Kripke Logical Relations and PCF

Peter W. O'Hearn*
School of Computer and Information Science
Syracuse University
Syracuse, New York 13244

Jon G. Riecke
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, New Jersey 07974

Abstract

Sieber has described a model of PCF consisting of continuous functions that are invariant under certain (finitary) logical relations, and shown that it is fully abstract for closed terms of up to third-order types. We show that one may achieve full abstraction at *all* types using a form of "Kripke logical relations" introduced by Jung and Tiuryn to characterize λ-definability.

To appear in *Information and Computation.*
(Accepted, October 1994)

*Supported by NSF grant CCR-92110829.

1 Introduction

The nature of sequential functional computation has fascinated computer scientists ever since Scott remarked on a curious incompleteness phenomenon when he introduced LCF (Logic for Computable Functions) and its continuous function model in 1969 (Scott, 1993). Scott noted that although the functionals definable by terms in PCF—the term language of LCF—admitted a sequential evaluation strategy, there were functions in the model that seemed to require a parallel evaluation strategy. "Sequential evaluation" means, roughly, that computation proceeds in a single thread and any subcomputation finishes before proceeding with another. Scott's example of a non-sequential function is "parallel or", which returns true even if one argument is true and the other diverges. Plotkin explored this phenomenon further, and showed that by enlarging PCF to include determinate parallel facilities related to "parallel or" one could achieve "full abstraction" (Plotkin, 1977). A model is called **fully abstract** if two terms are semantically equal precisely when they are observationally equal, *i.e.*, when they may interchanged in all programs with no difference in observable behaviour. Plotkin also observed that Scott's model is not fully abstract *without* parallel facilities, and left open the problem of finding a fully abstract model for the original, sequential language.

The concept of a "sequentially computable functional" is surely intuitively compelling, but the concept has been notoriously difficult to describe in any abstract semantic sense. Some important advances include Milner's syntactic construction of a fully abstract model and the proof that it is unique under certain reasonable assumptions (Milner, 1977), and good, but not fully abstract models based on sequential algorithms (Berry and Curien, 1982) and stable functions (Berry, 1978); *cf.* (Berry et al., 1985; Meyer and Cosmadakis, 1988; Stoughton, 1988) for more discussion and references. This issue of sequential versus parallel functionals was also partially foreshadowed by developments in basic recursion theory, *e.g.*, in different notions of relative computability—given by Turing and weak Turing reducibility—and the associated notions of functional; *cf.* (Odifreddi, 1989) for more discussion and references.

This paper constructs a new model of PCF. At functional type, the model consists of continuous functions that are invariant under certain kinds of *logical relations*. The relations impose conditions that rule out the parallel functions present in the continuous model. We prove that the model is **inequationally fully abstract**, *i.e.*, denotational approximation coincides with a contextually-defined observational approximation relation. The model is isomorphic to Milner's fully abstract, but syntactically constructed, model.

The logical relation approach to building a model of PCF, *i.e.*, using logical relations to constrain the construction of function types, was pioneered in (Sieber, 1992). Instead of admitting all Scott continuous functions from one type to another as the meaning of function types, Sieber's construction admits only those continuous functions that are invariant under "sequential logical relations". A sequential logical relation has arbitrary but fixed finite arity, and is defined by a base type relation $R^{\mathbf{nat}} \subseteq (\mathbf{N}_\perp \times \ldots \times \mathbf{N}_\perp)$ with certain properties; the base type relation determines higher-type relations $R^{\sigma\to\tau}$ in a standard way. Sieber's model, then, takes only those elements of the continuous function model that are invariant, *i.e.*, only those elements f such that $R^\sigma(f, f, \ldots, f)$ holds for all base relations $R^{\mathbf{nat}}$. Sieber proved that for closed terms M, N of up to third-order type, M and N are observationally equivalent iff $[\![M]\!] = [\![N]\!]$—*i.e.*, the model is fully abstract up to third-order—and left open the problem of whether the model is fully abstract for higher types.

Our model construction resembles Sieber's except in the choice of logical relations: we use a form of Kripke logical relation in place of finitary relations. This is reminiscent of Plotkin's seminal work

(Plotkin, 1980), where binary relations suffice for a λ-definability result for second-order types but Kripke relations are used for definability at higher types. The kind of Kripke relation is a domain-theoretic version of that introduced in (Jung and Tiuryn, 1993), where a quite general definability result for pure simply-typed λ-calculus is obtained. Jung and Tiuryn's relations generalize both those of Sieber and Plotkin, and are themselves a specialization of (unary) logical relations in a functor category. The basic new idea in Jung and Tiuryn's relations lies in the "varying arity" of the relations: the elements of the worlds that index the relations are themselves the indices of the elements of the relation. Thus, for example, in a Kripke structure with a world w of two elements and a world w' of three elements, the logical relation at world w is binary and the relation at world w' is ternary.

Although we concentrate on PCF, the definition of the model and proof of full abstraction apply in wider circumstances. Given ground pointed cpos and a collection of first-order functions, the model definition in section 3 works by choosing 'strict" and "complete" finitary logical relations that preserve the given first-order functions; the proof of definability of finite elements requires that the ground domains are SFP with definable finite projections. Moreover, when the ground domains are consistently complete and satisfy the "articulating" conditions of (Milner, 1977), the resulting model is the unique one identified by Milner. In contrast with Milner's construction, the elements of the model are described directly here, not as an inverse limit of finite cpo's: note especially that the finite projections are used only in the proof of full abstraction, not in the construction of the model. (In the case that the ground domains have infinite height, a slight adjustment is required: the relation R_n^w in section 4 becomes tuples that are lubs of directed sets of definable *finite* elements, and the order-theoretic property of finite elements is used to show the relation directed-complete.)

Based on this generality, one might sense that our results do not provide a further analysis of sequentiality *per se*, but rather concern lifting finitary first-order principles to higher types. Nevertheless, in the case of PCF one further simplification is possible: we can use Sieber's characterization of those relations that are preserved by first-order PCF constants. The result is a construction of the fully abstract model of PCF in which the semantics of types does not mention the interpretation of the first-order constants.

2 Sieber's Sequentiality Relations

We begin by reviewing the basic elements of Sieber's construction. To fix notation, **CPO** denotes the category of cpos and continuous functions. Here, a cpo is a directed-complete poset possessing a least element. $\mathbf{N}_\perp$ is the flat natural numbers. $D \times E$ denotes the componentwise-ordered product of cpos, and $[D \to E]$ the continuous function space with pointwise order. If w is a set and D a cpo, we write $[w \to D]$ for the pointwise-ordered cpo of functions, viewing w as discretely ordered. Finally, define the functions

$$\mathsf{succ} : \mathbf{N}_\perp \to \mathbf{N}_\perp$$
$$\mathsf{succ}(m) = \begin{cases} (m+1) & \text{if } m \in \mathbf{N} \\ \perp & \text{otherwise} \end{cases}$$

$$\mathsf{pred} : \mathbf{N}_\perp \to \mathbf{N}_\perp$$
$$\mathsf{pred}(m) = \begin{cases} (m-1) & \text{if } m \geq 1 \\ \perp & \text{otherwise} \end{cases}$$

$$\mathsf{ifz} \;:\; \mathbf{N}_\perp \times \mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$$
$$\mathsf{ifz}(m,n,p) = \begin{cases} n & \text{if } m = 0 \\ p & \text{if } m > 0 \\ \perp & \text{otherwise} \end{cases}$$

The model in (Sieber, 1992) is based on certain relations on the flat naturals.

