S\vec{F}U \rightarrow_d^+ \mathsf{por}\ (S\vec{F}U_1)(S\vec{F}U_2)$, where U_1 and U_2 are PCF terms such that $U_1 \rightarrow^* \lceil N_1 \rceil$ and $U_2 \rightarrow^* \lceil N_2 \rceil$. Therefore $S\vec{F}U_1 \rhd N_1[V_i^j/x_i^j]$ and similarly for U_2. Now, as $(\mathsf{por}\ N_1 N_2)[V_i^j/x_i^j] = N[V_i^j/x_i^j]$ reduces to n in l steps, then, in $< l$ steps, either one of $N_1[V_i^j/x_i^j]$, $N_2[V_i^j/x_i^j]$ reduce to 0 or both reduce to a positive numeral. So we may apply the induction hypothesis and obtain corresponding reductions of $S\vec{F}U_1$ and $S\vec{F}U_2$, and hence of $S\vec{F}U$. The cases where N has any of the forms $\mathsf{succ}\ N_1$, $\mathsf{pred}\ N_1$ or $\mathsf{cond}\ N_1 N_2 N_3$ are similar.

Next, let us suppose that $N = (\lambda z^\tau.M)N_1\ldots N_a$. Then we have that $S\vec{F}U \rightarrow_d^+ S\vec{F}\ |\lceil M[N_1/z]N_2\ldots N_a \rceil| \trianglerighteq (M[N_1/z]N_2\ldots N_a)[V_i^j/x_i^j] = Q'$, say. We can now apply the induction hypothesis, as there is a reduction of Q' to n in $l-1$ steps since we have the *deterministic* reduction $Q \rightarrow_d Q'$. The case where $N = \mathsf{Y}_\tau N_1 N_2\ldots N_a$ is similar.

Finally, suppose $N = x_i^j N_{p+1}\ldots N_q$. Here we have $P = S\vec{F}U \rightarrow_d^+ F_i\, j\, (S_{p+1}\vec{F}U_{p+1})\ldots(S_q\vec{F}U_q)$, where, for $p < i' \leq q$, $U_{i'}$ is a PCF term such that $U_{i'} \rightarrow^* N_{i'}$. But then, we have that $V_i^j(S_{p+1}\vec{F}U_{p+1})\ldots(S_q\vec{F}U_q) \trianglerighteq$

$V_i^j(N_{p+1}[V_i^j/x_i^j])\dots(N_q[V_i^j/x_i^j]) = Q$, and we are in the third case with the same shortest reduction sequence. So, $V_i^j(S_{p+1}\vec{F}U_{p+1})\dots(S_q\vec{F}U_q) \to^* n$, by induction, and then, as $F_i\, j \approx V_i^j$, $P \to^+ n$.

In the second case, $P = S_i\vec{F}U\vec{C}[\vec{P}]$ (abbreviating $C_1[\ \ ],\dots,C_t[\ \ ]$ to $\vec{C}[\ \ ]$) and $Q = N[V_i^j/x_i^j]\vec{C}[\vec{Q}]$ where $N{:}\tau_i$ is σ-open, where $F_i j \approx V_i^j : \tau_i$ for each i,j, where the F_i and V_i^j are closed, and where U is a PCF term such that $U \to^* \lceil N\rceil$. Then $S_i\vec{F}U \to^+ \lambda y_{p+1}^{\tau_{p+1}}\dots y_q^{\tau_q}.\, S\mathsf{push}_{p,q}(y_{p+1},\dots,y_q;\vec{F})U'$ where U' is a PCF term such that $U' \to^* \lceil(N{\uparrow}_{p,q})x_{p+1}^0\dots x_q^0\rceil$. So we have that $S_i\vec{F}U\vec{C}[\vec{P}] \to^+ S\mathsf{push}_{p,q}(\vec{C}[\vec{P}];\vec{F})U'$.

