Modelling local variables: possible worlds and object spaces

Guy McCusker[1]

Department of Computer Science
University of Bath
Bath BA2 7AY, United Kingdom

John Power[2]

Department of Computer Science
University of Bath
Bath BA2 7AY, United Kingdom

Abstract

Local variables in imperative languages have been given denotational semantics in at least two fundamentally different ways. One is by use of functor categories, focusing on the idea of *possible worlds*. The other might be termed *event-based*, exemplified by Reddy's object spaces and models based on game semantics. O'Hearn and Reddy have related the two approaches by giving functor category models whose worlds are object spaces, then showing that their model is fully abstract for Idealised Algol programs up to order two. But the category of object spaces is not small, and so in order to construct a functor category that is locally small, and hence Cartesian closed, they need to work with a restricted collection of object spaces. This weakens the connection between the object spaces model and the functor-category model: the Yoneda embedding no longer provides a full embedding of the original category of object spaces into the functor-category. Moreoever the choice of the restricted collection of object spaces is ad hoc. In this paper, we refine the approach by proving that the finite objects form a small dense subcategory of a simplified object-spaces model. The functor category over these finite objects is therefore locally small and Cartesian closed, and contains the object-spaces category as a full subcategory. All this work is necessarily enriched in **Cpo**. We further refine their full abstraction result by showing that full abstraction fails at order three.

Keywords: Denotational semantics, programming languages, imperative programming, algol-like languages, local state, possible worlds, category theory.

[1] Email: G.A.McCusker@bath.ac.uk
[2] Email: A.J.Power@bath.ac.uk

1 Introduction

Over recent decades, there have been at least two major ways in which local variables in imperative languages have been given denotational semantics. One approach, developed in particular by O'Hearn and Tennent, building heavily upon the work of others such as Oles, has been based on functor categories: one constructs a small category $\mathbf{W}$ of worlds and models the language in the functor category $\mathbf{Set}^{\mathbf{W}}$ or perhaps in $\mathbf{Cpo}^{\mathbf{W}}$ to include recursion. The various small categories $\mathbf{W}$ of worlds have gradually become of increasing sophistication in order to model increasingly sophisticated features of local variables, as outlined in the collection of articles [13].

The other approach we have in mind here has been that most associated with Reddy [14]. He developed object spaces to model interference-controlled Algol, recording the behaviour of stateful objects using traces. Reddy's model has been an important precursor of the game semantics of imperative programs [1,2], which provided the first fully abstract models of Idealised Algol. These models were not given by functor categories but rather directly in terms of natural but sophisticated structures on sets.

This leaves the question whether the two approaches can be unified. O'Hearn and Reddy made a start on that question in [12], providing a fully faithful embedding of a variant of Reddy's category into a functor category, and describing a model of Idealised Algol in that category. Although a substantial step forward, that still leaves a substantial step, indeed from a category-theoretic perspective, a fundamental step, unresolved. Specifically the problem is one of size: Reddy's category is locally small but not small. If, as O'Hearn and Reddy do, one makes an ad hoc restriction to a small subcategory, one loses fundamental category-theoretic structure such as completeness and cocompleteness properties. But if one starts with a category $\mathbf{W}$ that is not small, then the functor category $\mathbf{Set}^{\mathbf{W}}$ typically will not be locally small, again causing major disruption, for instance not allowing one to conclude cartesian closedness.

We resolve that difficulty in this paper. Specifically, we address it by the study of density: a small full subcategory $\mathbf{D}$ of a category $\mathbf{C}$ is dense if a canonical functor embeds $\mathbf{C}$ fully into the functor category $\mathbf{Set}^{\mathbf{D}}$, which is necessarily locally small. Density is a fundamental notion within category theory [7] and its use brings us to a very satisfactory situation: we have a functor-category model of Idealised Algol, an object-spaces model of interference-controlled Algol with no unnatural size restriction, and a full embedding between them. There is one complication with this: we provably need not just density but an enriched version of density, with enrichment in the category $\mathbf{Cpo}$. We provide a counter-example to show that ordinary density is insufficient in Section 5, then show that we have the $\mathbf{Cpo}$-enriched density we seek. The enriched notion of density is a standard part of enriched category theory [6].