Definition 1 Suppose w is a finite set. For any subsets $A \subseteq B \subseteq w$, define $S^w_{A,B} \subseteq [w \to \mathbf{N}_\perp]$ as follows:

$$\text{If } g \in [w \to \mathbf{N}_\perp], \text{ then } g \in S^w_{A,B} \text{ iff } (\exists i \in A.\, g(i) = \perp) \text{ or } (\forall i,j \in B.\, g(i) = g(j)).$$

Then $R \subseteq [w \to \mathbf{N}_\perp]$ is a **sequentiality relation** if it is the intersection of a collection of relations of the form $S^w_{A,B}$.

Here, a finite set determines the arity of a relation, *e.g.*, binary relations are obtained by taking w to be a two element set. We use this non-standard notation to be consistent with the definition of "Kripke relations" below. Sieber's first main result about sequentiality relations is the following

Proposition 2 *A relation $R \subseteq [w \to \mathbf{N}_\perp]$ is a sequentiality relation iff for any $g_1, g_2, g_3 \in R$,*

1. $(\underline{\lambda} i \in w.\, 0) \in R$;
2. $(\underline{\lambda} i \in w.\, \mathsf{succ}(g_1(i))) \in R$;
3. $(\underline{\lambda} i \in w.\, \mathsf{pred}(g_1(i))) \in R$; *and*
4. $(\underline{\lambda} i \in w.\, \mathsf{ifz}(g_1(i), g_2(i), g_3(i))) \in R$,

where, for instance, $(\underline{\lambda} i \in w.\, 0)$ is the function that returns 0 given any argument.

In other words, sequentiality relations are precisely those finitary relations that are invariant under the first-order operations 0, succ, pred and ifz.

Sequentiality relations may be lifted to higher types in the standard way. Let $[\![\mathbf{nat}]\!] = \mathbf{N}_\perp$ and $[\![\sigma \to \tau]\!] = [[\![\sigma]\!] \to [\![\tau]\!]]$. Using the notation of relations as function spaces and starting from a base sequentiality relation $R = R^{\mathbf{nat}}$, we define

$$R^{\sigma \to \tau} = \{g \in [w \to [[\![\sigma \to \tau]\!]]] \mid \text{for any } h \in R^\sigma,\, (\underline{\lambda} i \in w.\, g(i)\,(h(i))) \in R^\tau\}.$$

The strictness and completeness properties of sequentiality relations is preserved by lifting to higher types. This fact and the proposition, together with the "main lemma of logical relations" (Plotkin, 1980), has an important consequence: in the continuous type hierarchy over $\mathbf{N}_\perp$, any element that is λ-definable from 0, succ, pred, ifz, and least fixpoint must be invariant under logical relations induced by sequentiality relations at base type. Stated contrapositively, if an element $d \in [\![\sigma]\!]$ is *not* invariant under R^σ starting from a sequentiality relations, the element d must not be definable. This fact may be used in reasoning about the non-definability of certain elements of the model.

Example 1 (Sieber, 1992) Consider the following variation on parallel or:

$$\mathsf{por} : \mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$$

$$\mathsf{por}(m,n) = \begin{cases} 0 & \text{if } m = 0 \text{ or } n = 0 \\ 1 & \text{if } \perp \neq m > 0 \text{ and } \perp \neq n > 0 \\ \perp & \text{otherwise} \end{cases}$$

Intuitively, por is not sequential because any sequential function of type $\mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$ must be constant or evaluate one of its arguments first. Since $\mathsf{por}(\perp, 0) = \mathsf{por}(0, \perp) = 0$, $\mathsf{por}(\perp, \perp)$ would have to be 0 if por was sequential, but $\mathsf{por}(\perp, \perp) = \perp$. This informal argument can be represented quite directly with the relation $R \subseteq [\{1,2,3\} \to \mathbf{N}_\perp]$, where $d \in R$ iff either $d(1)$ or $d(2) = \perp$ or $d(1) = d(2) = d(3)$; in other words, R here is the sequentiality relation $S^{\{1,2,3\}}_{\{1,2\},\{1,2,3\}}$. In tuple notation, if we consider two elements $d_1 = (0, \perp, \perp)$ and $d_2 = (\perp, 0, \perp)$ in R, applying por componentwise results in a tuple $d_3 = (0, 0, \perp)$ that is not in R.

$$\mathsf{por} \left| \begin{array}{c|c} d_1 & d_2 \\ \hline 0 & \perp \\ \perp & 0 \\ \perp & \perp \end{array} \right| = \left| \begin{array}{c} d_3 \\ \hline 0 \\ 0 \\ \perp \end{array} \right|$$

One may think of the top two rows in the table as testing the strictness (in one argument) condition on functions $\mathbf{N}_\perp \times \mathbf{N}_\perp \to \mathbf{N}_\perp$. Having found that the function is strict in neither argument, we test the constancy property by supplying $\perp$ in the bottom row: the bottom entry of the rightmost column would have to be 0 for it to represent an element of R.

Example 2 Example 1 concerns first-order sequentiality. What is interesting is that we can lift the first-order property represented by R to higher types and reason about higher-order functions. As an example of the properties thus obtained, consider the functional

$$\mathsf{fpor} : [\mathbf{N}_\perp \to \mathbf{N}_\perp] \to \mathbf{N}_\perp$$

$$\mathsf{fpor}(f) = \begin{cases} 0 & \text{if } f(0) = 0 \text{ or } f(1) = 0 \\ 1 & \text{if } \perp \neq f(0) > 0 \text{ and } \perp \neq f(1) > 0 \\ \perp & \text{otherwise} \end{cases}$$

The non-sequentiality of this function corresponds to the failure of the following logical property for the relation $R^{(\mathbf{nat}\to\mathbf{nat})\to\mathbf{nat}}$, written in tuple notation:

> if $\forall (n_1, n_2, n_3) \in R \,.\, (f_1(n_1), f_2(n_2), f_3(n_3)) \in R$
> then $(\mathsf{fpor}(f_1), \mathsf{fpor}(f_2), \mathsf{fpor}(f_3)) \in R$.

A counterexample is given by an argument tuple (f_1, f_2, f_3) where $f_1(0) = f_2(1) = 0$, $f_1(1) = f_2(0) = \perp$ and f_3 constantly $\perp$. It may be argued that this example is still essentially about first-order sequentiality, since the function fpor is a simple variation on por. However, it does illustrate well how the purely first-order properties encoded by R can be effectively lifted to higher types using logical relations. More sophisticated examples of using logical relations for reasoning about PCF may be found in (Sieber, 1992; Stoughton, 1994).

The question naturally arises of how far one can apply this mode of reasoning. Sieber's second main result is that, up to second-order type in the continuous type hierarchy over $\mathbf{N}_\perp$, any element that is invariant under all sequentiality relations is the lub of a directed set of PCF-definable elements. This can be turned into a full abstraction result for closed terms of order-three types by using logical relations to constrain function types. We emphasize again that sequentiality relations themselves codify essentially first-order principles of sequential functions. This result is quite astonishing precisely because lifting these first-order principles furnishes a characterization of PCF definability at second order. The question of what happens at types higher than level two was left open by Sieber. We have not been able to settle this question; presently, whether or not sequentiality relations characterize PCF definability at higher types is unknown. In the following sections we show that full abstraction may be obtained by replacing Sieber's fixed-arity relations with a varying-arity form of Kripke relations due to Jung and Tiuryn.

3 Model of PCF

This section first defines a notion of "Kripke logical relation." Then the logical relations are used to describe a suitable cartesian closed category and the resultant model of PCF.