Now, for $i' = 1,\dots,k$, define $W_{i'}^j$ so that, for $i' = p+1,\dots,q$, $W_{i'}^0$ is $C_{i'}[\vec{Q}]$, $W_{i'}^{j+1}$ is $V_{i'}^j$ and, for all other i', $W_{i'}^j$ is $V_{i'}^j$. Then we have that $S\mathsf{push}_{p,q}(\vec{C}[\vec{P}];\vec{F})U' \rhd ((N{\uparrow}_{p,q})x_{p+1}^0\dots x_q^0)[W_{i'}^j/x_{i'}^j] = N[V_{i'}^j/x_{i'}^j]\vec{C}[\vec{Q}]$, and we have reduced to the first case, with the same reduction sequence.

Finally, we consider the third case, where no hole is in "head" position in $C[\ \ ,\dots,\ \ ]$ (which we abbreviate to $C[\ \ ]$). The proof divides into subcases according to the form of $C[\ \ ]$ We consider two of these; the others are similar. The first is where $C[\ \ ]$ is $\mathsf{por}\, C_1[\ \ ]C_2[\ \ ]$. Here we have reductions to numerals in $< l$ steps of one or both of $C_1[\vec{(Q)}]$, $C_2[\vec{(Q)}]$, as in the previous case involving por, and we can again apply the induction hypothesis. The second is where $C[\ \ ]$ is $(\lambda z^\tau.C'[\ \ ])C_1[\ \ ]\dots C_a[\ \ ]$. Here $P = (\lambda z^\tau.C'[\vec{P}])C_1[\vec{P}]\dots C_a[\vec{P}] \to (C'[\vec{P}][C_1[\vec{P}]/z])C_2[\vec{P}]\dots C_a[\vec{P}] \trianglerighteq (C'[\vec{Q}][C_1[\vec{Q}]/z])C_2[\vec{Q}]\dots C_a[\vec{Q}] = Q'$, say (we use the closure of $\trianglerighteq$ under substitution here). We may now apply the induction hypothesis, for, as $Q \to_d Q'$, there is a reduction of Q' to n in $l-1$ steps. $\Box$

Lemma 11 *If $P{:}\iota$ is closed, $P \trianglerighteq Q$ and $P \to^* n$ then $Q \to^* n$.*

Proof By induction on the length of shortest reduction sequence from P to n. If this is 0, the result is immediate. Otherwise, the cases are organised as in the proof of Lemma 10(ii), but—unlike there—the length is always reduced. Let us consider the first case, where $P = S\vec{F}U$ and $Q = N[V_i^j/x_i^j]$ where $N{:}\iota$ is σ-open, where $F_i\, j \approx V_i^j : \tau_i$ for each i,j, where the F_i and V_i^j are closed, and where U is a PCF term such that $U \to^* \lceil N\rceil$.

Let us first suppose $N = \mathsf{por}\ N_1N_2$. Then $S\vec{F}U \to_d^+ \mathsf{por}\ (S\vec{F}U_1)(S\vec{F}U_2)$, where U_1, U_2 are PCF terms which reduce to $\lceil N_1\rceil$ and $\lceil N_2\rceil$, respectively. As the reduction from P to $\mathsf{por}\ (S\vec{F}U_1)(S\vec{F}U_2)$ is deterministic, the shortest reduction from P to n must proceed via $\mathsf{por}\ (S\vec{F}U_1)(S\vec{F}U_2)$, and the proof is now similar to that of Lemma 10. The cases where N has any of the forms $\mathsf{succ}\, N_1$, $\mathsf{pred}\, N_1$ or $\mathsf{cond}\ N_1N_2N_3$ are similar.

Next, let us suppose that $N = (\lambda z^\tau.M)N_1\dots N_a$. Then we have that $S\vec{F}U \to_d^+ S\vec{F}U'$ where U' is a PCF term such that $U' \to^* \lceil M[N_1/z]N_2\dots N_a\rceil$. So $S\vec{F}U' \trianglerighteq (M[N_1/z]N_2\dots N_a)[V_i^j/x_i^j]$, and we may now apply the induction hypothesis as there is a shorter reduction sequence of $S\vec{F}U'$ to a numeral. The case where $N = \mathsf{Y}_\tau N_1 N_2 \dots N_a$ is similar.