Our work is not a direct refinement of O'Hearn and Reddy's, because we do not work with the original formulation of the category of object spaces, but with a simplification of it, along the lines developed in [8,10], which abandons the coherence-space structure exploited by Reddy. The resulting category of worlds has the advantage of being simpler to describe than Reddy's category: it is the category of free quantales over finite sets. It nevertheless gives rise to the same model of interference-controlled Algol as Reddy's category, and the functor-category possesses the same full abstraction properties as claimed in O'Hearn and Reddy's work.

As an additional contribution, we refine O'Hearn and Reddy's full abstraction result. In their paper, it was shown that the functor-category model was fully abstract up to order two, that is, that for terms of order two and below, equality in the model coincides with contextual equivalence in an appropriate operational semantics. Their argument for full abstraction carries almost directly over to our model. We give a counterexample to full abstraction at order three, showing that this result is as far as one can go.

2 Two Algol-like Languages

This section briefly presents the syntax of a version of Idealised Algol, and its interference-controlled variant. We omit details of the operational semantics, which may be found in the literature; O'Hearn and Tennent's edited collection of articles is particularly useful [13].

The languages are built over three base types: `comm`, the type of commands; `nat`, the type of natural-number valued expressions; and `var`, the type of assignable variables. Idealised Algol itself is an applied simply-typed lambda-calculus over these base types. Its types are therefore generated from the base types via product and function operations, and its syntax is that of the lambda-calculus together with a stock of constants for imperative programming with recursion, as follows.

- $0 : \mathtt{nat}$
- $\mathtt{succ} : \mathtt{nat} \to \mathtt{nat}$
- $\mathtt{pred} : \mathtt{nat} \to \mathtt{nat}$
- $\mathtt{ifzero}_B : \mathtt{nat} \times B \times B \to B$
- $\mathtt{Y}_A : (A \to A) \to A$

- $\mathtt{skip} : \mathtt{comm}$
- $\mathtt{assign} : \mathtt{var} \times \mathtt{nat} \to \mathtt{comm}$
- $\mathtt{deref} : \mathtt{var} \to \mathtt{nat}$
- $\mathtt{seq}_B : \mathtt{comm} \times B \to B$
- $\mathtt{while} : \mathtt{nat} \times \mathtt{comm} \to \mathtt{comm}$
- $\mathtt{new} : (\mathtt{var} \to \mathtt{comm}) \to \mathtt{comm}$
- $\mathtt{mkvar} : (\mathtt{nat} \to \mathtt{comm}) \times \mathtt{nat} \to \mathtt{var}$

where B ranges over base types and A over all types.

The constants in the left column provide for arithmetic and recursion, just as in PCF for example. The remainder provide a language of imperative programs: `skip` is the do-nothing command; $\texttt{assign}(V, N)$ stores the value of expression N in variable V; $\texttt{deref}(V)$ returns the value stored in V; $\texttt{seq}(C, M)$ executes command C and then evaluates M — note that the variant where B is `comm` is the familiar sequential composition of commands, while the others allows expressions of type `nat` or `var` to have side-effects on the store; $\texttt{while}(M, C)$ repeatedly executes C until M becomes non-zero; $\texttt{ifzero}_B$ provides for conditional branching; and $\texttt{new}(\lambda x : \texttt{var}.C)$ executes command C in an environment where identifier x has been bound to a fresh memory cell. Finally `mkvar` is a "variable constructor" which builds an expression of type `var` out of a "write-method" of type $\texttt{nat} \to \texttt{comm}$ and a "read method" of type `nat`. This object-oriented view of variables was proposed by Reynolds in his paper *The Essence of Algol* [16] and is by now standard. Indeed, some presentations including that of O'Hearn and Reddy go as far as identifying `var` with the product type $(\texttt{nat} \to \texttt{comm}) \times \texttt{nat}$. It has been shown that the presence of `mkvar` makes no difference to the notion of program equivalence in this language [9].