3.1 Kripke Logical Relations

The usual notion of "Kripke logical relations" (*cf.* (Mitchell, 1990; Plotkin, 1973; Plotkin, 1980; Reynolds, 1983)) extends the definition of logical relations with structure similar to the Kripke semantics of intuitionistic logic. One begins with a poset of *worlds* and a finite fixed arity, and then chooses a relation of that arity at base type for each world that must fit together with the poset structure on worlds; the relations of higher type are determined from the base type relations. Jung and Tiuryn's "Kripke logical relations" (Jung and Tiuryn, 1993) are slightly different in two respects. First, one begins with a *category* of worlds which are sets, generalizing the usual definition based on posets. Second, instead of having some finite, fixed arity, Jung and Tiuryn's relations are sets of *functions*, so, for instance, if w is a world and A is the meaning of the base type, the relation R^w is a subset of $[w \to A]$. Notice that there is no finite arity restriction, and that the arity (size of w) may in fact vary from world to world. Jung and Tiuryn's Kripke logical relations are themselves a special case of the categorical forms of logical relation studied in (Ma and Reynolds, 1992; Mitchell and Scedrov, 1993).

For our model of PCF, we need to extend Jung and Tiuryn's relations to the domain-theoretic setting. Let $\mathbf{C}$ be a small subcategory of the category of sets and functions. Then, formally,

Definition 3 A **C-Kripke relation** on a cpo D is a family of subsets $S^w \subseteq [w \to D]$ indexed by $\mathbf{C}$-objects w (for world) that satisfy the following conditions:

- **Completeness**: S^w is a complete subset of $[w \to D]$, *i.e.*, $\perp \in S^w$ and S^w is closed under least upper bounds of directed sets.
- **Kripke Monotonicity**: If $\varphi : v \xrightarrow{\mathbf{C}} w$ in $\mathbf{C}$ and $g \in S^w$, then $(\varphi; g) \in S^v$.

We often omit the $\mathbf{C}$ from "$\mathbf{C}$-Kripke relation" when no confusion is likely.

We need a few notational conventions. If S_1 is a Kripke relation on cpo D_1, S_2 is a Kripke relation on cpo D_2, and $f : D_1 \to D_2$ is a continuous function, then we write $f : S_1 \to S_2$ if for

all $w \in \mathrm{Ob}(\mathbf{C})$ and $h \in S_1^w$, $(h; f) \in S_2^w$. This definition corresponds to a notion of "morphism of relations" as found in (Ma and Reynolds, 1992; Mitchell and Scedrov, 1993).[1] Similarly, if S is a Kripke relation on D and $a \in D$, then we write $a : S$ (read "a is invariant under the $\mathbf{C}$-Kripke relation S") if for all $w \in \mathrm{Ob}(\mathbf{C})$, $(\underline{\lambda} i \in w.\, a) \in S^w$.

3.2 Kripke Sequentiality Relations

To interpret PCF we consider certain Kripke relations on $\mathbf{N}_\perp$. The following definition adapts Sieber's notion of sequentiality relation to take into account the world structure of Kripke relations.

Definition 4 Suppose R is a $\mathbf{C}$-Kripke relation on $\mathbf{N}_\perp$. For any object w of $\mathbf{C}$ and $A \subseteq B \subseteq w$, define

$$\text{If } g \in [w \to \mathbf{N}_\perp], \text{ then } g \in S^w_{A,B} \text{ iff } (\exists i \in A.\, g(i) = \perp) \text{ or } (\forall i, j \in B.\, g(i) = g(j)).$$

Then R is a **Kripke sequentiality relation** if each R^w is the intersection of a collection of relations of the form $S^w_{A,B}$.

The operations of PCF are invariant under Kripke sequentiality relations, but we have not been able to establish a direct converse to this fact, or a counterexample showing the converse to be false. However, Sieber's characterization of *finitary* sequentiality relations carries over for Kripke relations in which each of the worlds is finite, and these are enough to carry out the proof of full abstraction.

Definition 5 R is a **finitary C-Kripke relation** if each object of the category $\mathbf{C}$ is a finite set.

Note that in a finitary Kripke relation each R^w has finite arity, but there is not necessarily a fixed finite bound on the arities of all the R^w's because the category $\mathbf{C}$ might still have an infinite number of objects. This definition thus generalizes Sieber's sequentiality relations, where in Sieber's relations the category $\mathbf{C}$ has only one object of finite size.

Proposition 6 *Suppose R is a finitary Kripke relation on $\mathbf{N}_\perp$. Then R is a sequentiality relation iff*

1. $0 : R$;
2. $\mathsf{succ} : R \to R$;
3. $\mathsf{pred} : R \to R$; *and*
4. $\mathsf{ifz} : R \times R \times R \to R$,

where $(R \times R \times R)^w = \{\langle f, g, h\rangle : [w \to \mathbf{N}_\perp \times \mathbf{N}_\perp \times \mathbf{N}_\perp] \mid f, g, h \in R^w\}$.

Proof: Since the preservation conditions for morphisms of Kripke relations $f : S_1 \to S_2$ are determined pointwise on the objects of category $\mathbf{C}$, R is invariant under a PCF operation iff each R^w is invariant under that operation. The result follows from Proposition 2. ■

[1] To be more specific, the resultant category is, in the notation of (Ma and Reynolds, 1992), a subcategory of $\mathrm{Rel}(\mathbf{D}, \mathbf{D}^{\mathbf{C}^{\mathrm{op}}}, F)$, where $\mathbf{D}$ is the category of predomains and $F : \mathbf{D} \to \mathbf{D}^{\mathbf{C}^{\mathrm{op}}}$ sends E to $E^{(-)}$.

3.3 Semantic Category

A category suitable for interpreting PCF may be constructed using finitary Kripke sequentiality relations. In the definition we use a quantifier "for all R" to mean "for all subcategories $\mathbf{C}$ of the category **Finset** of finite sets, and all $\mathbf{C}$-Kripke sequentiality relations R." There are no real size difficulties associated with this quantifier, *e.g.*, our model construction could alternatively use a small category that is equivalent to **Finset**. Define the category $\mathcal{SR}$ (for *sequentiality-relation preserving functions*) as follows.

- OBJECTS. An object A consists of a cpo $|A|$ and a $\mathbf{C}$-Kripke relation $A(R)$ on $|A|$ for each subcategory $\mathbf{C}$ of **Finset** and each $\mathbf{C}$-Kripke sequentiality relation R. Objects must also satisfy the

  Concreteness Condition: For all R and all $a \in |A|$, $a : A(R)$.

- MORPHISMS. A morphism $f : A \to B$ is a continuous function $f : |A| \to |B|$ satisfying the

  Uniformity Condition: For all R, $f : A(R) \to B(R)$.

Composition and identities are inherited from **CPO**. $\mathcal{SR}$ is related to the categories defined for giving models of languages with local variables (Sieber, 1993; O'Hearn and Tennent, 1993).

Notice that $\mathcal{SR}$ does not consist of arbitrary continuous functions, certain of which are singled out using relations; rather, we use a parametricity condition to constrain hom sets from the very beginning. The model is therefore *not* a collapse of the full continuous model of PCF using logical relations to pick out certain invariant elements. Instead, the relations constrain the construction of the model so that *all* elements are invariant. Thus, there is no need for quotienting or a collapse to guarantee that all elements of the model are extensional functions—they are already by the definition. Sieber also has a presentation of his model of PCF which does not rely on extensional collapse (Sieber, personal communication, July 1993).

$\mathcal{SR}$ has enough structure to interpret the simply-typed λ-calculus, *i.e.*, it is a cartesian-closed category. The terminal object $\mathbf{1}$ is given by

- $|\mathbf{1}|$ is a one-point cpo, and
- $\mathbf{1}(R)^w$ is the singleton subset consisting of the unique function in $[w \to |\mathbf{1}|]$.

Products are constructed by

- $|A \times B| \;=\; |A| \times |B|$, using the product in **CPO**, and
- $(A \times B)(R)^w \;=\; \{\langle f, g\rangle \mid f \in A(R)^w \text{ and } g \in B(R)^w\}$.