Finally, suppose $N = x_i^j N_{p+1} \dots N_q$. Here we have $P = S\vec{F}U \to_d^+ F_i\, j\, (S_{p+1}\vec{F}U_{p+1}) \dots (S_q\vec{F}U_q) = P'$, say, where, for $p < i' \leq q$, $U_{i'}$ is a PCF term such that $U_{i'} \to^* N_{i'}$. But $P' \trianglerighteq F_i\, j\, (N_{p+1}[V_i^j/x_i^j]) \dots (N_q[V_i^j/x_i^j])$, and so, by induction, we have that $F_i\, j\, (N_{p+1}[V_i^j/x_i^j]) \dots (N_q[V_i^j/x_i^j]) \to^* n$. But then as $F_i\, j \approx V_i^j$ and $V_i^j(N_{p+1}[V_i^j/x_i^j]) \dots (N_q[V_i^j/x_i^j]) = Q$ we have that $Q \to^* n$.

In the second case, $P = S_i\vec{F}U\vec{C}[\vec{P}]$ and $Q = N[V_i^j/x_i^j]\vec{C}[\vec{Q}]$ where $N : \tau_i$ is σ-open, where $F_i j \approx V_i^j : \tau_i$ for each i, j, where the F_i and V_i^j are closed, and where U is a PCF term such that $U \to^* \lceil N \rceil$. We have that $S_i\vec{F}U\vec{C}[\vec{P}] \to_d^+ S\mathsf{push}_{p,q}(\vec{C}[\vec{P}]; \vec{F})U'$ where U' is a PCF term such that $U' \to^* \lceil (N{\uparrow_{p,q}})x_{p+1}^0 \dots x_q^0 \rceil$. Now, defining $W_{i'}^j$ as before, we see that $S\mathsf{push}_{p,q}(\vec{C}[\vec{P}]; \vec{F})U' \rhd ((N{\uparrow_{p,q}})x_{p+1}^0 \dots x_q^0)[W_{i'}^j/x_{i'}^j] = N[V_{i'}^j/x_{i'}^j]\vec{C}[\vec{Q}]$, and we may apply the induction hypothesis.

Finally, in the third case no hole is in head position in $C[\quad]$ and the proof again proceeds according to the form of $C[\quad]$; the details are omitted. $\square$

The proof of Theorem 3 is complete.

5 Remarks and Open Problems

We conclude by drawing together some miscellaneous observations and suggesting some directions for further research.

It is easy to show using Proposition 5 that standard LFA interpretations for each of our languages $\mathbf{L}$ do in fact exist. Specifically, let $\mathcal{C}_\mathbf{L}$ be the "syntactic category" whose objects are environments Γ, and whose morphisms from Γ to Δ are appropriate tuples of terms in environment Γ modulo observational equivalence. Then the canonical standard interpretation $\mathcal{I}_\mathbf{L}$ of $\mathbf{L}$ in $\mathcal{C}_\mathbf{L}$ is clearly EFA and universal, and thus LFA. In fact $\mathcal{I}_\mathbf{L}$ is essentially the *only* "sensible" LFA interpretation for $\mathbf{L}$. For suppose that $\mathcal{I}$ is a standard and cartesian-closed LFA interpretation of $\mathbf{L}$ in $\mathcal{D}$. Then, by Theorem 3 and a previous remark, it is EFA and universal in the strong sense. It follows that the full subcategory of $\mathcal{D}$ consisting of the objects $\mathcal{I}[\![\, \sigma \,]\!]$ is equivalent to $\mathcal{C}_\mathbf{L}$, and then that $\mathcal{I}_\mathbf{L}$ and $\mathcal{I}$ are identical, modulo the equivalence.