Interference-controlled Algol consists of the *affine* lambda-calculus, with products, over the same base types, with the same stock of constants, except that `Y` is excluded. The affine lambda-calculus restricts function application so that function and argument cannot share any identifiers. This leads to a situation in which distinct identifiers can never be bound to interfering program phrases, which makes program analysis more straightforward and was Reynolds's motivation for introducing the restricted language in *Syntactic Control of Interference* (SCI) [15]. The SCI type system was later refined by O'Hearn et al. [11], but we focus on the basic system here. Note that in this system, the two components of a product may still share identifiers and thus interfere, so the typing of constants such as `while` means that executing the body of a loop may have an impact on future execution of the guard. This is of course essential if one is to retain any interesting imperative programs.

These two languages can be equipped with an operational semantics, given in terms of *stores*: functions from locations to natural numbers. The operational semantics consists of judgements $s, M \Downarrow s', N$, stating that term M in store s evaluates to term N, altering the store to s'. Terms M and M' are *contextually equivalent* if, for every context $C[-]$ such that $C[M]$ and $C[M']$ are closed terms of type `comm`, $C[M] \Downarrow \texttt{skip}$ if and only if $C[M'] \Downarrow \texttt{skip}$; there are no stores in these judgements because closed terms operate in empty stores.

3 Object spaces (simplified)

Reddy's *object spaces* model of interference-controlled Algol [14] was the first to interpret higher-order imperative programs not using functions from starting states to finishing states, but by recording the *behaviour* of stateful objects using traces. As such, it is an important precursor of the game semantics of imperative programs[1,2] which were the first fully abstract models of Idealised Algol. Perhaps more surprisingly, Reddy's model is itself fully abstract for interference-controlled Algol, though this was not known until more recently [8]. In this section we present a simplified version of Reddy's category, which does away with the coherence relation at the expense of allowing nondeterminism into the model. Nondeterminism is a conservative extension of Idealised Algol, in both its regular and interference-controlled flavours, so this makes no difference to our results on full abstraction, while simplifying the technical details considerably.

Let A, B and C be sets. We use $\mathcal{P}A$ to denote the power-set of A, and A^* for the set of finite sequences drawn from A. Let $f : A \to \mathcal{P}B^*$ and $g : B \to \mathcal{P}C^*$ be any functions. Define the composite $g \circ f : A \to \mathcal{P}C^*$ as follows. For any $a \in A$, $(g \circ f)(a)$ is the set

$$\{s_1 \cdots s_n \mid \exists b_1, \ldots, b_n . b_1 \ldots b_n \in f(a) \wedge \forall i \in \{1, \ldots, n\} s_i \in g(b_i)\}$$

where $s_1 \cdots s_n$ is the concatenation of sequences $s_1, \ldots, s_n$.

This notion of composition is associative, and the map $A \to \mathcal{P}A^*$ which sends each a to $\{[a]\}$, the singleton set containing the singleton sequence, is an identity for it. We therefore have a category **FQ**, so named because it turns out to be the category of free quantales over sets.

Alternatively we may present this category as the Kleisli category $\mathbf{Set}_{\mathcal{P}T}$, where T is the monad delivering the free monoid on a set. The composite functor $\mathcal{P}T$ itself becomes a monad by virtue of a distributive law

$$\lambda : T\mathcal{P} \longrightarrow \mathcal{P}T$$

whose action is as follows. For any set A, λ_A takes a sequence $[S_1, \ldots, S_n]$ of subsets of A to the set of sequences

$$\{a_1 \ldots a_n \mid \forall i \in 1, \ldots, n . a_i \in S_i .\}$$

The first author has shown that the opposite of this category gives rise to a model of interference-controlled Algol which is essentially the same as Reddy's object spaces model, and that this model is fully abstract [8,10]. This presentation also has much in common with the development of Engeler-style models of the lambda-calculus in [5], and with the relational models presented in [3]; the connections remain to be explored.