For exponents,

- $|B^A| \;=\; Hom_{\mathcal{SR}}(A, B)$, ordered pointwise,
- $B^A(R)^w \;=\; \{g \in [w \to |B^A|] \mid \forall \varphi : v \xrightarrow{\mathbf{C}} w.\, \forall h \in A(R)^v.\; (\underline{\lambda} i \in v.\, (g\ (\varphi\ (i)))\ (h(i))) \in B(R)^v\}$.

Note the interesting symbiotic relationship between the construction of the meanings of higher types and the relational meaning of higher types. The set $|B^A|$ is determined using the results of B and A on *all* sequentiality relations, whereas $B^A(R)^w$ picks out elements from $[w \to |B^A|]$ using the particular relation R. The definition of $B^A(R)$ therefore relies explicitly on the particular Kripke sequentiality relation R chosen as the basis, and implicitly on *all* sequentiality relations.

Lemma 7 *(a) $|B^A|$ is a cpo.*

(b) $B^A(R)$ satisfies completeness and Kripke monotonicity.

(c) B^A satisfies the concreteness condition.

Proof: We prove (c); (a) can be shown by a routine calculation, and (b) follows from the definition, using both the Kripke monotonicity and completeness properties of B and A. Suppose $f \in |B^A|$, that is, $f : A \to B$. We need to show that $(\underline{\lambda} i \in w.\, f) \in B^A(R)^w$. From the definition, we must show that, for $\varphi : v \xrightarrow{\mathrm{C}} w$ and $h \in A(R)^v$,

$$(\underline{\lambda} j \in v.\, ((\underline{\lambda} i \in w.\, f)\,(\varphi\,(j)))\,(h\,(j))) \in B(R)^v.$$

But this just reduces to $(\underline{\lambda} j \in v.\, f\,(h\,j)) \in B(R)^v$, which is the uniformity condition on $\mathcal{SR}$-morphisms f. ■

Lemma 8 *(a) $A \times B$ and B^A can be extended to bifunctors on $\mathcal{F}$, with B^A contravariant in A.*

(b) $(-) \times B$ is left adjoint to $B^{(-)}$.

Proof: *Proof of (a).* $f \times g$ is just the function induced by the underlying product in **CPO**. f^g has the usual definition: $f^g(h) = g; h; f$. Preservation of identities and compositions and various continuity conditions are straightforward to check, as is the uniformity condition for $f \times g$. We check the uniformity condition for f^g. First, the uniformity condition is preserved by composition, as is relevant domain-theoretic structure, so we may conclude that $g; h; f \in |B'^{A'}|$, where $f : B \to B'$ and $g : A' \to A$. To see that $f^g = g; -; f$ satisfies the uniformity condition, consider an R and $m \in B^A(R)^w$: we need that $(\underline{\lambda} i \in w.\, g; m(i); f) \in B'^{A'}(R)^w$, which in turn requires that, for $\varphi : v \xrightarrow{\mathrm{C}} w$ and $n \in A'(R)^v$,

$$(*) \qquad (\underline{\lambda} j \in v.\, (g; m(\varphi(j)); f)(n(j))) \in B'(R)^v.$$

By the uniformity condition for g, $(\underline{\lambda} j \in v.\, g(n(j))) \in A(R)^v$. Then, by the definition of $B^A(R)$ we obtain $(\underline{\lambda} j \in v.\, m(\varphi(j))(g(n(j)))) \in B(R)^v$, and a final application of uniformity for f gives the desired result $(*)$. Thus, we may conclude that f^g satisfies the uniformity condition.

Proof of (b). For $f : A \times B \to C$, $\mathrm{curry}(f) : A \to C^B$ is $\mathrm{curry}(f)a\,b = f\langle a, b\rangle$. For $g : A \to C^B$, $\mathrm{uncurry}(g) : A \times B \to C$ is $\mathrm{uncurry}(g)\langle a, b\rangle = (g\,a\,b)$. Of course, these are the same defining equations as in **CPO** (and many other categories). The point, however, is that the definition of C^B is just right to make these inverse isomorphisms. Clearly, $\mathrm{uncurry}(\mathrm{curry}(f)) = f$ and $\mathrm{curry}(\mathrm{uncurry}(g)) = g$, using the same argument as in **CPO**, as long as we can show that $\mathrm{uncurry}(g)$ and $\mathrm{curry}(f)$ are actually defined. For this we need only verify the appropriate parametricity conditions, as continuity and naturality properties are straightforward. We treat $\mathrm{curry}(f)$, leaving the similar case of $\mathrm{uncurry}(g)$ to the reader.

We need to show that curry(f) is a well-defined function from $|A|$ to $|C^B|$ and that it satisfies the uniformity condition. First, for well-definedness, we must show that for any $a \in |A|$, R, and w, if $h \in B(R)^w$ then $(\underline{\lambda} i \in w.\, \mathrm{curry}(f)\, a\, (h(i))) \in C(R)^w$. By the concreteness condition, $a \in |A|$ implies that $(\underline{\lambda} i \in w.\, a) \in A(R)^w$, which means that $(\underline{\lambda} i \in w.\, \langle a, h(i)\rangle) \in (A \times B)(R)^w$. Uniformity for f then gives $(\underline{\lambda} i \in w.\, f\langle a, h(i)\rangle) \in C(R)^w$, which by definition of curry(f) is what we wanted to show. Second, for uniformity of curry(f), suppose $k \in A(R)^w$; we need to show that $(\underline{\lambda} i \in w.\, \mathrm{curry}(f)(k(i))) \in C^B(R)^w$, which, from the definition of $C^B(R)$, requires proving

$$(**) \quad (\underline{\lambda} j \in v.\, f\langle k'(j), h(j)\rangle) \in C(R)^v$$

for $\varphi : v \xrightarrow{\mathrm{C}} w$ and $h \in B(R)^v$, where $k' = (\varphi; k)$. Since A satisfies Kripke monotonicity, we know that $k' \in A(R)^v$, and so the desired property $(**)$ is immediate from the uniformity condition for f. ■

Proposition 9 *$\mathcal{SR}$ is a cpo-enriched cartesian closed category, with $Hom_{\mathcal{SR}}(A, B)$ ordered pointwise. It is order-extensional in the following sense:*

$$f \sqsubseteq g : A \to B \quad \Leftrightarrow \quad \forall e : \mathbf{1} \to A\,.\, e; f \sqsubseteq e; g$$

Proof: That $\times, \mathbf{1}$ is a cartesian product structure should be clear; the projections and pairing are just as in **CPO**. The previous lemma shows cartesian closure, and the preservation of relevant cpo-enriched structure is straightforward. The concreteness condition implies that $\mathcal{SR}$ is a concrete (well-pointed) category, which is to say that two maps f, g are equal iff $(e; f) = (e; g)$ for all maps e out of $\mathbf{1}$. Order-extensionality is then immediate from the pointwise ordering of hom sets. ■

The least fixpoint map $Y : A^A \to A$ is standard: $(Y\, f)$ is the least fixed-point of the function $f : |A| \to |A|$, defined $(Y\ f) = \bigsqcup_{n \geq 0} \{f^n(\bot_{|A|})\}$. The operator Y satisfies the uniformity condition by the completeness property of Kripke relations.