Given that standard LFA interpretations exist, one can prove, using an appropriate form of the Upward Löwenheim-Skolem Theorem, that *non-standard* (and thus non-universal) LFA interpretations also exist. This shows that the standardness condition in Theorem 3 is indeed necessary.

The syntactic interpretations $\mathcal{C}_\mathbf{L}$ assure us of the existence of LFA interpretations, but these interpretations may not be very useful since questions about $\mathcal{C}_{\mathbf{L}}$ are no easier than questions about $\mathcal{L}$ itself. It is more interesting to ask whether one can give more "semantic" constructions of LFA interpretations. Note that since any standard LFA interpretation is universal it must have some notion of "computability" built in; the classical category of cpos does *not* provide an LFA interpretation for PCF^{++}, for instance, because of the existence of non-computable elements (this observation is sharpened in Proposition 12 below). For PCF^{++}, there are several natural examples of LFA interpretations: the category of *effective Scott domains* [14] and many *realizability interpretations* [11] provide instances. Examples of LFA interpretations for sequential PCF are given by the recursive versions of categories of games [1, 8]. Note in passing that the evident r.e. sub-interpretation of Milner's EFA interpretation for PCF [12] does *not* provide an LFA interpretation, as there exist first-order functions that are effective and sequential but not PCF-definable (see e.g., [19]). We do not know of any natural LFA interpretations for PCF^{+}.

Although in this paper we have concentrated mainly on LFA interpretations for the whole of $\mathbf{J_L}$, it is also natural to consider logical full abstraction for fragments of the language. One way to obtain such a fragment is to restrict attention to sentences of a certain *logical complexity*, e.g., the Π_n-sentences, for some n. (Note that, by a previous remark, logical full abstraction for Π_n-sentences and Σ_n-sentences are equivalent.) Our proof of Theorem 3 shows that if a standard interpretation is LFA for Π_2-sentences then it is LFA for the whole of $\mathbf{J_L}$. In fact, the proof shows more, that it suffices to be LFA for Π_2-sentences with equational matrix, that is, of the form $\forall x^{\sigma}.\exists y^{\tau}.\, M = N$—one can even take τ to be ι.

The next result shows that logical full abstraction for Σ_1-sentences with equational matrix is already stronger than equational full abstraction (for any of PCF^{+}, PCF^{+} or PCF^{++}).

Proposition 12 *Neither Milner's EFA interpretation for PCF, nor the standard cpo interpretation, whether taken for PCF^{+} or PCF^{++}, are LFA for sentences of the form $\exists f^{\iota\to\iota}.\, M = N$.*

Proof Let $\mathcal{K}$ denote Kleene's singular tree (see [2, Chapter IV])—recall that $\mathcal{K}$ is a recursive prefix-closed set of finite binary sequences such that $\mathcal{K}$ contains arbitrarily long finite sequences but no *recursive* infinite path. Let $\lceil - \rceil$ be an effective coding of finite binary sequences as natural numbers, and let $T : \iota \to \iota$ be such that $T\lceil s\rceil \to^{+} 0$ iff $s \in \mathcal{K}$. We also require a term $P : \iota \to \iota \to \iota$ such that $P\lceil s\rceil\lceil t\rceil \to^{+} 0$ iff s is a proper prefix of t, and a term $Z : \iota \to \iota \to \iota$ such that $Z\, m\, n \to^{+} 0$ iff $m = n = 0$. Now consider the sentence

$$\exists f^{\iota\to\iota}.\ \ (\lambda x^{\iota}.\, Z\ (T(fx))\ (P(fx)(f(\mathsf{succ}\ x))))\ =\ (\lambda x^{\iota}.\, \mathsf{cond}\ x\, 0\, 0).$$

It is easy to see that this sentence is denotationally true in all the interpretations since by König's Lemma there exists an infinite path through $\mathcal{K}$, but not operationally true as there is no recursive such path (Milner's interpretation coincides with the cpo interpretation at type $\iota \to \iota$). $\Box$

The (open) problem is now to distinguish LFA for Π_1-sentences from LFA.