In fact when modelling interference controlled Algol, it turns out to be more convenient to work with (the opposite of) the Kleisli category for $\mathcal{P}$ on T-Alg, the category of algebras for the monad T, that is, the category of monoids. This gives access to some additional objects which makes the formulation of the model simpler. However, for all intents and purposes $\mathbf{FQ}^{\mathrm{op}}$ is the category which houses the fully abstract model of interference-controlled Algol in *op. cit.*, and so it is this category we shall use.

3.1 Semantics of Interference-Controlled Algol

We now review the semantics of the types and constants of the language in $\mathbf{FQ}^{\mathrm{op}}$. Objects of the category are sets, and a map $A \to B$ can be seen as a relation between A^* and B. Disjoint union of sets gives a product in this category (because it gives a coproduct in **Set** and hence also in $\mathbf{Set}_{\mathcal{P}T}$). The larger category $(T{-}\mathrm{Alg}_{\mathcal{P}})^{\mathrm{op}}$ has a monoidal closed structure, where the tensor product is given by cartesian product of sets, and the internal hom $A \multimap B$ is given by the set $A^* \times B$. In $\mathbf{FQ}^{\mathrm{op}}$ the monoidal structure is not available, but the function types are, and this is enough to enable us to model the language.

The types are interpreted as shown below, where $\mathbf{N}$ denotes the set of natural numbers:

$$\begin{aligned}
[\![\mathtt{nat}]\!] &= \mathbf{N} \\
[\![\mathtt{comm}]\!] &= \{*\} \\
[\![\mathtt{var}]\!] &= \{\mathsf{write}(n), \mathsf{read}(n) \mid n \in \mathbf{N}\} \\
[\![A \times B]\!] &= [\![A]\!] + [\![B]\!] \\
[\![A \to B]\!] &= [\![A]\!]^* \times [\![B]\!].
\end{aligned}$$

(In some later calculations, we will identify the base types `comm`, `nat` and `var` with the sets they denote in this model.)

We now turn our attention to the constants. For each constant $c : A$ we define $[\![c]\!]$ as an element of $[\![A]\!]$. We use juxtaposition to denote concatenation of sequences, and identify a singleton sequence with its single element. When working with disjoint unions of sets, as in the semantics of product types, we use superscripts to indicate which component of the disjoint union a given element belongs to, so that for example in $\mathbf{N} + \mathbf{N} + \mathbf{N}$, n^3 is the value n in the rightmost component.

Note that a sequence $s \in \mathtt{var}^*$ consists of elements $\mathsf{write}(n)$ and $\mathsf{read}(m)$. We say that such a sequence is a *cell-trace* if the value in each $\mathsf{read}(-)$ entry matches the most recent $\mathsf{write}(-)$ entry: the kind of behaviour one would associate with a memory cell. If no $\mathsf{write}(-)$ has yet occurred, we allow $\mathsf{read}(0)$ to occur, modelling a cell with the default initial value of zero. Cell traces are therefore generated by the regular expression

$$\mathsf{read}(0)^* \cdot (\Sigma_{n\in\mathbf{N}}\mathsf{write}(n) \cdot (\mathsf{read}(n)^*))^* .$$

Armed with this, we give the semantics of constants as follows.

- $[\![0]\!] = 0$
- $[\![\mathtt{succ}]\!] = \{(n, n+1) \mid n \in \mathbf{N}\}$
- $[\![\mathtt{pred}]\!] = \{(n+1, n) \mid n \in \mathbf{N}\}$
- $[\![\mathtt{ifzero}_B]\!] = \{(0^1 v^2, v) \mid v \in B\} \cup \{(n^1 v^3, v) \mid n \in \mathbf{N}, n \neq 0, v \in B\}$
- $[\![\mathtt{skip}]\!] = *$