3.4 Interpretation of Types and Terms

We now give a concrete description of the programming language PCF and its model in this category. The version of PCF used here has one base type **nat** of natural numbers for simplicity. The types are given by the grammar

$$s, t ::= \mathbf{nat} \mid (s \to t).$$

A typing judgement is a formula of the form $\Gamma \vdash M : t$ where M is a term, t a type, and Γ is a **PCF typing context**, *i.e.*, a finite function from variables to types. Standard rules for deriving typing judgements are as follows.

$$\frac{}{\Gamma, x : t \vdash x : t} \qquad \frac{}{\Gamma \vdash 0 : \mathbf{nat}}$$

$$\frac{\Gamma, x : t \vdash M : s}{\Gamma \vdash (\lambda x : t.M) : t \to s} \qquad \frac{\Gamma \vdash M : t \to s \quad \Gamma \vdash N : t}{\Gamma \vdash (M\ N) : s} \qquad \frac{\Gamma \vdash M : t \to t}{\Gamma \vdash (\mathbf{Y}_t\, M) : t}$$

$$\frac{\Gamma \vdash M : \mathbf{nat}}{\Gamma \vdash (\mathbf{succ}\ M) : \mathbf{nat}} \qquad \frac{\Gamma \vdash M : \mathbf{nat}}{\Gamma \vdash (\mathbf{pred}\ M) : \mathbf{nat}} \qquad \frac{\Gamma \vdash M_i : \mathbf{nat}}{\Gamma \vdash (\mathbf{ifz}\ M_1\ \mathbf{then}\ M_2\ \mathbf{else}\ M_3) : \mathbf{nat}}$$

The interpretation of PCF types is straightforward in $\mathcal{SR}$. The base type **nat** is interpreted as an $\mathcal{SR}$-object $[\![\mathbf{nat}]\!]$ by

$$\begin{aligned} |[\![\mathbf{nat}]\!]| &= \mathbf{N}_\perp \\ [\![\mathbf{nat}]\!]\, R &= R. \end{aligned}$$

For function types we use the exponent in $\mathcal{SR}$: $[\![s \to t]\!] = [\![t]\!]^{[\![s]\!]}$.

The interpretation of PCF terms is also relatively straightforward. The maps pred, succ, and ifz defined earlier are morphisms in $\mathcal{SR}$, *i.e.*, $\mathsf{pred}, \mathsf{succ} \in Hom_{\mathcal{SR}}([\![\mathbf{nat}]\!], [\![\mathbf{nat}]\!])$ and $\mathsf{ifz} \in Hom_{\mathcal{SR}}([\![\mathbf{nat}]\!] \times [\![\mathbf{nat}]\!] \times [\![\mathbf{nat}]\!], [\![\mathbf{nat}]\!])$. This, together with Proposition 9, is enough to determine a model of PCF, but we give a concrete description of the semantics of terms to settle notation for the proofs that follow. If $\Gamma = x_1 : t_1, ..., x_n : t_n$ is a typing context then $[\![\Gamma]\!] = [\![t_1]\!] \times ... \times [\![t_n]\!]$. (The order is not important here, as we could rely on some fixed ordering of $x_i : t_i$ pairs.) In the case that Γ is empty $[\![\Gamma]\!]$ is the terminal object $\mathbf{1}$. For an environment $\rho \in |[\![\Gamma]\!]|$, we often write $\rho(x)$ for projection to the component corresponding to variable x. The meaning of a judgement $\Gamma \vdash M : t$ is an $\mathcal{SR}$-morphism $[\![\Gamma \vdash M : t]\!] : [\![\Gamma]\!] \longrightarrow [\![t]\!]$ satisfying the equations

$$\begin{aligned} [\![\Gamma, x : t \vdash x : t]\!]\rho &= \rho(x) \\ [\![\Gamma \vdash (M\ N) : t]\!]\rho &= ([\![\Gamma \vdash M : (s \to t)]\!]\rho)\,([\![\Gamma \vdash N : s]\!]\rho) \\ [\![\Gamma \vdash (\lambda x : s.\, M) : (s \to t)]\!]\rho &= f, \quad \text{where } f(d) = [\![\Gamma, x : s \vdash M : t]\!]\rho[x \mapsto d] \\ [\![\Gamma \vdash (\mathbf{Y}\ M) : s]\!]\rho &= (Y\ [\![\Gamma \vdash M : (s \to s)]\!]\rho) \\ [\![\Gamma \vdash 0 : \mathbf{nat}]\!]\rho &= 0 \\ [\![\Gamma \vdash (\mathbf{succ}\ M) : \mathbf{nat}]\!]\rho &= \mathsf{succ}([\![\Gamma \vdash M : \mathbf{nat}]\!]\rho) \\ [\![\Gamma \vdash (\mathbf{pred}\ M) : \mathbf{nat}]\!]\rho &= \mathsf{pred}([\![\Gamma \vdash M : \mathbf{nat}]\!]\rho) \\ [\![\Gamma \vdash (\mathbf{ifz}\ M\ \mathbf{then}\ N\ \mathbf{else}\ P) : \mathbf{nat}]\!]\rho &= \mathsf{ifz}([\![\Gamma \vdash M : \mathbf{nat}]\!]\rho, [\![\Gamma \vdash N : \mathbf{nat}]\!]\rho, [\![\Gamma \vdash P : \mathbf{nat}]\!]\rho) \end{aligned}$$

where $\rho[x \mapsto d]$ denotes the environment in which the x component is extended (or overwritten) to d. If $\emptyset \vdash M : s$, we write $[\![M]\!]$ for the corresponding element $[\![\emptyset \vdash M : s]\!]\emptyset \in [\![s]\!]$.

4 Kripke Invariance and PCF Definability

In this section we show that every element in the model is a least upper bound of a directed set of definable elements. The proof is based on ideas from (Jung and Tiuryn, 1993), and proceeds by considering specific finitary Kripke sequentiality relations over specific categories.

For the proof to work with finitary Kripke relations we use the fact that the general form of the construction—and the fact that we are dealing with PCF—forces each cpo to be an SFP object. Define

$$\begin{aligned} P^n_{\mathbf{nat}} &= \lambda x : \mathbf{nat}.\, \mathbf{ifz}\ x\ \mathbf{then}\ x\ \mathbf{else}\ (\ldots \mathbf{ifz}\ (\mathbf{pred}^n\ x)\ \mathbf{then}\ x\ \mathbf{else}\ \Omega \ldots) \\ P^n_{s \to t} &= \lambda x : (s \to t).\, \lambda y : s.\, P^n_t\ (x\ (P^n_s\ y)) \end{aligned}$$

and let $\psi^n_s = [\![P^n_s]\!]$. Since we have a model of PCF built from continuous functions, $\psi^n_s : [\![s]\!] \to [\![s]\!]$ is continuous. One may use this fact to prove

Lemma 10 *(Milner, 1977) For any s and $d \in |[\![s]\!]|$, $d = \bigsqcup_{n<\omega} \psi^n_s(d)$. Furthermore, each ψ^n_s is idempotent.*

For each natural number n we define a category $\mathbf{C}_n$. First, for each PCF type s, let D^n_s be the set $\{d \in |[\![s]\!]| \mid d = \psi^n_s(d)\}$. Then the objects of $\mathbf{C}_n$ are products

$$[D^n_{s_1}, \ldots, D^n_{s_m}] = (D^n_{s_1} \times \ldots \times D^n_{s_m})$$

It is understood that as an object of $\mathbf{C}_n$ the domain-theoretic structure is forgotten, so that these are considered simply as sets. The morphisms are projections from $[D^n_{s_1}, \ldots, D^n_{s_{m+k}}]$ to $[D^n_{s_1}, \ldots, D^n_{s_m}]$.