Another way to obtain fragments of $\mathbf{J_L}$ is via a notion of *type complexity.* The *level* of a type is defined recursively:

$$\text{level}(\iota) = 0;\ \text{level}(\sigma \to \tau) = \max(\text{level}(\sigma) + 1, \text{level}(\tau)).$$

The level of a formula is then taken to be the maximum of the levels of its quantified variables. (One could also consider stronger alternative definitions placing restrictions on the level of subexpressions.) A standard EFA interpretation is (evidently) logically fully abstract for sentences of level 0. For PCF^{++} one can say more, but first we need a lemma. A type σ is said to be an $\mathbf{L}$-*retract* of a type τ if there are closed $\mathbf{L}$-terms $L^\sigma_\tau : \sigma \to \tau$ and $R^\tau_\sigma : \tau \to \sigma$ such that $\lambda x^\sigma . R^\tau_\sigma(L^\sigma_\tau(x)) \approx \lambda x^\sigma . x$ holds in $\mathbf{L}$.

Lemma 13 *Every type is a PCF^{++}-retract of $\iota \to \iota$.*

Proof We use the "effective universality" remarked in [15], that every effectively given coherent ω-continuous cpo is a computable retract of T^ω. In the interpretation $\mathcal{C}$ of PCF^{++} provided by the classical category of cpos, every $\mathcal{C}[\![\,\sigma\,]\!]$ is such a cpo; further T^ω is a computable retract of the cpo $\mathcal{C}[\![\,\iota \to \iota\,]\!]$. Since any computable element of any $\mathcal{C}[\![\,\tau\,]\!]$ is PCF^{++}-definable [14] we therefore have PCF^{++}-terms defining the retracts. The conclusion follows, as the classical interpretation is EFA for PCF^{++}. $\Box$

With this we can see that a standard interpretation $\mathcal{I}$ of PCF^{++} is LFA iff it is for sentences of the form $\forall f^{\iota\to\iota} . \exists m^\iota . M = N$. Any such interpretation must be EFA. But now we can apply the above remarks on LFA for Π_2-sentences, as:

$$\models_\mathcal{I} \forall x^\sigma . \exists m^\iota . M = N \text{ iff } \models_\mathcal{I} \forall f^{\iota\to\iota} . \exists m^\iota . M[R^{\iota\to\iota}_\sigma f/x] = N[R^{\iota\to\iota}_\sigma f/x].$$

It is an open question as to whether PCF or PCF^{+} permit any such reduction in type complexity. It would also be interesting to understand which retractions hold for these languages.

The results in this paper should apply not just to the languages we have considered but to a wider class. As regards functional languages, one would certainly wish to consider the lazy and call-by-value variants of PCF [6, 11]. A further useful extension would be to recursively typed languages, such as FPC [6, 4]. It would then be natural to consider polymorphic extensions of PCF; this seems not to be a straightforward matter. It would be also interesting to formulate an appropriate notion that would allow our results to be presented at their natural level of generality. This should at least include suitable extensions of PCF (in which regard see [9]), and perhaps a greater degree of abstraction is obtainable.

Finally, we have said very little about axioms and inference rules for $\mathbf{J}_{\mathbf{L}}$. It would be useful to work out the details of an axiomatization for our logics and show that our axioms were valid in some LFA interpretation. This would establish that they were also valid under the operational interpretation—thus we would obtain an attractive program logic. It seems that the appropriate axioms would be very similar to those of LCF, with a few additional "effectivity" principles. Of course, in this simple situation one can imagine that the validity of the axioms could be proved just as easily by syntactic methods, without the aid of a denotational interpretation. It would therefore be interesting to carry out a similar programme for more complex programming languages—it seems plausible that here semantic methods might show a distinct advantage over syntactic ones.

Acknowledgments

We would like to thank John Power for many helpful discussions.