- $[\![\texttt{assign}]\!] = \{(n^2 \mathsf{write}(n)^1, *) \mid n \in \mathbf{N}\}$
- $[\![\texttt{deref}]\!] = \{(\mathsf{read}(n), n) \mid n \in \mathbf{N}\}$
- $[\![\texttt{seq}_B]\!] = \{(*^1 v^2, v) \mid v \in B\}$
- $[\![\texttt{while}]\!] = \{0^1 *^2 0^1 *^2 \ldots 0^1 *^2 m^1, *) \mid m \in \mathbf{N}, m \neq 0\}$
- $[\![\texttt{new}]\!] = \{((s, *), *) \mid s$ is a cell-trace$\}$.
- $[\![\texttt{mkvar}]\!] = \{(n^2, \mathsf{read}(n)) \mid n \in \mathbf{N}\} \cup \{((n \ldots n, *)^1, \mathsf{write}(n)) \mid n \in \mathbf{N}\}$

In the semantics of `while`, the expression $0^1{*}^2 0^1{*}^2 \ldots 0^1{*}^2$ denotes any element of the regular language $(0^1{*}^2)^*$, including the empty sequence; and similarly $n \ldots n$ in the semantics of `mkvar` denotes any sequence of the form $nnnnn \ldots n$.

4 O'Hearn and Reddy's model of Algol

In [12], O'Hearn and Reddy present a model of full Idealised Algol obtained from Reddy's object-spaces model by means of the (enriched) Yoneda embedding. The approach is as follows. To model Idealised Algol, one requires:

- a Cartesian closed category, to model the typed lambda-calculus
- a stock of objects to interpret the base types of the language, together with maps between them to interpret the constants forming the simple imperative language at the core of Idealised Algol
- a fixed point combinator to interpret recursion: for each type A, a map $[[\![A]\!] \Rightarrow [\![A]\!]] \to [\![A]\!]$ with appropriate properties
- a map $[[\![\texttt{var}]\!] \Rightarrow [\![\texttt{comm}]\!]] \to [\![\texttt{comm}]\!]$ to interpret `new`.

The object spaces model gives us enough to interpret the base types and the simple constants, but is not Cartesian closed: though it has products, it lacks exponentials; and in the absence of exponentials it does not make sense to ask whether fixed points and `new` can be interpreted.

O'Hearn and Reddy's proposed solution involves embedding the category of object spaces in a Cartesian closed category, using the Yoneda embedding. The idea is that an appropriate category of presheaves over the object-spaces model is Cartesian closed and contains the object spaces as a full subcategory. This immediately satisfies the first two requirements on a model. Fixed points are obtained by working with **Cpo**-enriched categories, and it turns out that direct consideration of the interpretation of `var` $\to$ `comm` allows the construction of the semantics of `new`.

Setting aside recursion for now, let us illustrate the semantics of `new` in the functor category $\mathbf{Set}^{\mathbf{FQ}}$. $\mathbf{FQ}^{\mathrm{op}}$ embeds fully in this category via the Yoneda embedding, with an object A represented by the covariant hom-functor $\mathbf{FQ}(A, -)$. Products are given pointwise. Apart from a size issue, which we will address later, the category would have exponentials: given functors F and

$G : \mathbf{FQ} \to \mathbf{Set}$, the exponential $[F \Rightarrow Q]$ is the functor taking an object X to the set of natural transformations

$$\mathbf{FQ}(X, -) \times F \longrightarrow G.$$

In particular, $[[\![\mathtt{var}]\!] \Rightarrow [\![\mathtt{comm}]\!]](X)$ is the set of natural transformations

$$\mathbf{FQ}(X, -) \times \mathbf{FQ}([\![\mathtt{var}]\!], -) \longrightarrow \mathbf{FQ}([\![\mathtt{comm}]\!], -).$$

Since $\mathbf{FQ}$ has coproducts given by disjoint union of sets, and using the Yoneda lemma again, this is the same as the hom-set

$$\mathbf{FQ}^{\mathrm{op}}(X + [\![\mathtt{var}]\!], [\![\mathtt{comm}]\!]).$$

So our interpretation of `new` must give, for each X and in a natural way, a map from elements of this hom-set, that is, relations between

$$(X + \{\mathsf{read}(n), \mathsf{write}(n) \mid n \in \mathbf{N}\})^* \quad \text{and} \quad \{*\}$$

to elements of $[\![\mathtt{comm}]\!][X] = \mathbf{FQ}^{\mathrm{op}}(X, \{*\})$, that is, relations between X^* and the singleton set. We define $[\![\mathtt{new}]\!]$ as follows:

$$[\![\mathtt{new}]\!][X](f) = \{(s \restriction X, *) \mid (s, *) \in f, s \restriction [\![\mathtt{var}]\!] \text{ is a cell-trace}\}.$$