Next we define a $\mathbf{C}_n$-Kripke relation R_n. First, for any function $f \in [E_1 \to \ldots \to E_m \to E]$, let $uncurry_m(f) : [(E_1 \times \ldots \times E_m) \to E]$ be defined by $uncurry_m(f)(e_1, \ldots, e_m) = (f\ e_1 \ldots e_m)$. Then for any $w = [D^n_{s_1}, \ldots, D^n_{s_m}]$,

$$R^w_n = \{g \in [w \to \mathbf{N}_\perp] \mid \text{there is a closed } M \text{ such that } g = (e_w; uncurry_m([\![M]\!]))\},$$

where the function $e_w : w \to (|[\![s_1]\!]| \times \ldots \times |[\![s_m]\!]|)$ is the inclusion and the type of the closed terms M is $(s_1 \to \ldots \to s_m \to \mathbf{nat})$. Notice that, although the domain of $g \in [[t]]R^w_n$ is restricted, the range of g need not lie in D^t_n. This is in fact necesary if R_n is to be a sequentiality relation. The construction of this sequentiality relation is interesting because of the apparent "circularity": the particular sequentiality relation is defined on a category $\mathbf{C}_n$, and $\mathbf{C}_n$ is constructed using *all* sequentiality relations. Of course, this is not a real foundational issue, but the technique does resemble the proof of strong normalization of the Girard-Reynolds polymorphic λ-calculus using Girard's "reducibility candidates", *cf.* (Gallier, 1989; Girard et al., 1989).

Lemma 11 *R_n is a finitary Kripke sequentiality relation.*

Proof: Directed completeness follows from the fact that each w is finite and $\mathbf{N}_\perp$ is flat. To see Kripke monotonicity, suppose $w = [s_1, \ldots, s_m]$, $v = [s_1, \ldots, s_{m+k}]$, $\varphi : v \xrightarrow{\mathbf{C}} w$, and $g \in R^w_n$; we want to show that $(\varphi; g) \in R^v_n$. Let $s = (s_1 \to \ldots \to s_m \to \mathbf{nat})$ and $s' = (s_1 \to \ldots \to s_{m+k} \to \mathbf{nat})$. By definition, $g = (e_w; uncurry_m([\![\emptyset \vdash M_j : s]\!]))$ for some M_j. Then note that

$$(\varphi; g) = (e_v; uncurry_{m+k}([\![\emptyset \vdash (\lambda x : s_1.\ \ldots \lambda x_{m+k} : s_{m+k}.\ M_j\ x_1 \ldots x_m) : s']\!]))$$

and hence $(\varphi; g) \in R^v_n$ as desired. Thus, R_n is a Kripke relation.

To show that it is a sequentiality relation it suffices, by Proposition 6, to show that each of the base constants is invariant. We prove one of the closure conditions and leave the others to the reader. Suppose $g \in R^w_n$ where $w = [s_1, \ldots, s_m]$. Then $g = (e_w; uncurry_m([\![\emptyset \vdash M_j : s]\!]))$ where $s = (s_1 \to \ldots \to s_m \to \mathbf{nat})$. But

$$(\underline{\lambda} i \in w.\, \mathsf{succ}(g\ i)) = (e_w; (uncurry_m([\![\emptyset \vdash (\lambda x_1 : s_1.\ \ldots \lambda x_m : s_m.\, \mathbf{succ}\ (M_j\ x_1 \ldots x_m)) : s]\!])))$$

and hence $(\underline{\lambda} i \in w.\, \mathsf{succ}(g\ i)) \in R^w_n$ as required. ■

The proof of the following is an adaptation of the proof of the characterization of λ-definability in (Jung and Tiuryn, 1993).

Lemma 12 *If $f = \psi^n_s(f)$, then there is a closed expression M such that $f = [\![M]\!]$.*

Proof: We prove the following claim by induction on the type t:

Suppose $w = [D^n_{s_1}, \ldots, D^n_{s_m}]$ and $g \in [w \to |[\![t]\!]|]$ is such that $g = (g; \psi^n_t)$. Then $g \in [\![t]\!](R_n)^w$ iff there is a closed expression M such that $g = (e_w; uncurry_m([\![M]\!]))$.

Choosing $t = s$ and $w = [\,]$, we know

- By the concreteness condition and Lemma 11, $(\underline{\lambda} d \in [\,]. f) \in [\![t]\!](R)^w$, and
- $(\underline{\lambda} d \in [\,]. f) = (\underline{\lambda} d \in [\,]. \psi^n_s(f)) = ((\underline{\lambda} d \in [\,]. f); \psi^n_s)$,

and thus the claim will establish the lemma.

The basis when $t = \mathbf{nat}$ holds by the definition of R^w_n, so consider the induction case where $t = (t_0 \to t_1)$. To prove $(\Rightarrow)$, suppose $g = (g; \psi^n_t)$ and $g \in [\![t]\!](R_n)^w$, where $w = [D^n_{s_1}, \ldots, D^n_{s_m}]$. Let $v = [D^n_{s_1}, \ldots, D^n_{s_m}, D^n_{t_0}]$, $\varphi : v \to w$ be the projection, and

$$h = (e_v; uncurry_{m+1}([\![\lambda x_1 : s_1. \ldots \lambda x_m : s_m. \lambda x : t_0. x]\!])).$$

Notice that $h = (h; \psi^n_{t_0})$ since, for $d \in D^n_{t_0}$, $\psi^n_{t_0}(d) = d$. Thus, by induction $h \in [\![t_0]\!](R_n)^v$. Then since $g \in [\![t]\!](R)^w$, it follows that

$$g' = (\underline{\lambda} x \in v. (g\ (\varphi\ x))\ (h\ x)) \in [\![t_1]\!](R_n)^v.$$

Moreover, since $g = (g; \psi^n_t)$, it is easy to see that $g' = (g'; \psi^n_{t_1})$. Hence, by induction, there is an M such that $g' = (e_v; uncurry_{m+1}([\![M]\!]))$. Let

$$Q = \lambda x_1 : s_1. \ldots \lambda x_m : s_m. \lambda x : t_0. (M\ x_1 \ldots x_m\ (P^n_{t_0}\ x)).$$

Then for any $\langle d_1, \ldots, d_m \rangle \in w$ and $d \in |[\![t_0]\!]|$,

$$\begin{aligned}(e_w; uncurry_m([\![Q]\!]))\ \langle d_1, \ldots, d_m \rangle\ d &= g'\ \langle d_1, \ldots, d_m, (\psi^n_{t_0}\ d)\rangle \\ &= g\ \langle d_1, \ldots, d_m \rangle\ (\psi^n_{t_0}\ d) \\ &= g\ \langle d_1, \ldots, d_m \rangle\ d\end{aligned}$$

since $g = (g; \psi^n_t)$, so $g = (e_w; uncurry_m([\![Q]\!]))$ as desired.

For the $(\Leftarrow)$ direction, suppose $g = (g; \psi^n_t)$ and there is an M such that $g = (e_w; uncurry_m([\![M]\!]))$; we want to show $g \in [\![t]\!](R_n)^w$. So suppose $v = [D^n_{s_1}, \ldots, D^n_{s_{m+k}}]$, $\varphi : v \to w$ is the projection, and $h \in [\![t_0]\!](R_n)^v$. It follows that $(h; \psi^n_{t_0}) \in [\![t_0]\!](R_n)^v$ since, by the uniformity condition R_n is invariant under $\psi^n_{t_0}$. Then, since $(h; \psi^n_{t_0}) = (h; \psi^n_{t_0}); \psi^n_{t_0}$, by induction there is a closed term N such that $(h; \psi^n_{t_0}) = (e_v; uncurry_{m+k}([\![N]\!]))$. Let

$$P = \lambda x_1 : s_1. \ldots \lambda x_{m+k} : s_{m+k}. (M\ x_1 \ldots x_m)\ (N\ x_1 \ldots x_{m+k}).$$