References

[1] S. Abramsky, R. Jagadeesan and P. Malacaria, Full Abstraction for PCF (Extended abstract), in *Proceedings of TACS '94*, eds. M. Hagiya and J. Mitchell, LNCS 789, pp. 1–15, Springer-Verlag, Berlin, 1994; see also *Full Abstraction for PCF*, by the same authors, to appear.

[2] M. Beeson, *Foundations of Constructive Mathematics*, Springer-Verlag, Berlin, 1985.

[3] P-L. Curien, *Categorical Combinators, Sequential Algorithms, and Functional Programming*, Birkhäuser, Boston, 1993.

[4] M. Fiore and G. D. Plotkin, An Axiomatisation of Computationally Adequate Domain Theoretic Models of FPC, in *Proceedings of the Ninth Symposium on Logic in Computer Science, Paris*, pp. 92 –102. Washington, IEEE Computer Society Press,1994.

[5] M. Gordon, R. Milner and C. Wadsworth, *Edinburgh LCF*, LNCS 78, Springer-Verlag, Berlin, 1978.

[6] C. A. Gunter, *Semantics of Programming Languages*, MIT Press, Cambridge, 1992.

[7] J. M. E. Hyland, First Steps in Synthetic Domain Theory, in *Category Theory, Proceedings, Como 1990*, eds. A. Carboni, M. C. Pedicchio and G. Rosolini, LNM 1488, pp. 131–157, Springer-Verlag, Berlin, 1990.

[8] J. M. E. Hyland and C.-H. L. Ong, Pi-calculus, Dialogue Games and PCF, in *Proc. 7th ACM Conf. Functional Programming and Computer Architecture*, ACM Press, 1995; see also *On Full Abstraction for PCF: I, II and III*, by the same authors, to appear.

[9] T. Jim and A. R. Meyer, Full Abstraction and the Context Lemma (Preliminary Report), in *Proceedings of TACS '91*, eds. T. Ito and A. R. Meyer LNCS 526, pp. 131–151, Springer-Verlag, Berlin, 1991.

[10] J. Lambek and P. J. Scott, *Introduction to Higher-Order Categorical Logic*, Cambridge University Press, Cambridge, 1986.

[11] J. R. Longley, *Realizability Toposes and Language Semantics*, Ph.D. thesis, University of Edinburgh, LFCS technical report number ECS-LFCS-95-332, 1995.

[12] R. Milner, Fully Abstract Models of Typed λ-calculi, *Theoretical Comp. Sci.*, Vol.4, pp. 1–22, 1977.

[13] P. W. O'Hearn and J. G. Riecke, Kripke Logical Relations and PCF, Invited Lecture: Workshop on Logic Domains and Programming Languages, Darmstadt, 1995, to appear in *Information and Computation.*

[14] G. Plotkin, LCF Considered as a Programming Language, *Theoretical Comp. Sci.*, Vol. 5, pp. 223–255, 1977.

[15] G. Plotkin, T^ω as a Universal Domain, *JCSS*, Vol. 17, pp. 209–236, 1978.

[16] D. Scott, Outline of a Mathematical Theory of Computation, in *Proc. 4th Annual Princeton Conference on Information Sciences and Systems*, pp. 169–176, Princeton University, 1970.

[17] D. Scott and C. Strachey Towards a Mathematical Semantics for Computer Languages, in *Proc. Symp. on Computers and Automata*, Microwave Research Institute Symposia Series, Vol. 21, pp. 19–46, Polytechnic Press, Brooklyn, New York, 1971.

[18] A. Stoughton, Interdefinability of Parallel Operations in PCF, *Theoretical Comp. Sci.*, Vol. 79, pp. 357–358, 1991.

[19] M. B. Trakhtenbrot, On Representation of Sequential and Parallel Functions, in *Proc. 4th Symposium on Mathematical Foundations of Computer Science*, LNCS 32, pp. 411–417, Springer-Verlag, Berlin, 1975.