Note both the similarity with the interpretation of `new` in the object-spaces model, and the difference: the key distinction that in this model, the sequence s contains not only the variable-events for reading and writing, but also events from the set X, in an interleaved fashion. It is this that makes the interpretation of Algol work: events from X interrupt the flow of events in the variable, and allow us to record interfering behaviour.

However, there is an important size issue to overcome. When claiming that exponentials exist, we asserted that $[F \Rightarrow G](X)$ is given by the *set* of natural transformations $\mathbf{FQ}(X, -) \times F \longrightarrow G$. Since $\mathbf{FQ}$ is not a small category, there is no reason to believe that this collection of natural transformations is a set at all. In the absence of a detailed argument to this effect, we cannot claim that $\mathbf{Set}^{\mathbf{FQ}}$ is Cartesian closed, so we cannot use it to model Idealised Algol.

O'Hearn and Reddy are aware of the size issue: their paper clearly states that the functor-category construction they use applies only to small categories. But the category of object spaces is not small, and as written, their work applies the construction directly to this category, so the essential property of Cartesian closure for their model has not been established.

The category of object spaces is locally small (hom-sets really are sets), so one could remedy this situation straightforwardly by restricting attention to the full subcategory on some set of objects. But what set should we choose? At this point a tension emerges between the desire to make the construction work and the desire to have a mathematically natural and appealing model. A quick fix could be provided by restricting to (for instance) just those objects used in the interpretation of base types and their products, but immediately one loses the distinction between the syntax and its model: good models should be constructed entirely independently of syntactic considerations.

A more satisfying path to solution lies in identifying a *small dense subcategory* $\mathbf{C}$ of $\mathbf{FQ}^{\mathrm{op}}$ and considering the functor category $\mathbf{Set}^{\mathbf{C}}$. This is the approach we take in the remainder of the paper.

5 A small dense subcategory

Let $\mathbf{C}$ be a category with a small full subcategory $\mathbf{D}$, with inclusion functor $J : \mathbf{D} \hookrightarrow \mathbf{C}$. Recall that $\mathbf{D}$ is *dense* in $\mathbf{C}$ if the functor $\mathbf{C} \to \mathbf{Set}^{\mathbf{D}^{\mathrm{op}}}$ taking X to $\mathbf{C}(J-, X)$ is full and faithful. Equivalently, every object of $\mathbf{C}$ arises as a colimit of $\mathbf{D}$-objects in a canonical way; see Mac Lane [7].

Our goal is to find a small dense subcategory of $\mathbf{FQ}^{\mathrm{op}}$ closed under finite products. The category of presheaves over this will then be Cartesian closed, and $\mathbf{FQ}^{\mathrm{op}}$ will fully embed in this category in a product-preserving fashion. We will then be able to provide semantics for Idealised Algol along the lines indicated above.

As a first attempt, consider the full subcategory of $\mathbf{FQ}^{\mathrm{op}}$ whose objects are finite sets. We call this category $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$. It can equivalently be given as the subcategory with objects the natural numbers, that is, finite sets of the form $\{0, 1, \ldots, n\}$ for some $n \in \mathbf{N}$, which is clearly a small category. Every object of $\mathbf{FQ}^{\mathrm{op}}$ is a colimit of a diagram of these objects. Given sets A and B with $A \subseteq B$, the relation taking an element $a \in A$ to the singleton sequence $[a]$ provides maps in $\mathbf{FQ}^{\mathrm{op}}$ from A to B and back:

$$\begin{aligned} \mathsf{in}_{A,B} : A \to B &= \{([a], a) \mid a \in A\} \\ \mathsf{out}_{B,A} : B \to A &= \{([a], a) \mid a \in A\} \end{aligned}$$