Therefore, for any $\langle d_1, \ldots, d_{m+k} \rangle \in v$,

$$\begin{aligned}&(e_v; uncurry_{m+k}([\![P]\!]))\ \langle d_1, \ldots, d_{m+k} \rangle \\ &\quad = (uncurry_m([\![M]\!])\ (e_w\ \langle d_1, \ldots, d_m \rangle))\ (uncurry_{m+k}([\![N]\!])\ (e_v\ \langle d_1, \ldots, d_{m+k} \rangle)) \\ &\quad = (g\ (\varphi\ \langle d_1, \ldots, d_{m+k} \rangle))\ ((h; \psi^n_{t_0})\ \langle d_1, \ldots, d_{m+k} \rangle) \\ &\quad = (g\ (\varphi\ \langle d_1, \ldots, d_{m+k} \rangle))\ (\psi^n_{t_0}\ (h\ \langle d_1, \ldots, d_{m+k} \rangle)) \\ &\quad = (g\ (\varphi\ \langle d_1, \ldots, d_{m+k} \rangle))\ (h\ \langle d_1, \ldots, d_{m+k} \rangle)\end{aligned}$$

where the last line follows from the fact that $g = (g; \psi^n_t)$. By induction, $(\underline{\lambda} x \in v. (g\ (\varphi\ x))\ (h\ x)) \in [\![t_1]\!](R_n)^v$, and hence $g \in [\![t]\!](R_n)^w$. ■

We now have enough facts to establish the basic connections between the model and the language. We first define the observational approximation relation $\preceq$ as follows.

Definition 13 1. $C[\cdot]$ is a Γt-context if $\emptyset \vdash C[M] : \mathbf{nat}$, whenever $\Gamma \vdash M : t$.

2. $M \preceq_{\Gamma t} N$ if $\Gamma \vdash M : t$, $\Gamma \vdash N : t$, and for all Γt-contexts $C[\cdot]$, $[\![\Gamma \vdash C[M] : \mathbf{nat}]\!] \sqsubseteq [\![\Gamma \vdash C[N] : \mathbf{nat}]\!]$.

Here we have used the denotational semantics to determine "observable approximation." The adequacy of this model for the usual operational semantics can be shown using the standard computability method (*cf.* (Plotkin, 1977)). The proof of full abstraction follows from Lemma 10, Lemma 12, and continuity.

Theorem 14 (Full Abstraction) $M \preceq_{\Gamma t} N$ *iff* $[\![\Gamma \vdash M : t]\!] \sqsubseteq [\![\Gamma \vdash N : t]\!]$.

5 Conclusion

In this paper we have given a characterization of the (unique by (Milner, 1977)) inequationally fully abstract model of PCF. The results of this paper owe much to (Jung and Tiuryn, 1993) and (Sieber, 1992), and we make no claim of great originality. It is clearly interesting, however, that such a full abstraction result is possible using logical relations. We were led to the connection between the two works by our own work on translating PCF into a language with parametric polymorphism (O'Hearn and Riecke, 1994) (hence the connection to λ-definability and Jung and Tiuryn's work). We view our results as strengthening, and providing further justification for, the research program begun in (Sieber, 1992).

One crucial question remains: is the model based on Kripke relations actually different than Sieber's finitary relation model? All equivalences that we know of that are treated incorrectly by the continuous model *are* in fact treated correctly by the model based on fixed-arity finitary relations. This situation is rather like (Plotkin, 1980), where binary relations characterize λ-definability in the full type hierarchy over an infinite ground set up to type-level two, and Kripke relations characterize definability at all types, but there remains the nagging question of whether binary or finitary relations already suffice for definability (the example of (Statman, 1985) is not for the full type hierarchy). We know here that Kripke relations suffice for technical purposes but not whether they are necessary, *i.e.*, whether the simpler fixed-arity relations of Sieber suffice for full abstraction.

One may wonder, with all the previous constructions of models of PCF, whether this construction constitutes a solution to the "full abstraction problem". It has been remarked on a number of occasions (Abramsky et al., 1994; Berry et al., 1985; Jung and Stoughton, 1993) that there is no universal agreement on the requirements for a "solution." At the very least, one would like a construction that does not depend on the syntax or operational semantics of PCF. Although the syntax of PCF was used in the *proof* of full abstraction, the semantic category in which the model lives was defined without recourse to the type structure of PCF or to operational semantics, and so we feel that the construction satisfies this first criterion. A second criterion, argued in (Abramsky et al., 1994), is that the construction should exist in a cartesian closed category, so that in particular the function type is explained using an exponential construction. Our presentation also meets this criterion. In fact, it would also have been possible to use Kripke relations to characterize those

elements in the continuous function model that are lubs of definables, and then use the techniques of (Jung and Stoughton, 1993) to collapse to the fully abstract model. It is not clear at present whether our method could yield more useful information about PCF than this collapsing, though we agree that it is desirable to present the model in terms of a cartesian closed category.

Jung and Stoughton (1993) propose a third criterion: a solution should yield an effective presentation of finitary PCF, *i.e.*, PCF with just the boolean type. By "effective presentation" is meant, roughly, a procedure that prints out, for each type, a table of graphs of PCF-definable functions, *and* which indicates when the table for a type is complete. In other words, such a solution would guarantee that given the *graph* of a function in finitary PCF, one could tell whether it was in the model or not. Our model is *not* a solution in this sense, due to the complexity of the logical relations; if, for instance, Sieber's more tame relations determined the fully abstract model, there *would* be an effective presentation. Of course, there may be no solution meeting this third criterion. (The undecidability result of (Loader, 1994), for λ-definability in the full type hierarchy over a finite base type, is interesting but apparently not immediately relevant to the PCF definability problem.)

The results of this paper were obtained subsequent to the full abstraction results reported by Abramsky, Jagadeesan and Malacaria (Abramsky et al., 1994; Abramsky et al., 1993) and Hyland and Ong (Hyland and Ong, 1993) using games semantics. The games semantics approaches the full abstraction problem for PCF by first providing an *intensional* model, which is then quotiented to achieve extensionality. In contrast, here and in (Sieber, 1992) the starting point is manifestly extensional, and logical relations are used to impose stringent conditions on function types. The games semantics does a better job of explaining the "temporal" or "process" aspect of sequentiality, and in particular the structure in the intensional games semantics is already interesting and informative, prior to quotienting, and independent of questions of full abstraction.

Our construction probably does not offer a *definitive* account of sequential functional computation, even though Sieber's sequentiality relations, along with our variation on them, clearly exhibit some semantic aspects of sequentiality. For instance, the fully abstract models for sequential PCF and parallel PCF (with "parallel or") coexist in our category $\mathcal{SR}$. We have seen that $\mathcal{SR}$ contains the fully abstract model of PCF, but it *also* contains the continuous function model (Plotkin, 1977; Scott, 1993): for this, we would simply define each $[\![\mathbf{nat}]\!]R$ to be the evident everywhere-true Kripke relation on $\mathbf{N}_\perp$, that is, where $([\![\mathbf{nat}]\!]R)^w = [w \to \mathbf{N}_\perp]$. Nevertheless, we feel that the logical relation approach still has clear interest when it comes to principles for reasoning about sequential functions. As was remarked above, logical-relation reasoning handles many examples quite smoothly, and allows for an effective presentation of finitary PCF up to type-level two. This is illustrated well by Stoughton's implementation of an algorithm for definability problems (Stoughton, 1994), and its use on the subtle examples of (Curien, 1986).

Another closely related work is that of Cartwright, Curien, and Felleisen, where a fully abstract model is presented for SPCF, a "sequential" extension of PCF that includes errors and a version of the "catch" construct (Cartwright et al., 1994). While this result is not for PCF itself, the model, which turns out to be a version of sequential algorithms (Berry and Curien, 1982), is quite satisfactory. In particular, the preservation conditions for "manifestly sequential functions" are of sufficient quality to yield an effective presentation of a finitary version of SPCF (Felleisen, personal communication, February 1994). The possibility of finding something similar for PCF is one reason why further developments along the lines of, *e.g.*, (Bucciarelli and Erhardt, 1991; Brookes and Geva, 1994) continue to hold interest.

Acknowledgements: We thank Gerard Berry, Achim Jung, Albert Meyer, Gordon Plotkin, Kurt Sieber and Allen Stoughton for helpful discussions and comments, and especially Matthias Felleisen, Martin Odersky, and Ramesh Subrahmanyam for their encouragement.

References

Abramsky, S., Jagadeesan, R., and Malacaria, P. (1993). Games and full abstraction for PCF: preliminary announcement. Communicated to the TYPES electronic mail list, July 23, 1993.

Abramsky, S., Jagadeesan, R., and Malacaria, P. (1994). Full abstraction for PCF (extended abstract). In Hagiya, M. and Mitchell, J. C., editors, *Theoretical Aspects of Computer Software*, volume 789 of *Lecture Notes in Computer Science*, pages 1–15. Springer-Verlag.

Berry, G. (1978). Stable models of typed lambda calculi. In Ausiello, G. and Böhm, C., editors, *Automata, Languages and Programming*, volume 62 of *Lecture Notes in Computer Science*, pages 72–89, Berlin. Springer-Verlag.

Berry, G. and Curien, P.-L. (1982). Sequential algorithms on concrete data structures. *Theoretical Computer Science*, 20:265–321.

Berry, G., Curien, P.-L., and Lévy, J.-J. (1985). Full abstraction for sequential languages: the state of the art. In Nivat, M. and Reynolds, J. C., editors, *Algebraic Methods in Semantics*, pages 89–132. Cambridge University Press, Cambridge, England.

Brookes, S. and Geva, S. (1994). Sequential functions on indexed domains and full abstraction for a sublanguage of PCF. In Brookes, S. et al., editors, *Proceedings of the 8th Annual Symposium on Mathematical Foundations of Program Semantics*, volume 802 of *Lecture Notes in Computer Science*, pages 320–332, New Orleans. Springer-Verlag.

Bucciarelli, A. and Erhardt, T. (1991). Sequentiality and strong stability. In *Proceedings, 6th Annual IEEE Symposium on Logic in Computer Science*, Santa Cruz, California. IEEE Computer Society Press, Los Alamitos, California.

Cartwright, R., Curien, P.-L., and Felleisen, M. (1994). Fully abstract semantics for observably sequential languages. *Information and Computation*, 111:297–401.

Curien, P.-L. (1986). *Categorical Combinators, Sequential Algorithms and Functional Programming.* Wiley.

Gallier, J. H. (1989). On Girard's "candidats de reductibilité". In Odifreddi, P., editor, *Logic and Computer Science.* Academic Press.

Girard, J.-Y., Lafont, Y., and Taylor, P. (1989). *Proofs and Types.* Cambridge University Press.

Hyland, J. and Ong, L. (1993). Dialogue games and innocent strategies: An approach to (intensional) full abstraction for PCF (preliminary announcement). Communicated to the TYPES electronic mail list, July 26, 1993.

Jung, A. and Stoughton, A. (1993). Studying the fully abstract model of PCF within its continuous function model. In *Typed Lambda Calculi and Applications*, volume 664 of *Lecture Notes in Computer Science*, pages 230–244. Springer-Verlag.

Jung, A. and Tiuryn, J. (1993). A new characterization of lambda definability. In *Typed Lambda Calculi and Applications*, volume 664 of *Lecture Notes in Computer Science*, pages 245–257. Springer-Verlag.

Loader, R. (1994). The undecidability of λ-definability. In Zeleny, M., editor, *The Church Festschrift*. CSLI/University of Chicago Press.

Ma, Q. and Reynolds, J. C. (1992). Types, abstraction, and parametric polymorphism, part 2. In Brookes, S. et al., editors, *Mathematical Foundations of Programming Semantics*, volume 598 of *Lecture Notes in Computer Science*, pages 1–40. Springer-Verlag, Berlin. Proceedings of the 1991 Conference.

Meyer, A. R. and Cosmadakis, S. S. (1988). Semantical paradigms: notes for an invited lecture. In *Proceedings, 3rd Annual Symposium on Logic in Computer Science*, pages 236–53, Edinburgh, Scotland. IEEE Computer Society Press.

Milner, R. (1977). Fully abstract models of typed λ-calculi. *Theoretical Computer Science*, 4:1–22.

Mitchell, J. C. (1990). Type systems for programming languages. In van Leeuwen, J., editor, *Handbook of Theoretical Computer Science*, volume B, chapter 8, pages 365–458. Elsevier, Amsterdam, and The MIT Press, Cambridge, Mass.

Mitchell, J. C. and Scedrov, A. (1993). Notes on sconing and relators. In Boerger, E. et al., editors, *Computer Science Logic '92, Selected Papers*, volume 702 of *Lecture Notes in Computer Science*, pages 352–378. Springer-Verlag.

Odifreddi, P. (1989). *Classical Recursion Theory*, volume 125 of *Studies in Logic and Foundations of Mathematics*. North Holland.

O'Hearn, P. and Riecke, J. (1994). Fully abstract translations and parametric polymorphism. In Sannella, D., editor, *Programming Languages and Systems—ESOP '94*, volume 788 of *Lecture Notes in Computer Science*, pages 454–468. Springer-Verlag.

O'Hearn, P. W. and Tennent, R. D. (1993). Parametricity and local variables. Technical Report SU-CIS-93-30, Syracuse University. Preliminary version appeared in *Conf. Record 20th ACM Symp. on Principles of Programming Languages*, Charleston, South Carolina, pages 171–184. ACM, New York, 1993.

Plotkin, G. D. (1973). Lambda-definability and logical relations. Memorandum SAI-RM-4, School of Artificial Intelligence, University of Edinburgh.

Plotkin, G. D. (1977). LCF considered as a programming language. *Theoretical Computer Science*, 5:223–255.

Plotkin, G. D. (1980). Lambda-definability in the full type hierarchy. In Seldin, J. P. and Hindley, J. R., editors, *To H. B. Curry: Essays in Combinatory Logic, Lambda Calculus and Formalism*, pages 363–373. Academic Press.

Reynolds, J. C. (1983). Types, abstraction and parametric polymorphism. In Mason, R. E. A., editor, *Information Processing 83*, pages 513–523. North Holland, Amsterdam.

Scott, D. (1993). A type theoretical alternative to CUCH, ISWIM, OWHY. *Theoretical Computer Science*, 121:411–440. Published version of unpublished manuscript, Oxford University, 1969.

Sieber, K. (1992). Reasoning about sequential functions via logical relations. In Fourman, M. P., Johnstone, P. T., and Pitts, A. M., editors, *Applications of Categories in Computer Science*, volume 177 of *London Mathematical Society Lecture Note Series*, pages 258–269. Cambridge University Press, Cambridge, England.

Sieber, K. (1993). New steps towards full abstraction for local variables. In *ACM SIGLPLAN Workshop on State in Programming Languages*, pages 88–100. Available as Yale Technical Report YALEU/DCS/RR-968.

Statman, R. (1985). Logical relations and the typed λ-calculus. *Information and Computation*, 65:85–97.

Stoughton, A. (1988). *Fully Abstract Models of Programming Languages*. Research Notes in Theoretical Computer Science. Pitman, London, and Wiley, New York.

Stoughton, A. (1994). Mechanizing logical relations. In Brookes, S. et al., editors, *Proceedings of the 8th Annual Symposium on Mathematical Foundations of Program Semantics*, volume 802 of *Lecture Notes in Computer Science*, pages 359–377, New Orleans. Springer-Verlag.