Note that $\mathsf{out}_{B,A} \circ \mathsf{in}_{A,B} = \mathsf{id}_A$, and that if $A \subseteq B \subseteq C$ then $\mathsf{in}_{A,C} = \mathsf{in}_{B,C} \circ \mathsf{in}_{A,B}$, and similarly for the out maps. Now given any object X of $\mathbf{FQ}^{\mathrm{op}}$, the diagram consisting of all finite subsets of X and all $\mathsf{in}_{A,B}$ maps between them has colimit X, with the maps $\mathsf{in}_{A,X}$ forming the colimiting cone. Thus one might expect that the finite sets are dense in $\mathbf{FQ}^{\mathrm{op}}$, but unfortunately this is not the case, as the following counterexample shows.

Consider the set $\mathbf{R}$ of real numbers. If $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ is to be dense in $\mathbf{FQ}^{\mathrm{op}}$, there should be a bijective correspondence between maps $\mathbf{R} \to 1$ in $\mathbf{FQ}^{\mathrm{op}}$ and natural transformations

$$\mathbf{FQ}^{\mathrm{op}}(J-, \mathbf{R}) \to \mathbf{FQ}^{\mathrm{op}}(J-, 1)$$

where 1 is the singleton set. Given such a natural transformation α and sets $X \subseteq Y \subseteq \mathbf{R}$, naturality implies that $\alpha(\mathsf{in}_{Y,\mathbf{R}}) \circ \mathsf{in}_{X,Y} = \alpha(\mathsf{in}_{Y,\mathbf{R}} \circ \mathsf{in}_{X,Y}) = \alpha(\mathsf{in}_{X,\mathbf{R}})$. Thus the maps $\alpha(\mathsf{in}_{X,\mathbf{R}}) : X \to 1$ form a cocone over the $\mathsf{in}_{X,Y}$, and hence there is a unique map $f : \mathbf{R} \to 1$ such that $f \circ \mathsf{in}_{X,\mathbf{R}} = \alpha(\mathsf{in}_{X,\mathbf{R}})$ for every finite $X \subseteq \mathbf{R}$. Density of $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ amounts to the statement that $\alpha(g) = f \circ g$ for all maps $g : X \to \mathbf{R}$, rather than just for the in maps.

We now define a natural transformation α for which this does not hold. Given a finite set X and a map $f : X \to \mathbf{R}$ in $\mathbf{FQ}^{\mathrm{op}}$, $\alpha(f)$ is the relation on $X^* \times 1$ defined by

$$\{(s,*) \mid (s,x) \in f \text{ for uncountably many } x\}.$$

To see that this is natural, consider any $f : X \to \mathbf{R}$ and $g : Y \to X$. If some $(t,*) \in \alpha(f) \circ g$, there must be $(s,*) \in \alpha(f)$ such that $s = x_1 \dots x_n$, $t = t_1 \dots t_n$ and for each i, $(t_i, s_i) \in g$. This means there are uncountably many x such that $(s,x) \in f$, and hence also $(t,x) \in f \circ g$ for uncountably many x, so that $(t,*) \in \alpha(f \circ g)$. Hence $\alpha(f) \circ g \subseteq \alpha(f \circ g)$. For the other inclusion, suppose $(t,*) \in \alpha(f \circ g)$ for some t. Then there are uncountably many x such that $(t,x) \in f \circ g$. For each such x there is some s^x such that $(s^x, x) \in f$, and $s^x = s^x_1 \dots s^x_n$, $t = t^x_1 \dots t^x_n$ with each $(t^x_i, s^x_i) \in g$. But the s^x are drawn from the countable set X^*, so there must be some s such that $s^x = s$ for uncountably many x. It follows that $(s,x) \in f$ for uncountably many x, hence $(s,*) \in \alpha(f)$ and then $(t,*) \in \alpha(f) \circ g$ as required.

However, for every finite $X \subseteq \mathbf{R}$, $\alpha(\mathsf{in}_{X,\mathbf{R}}) = \emptyset$. The unique mediating map $f : \mathbf{R} \to 1$ is also the empty relation, but it is clearly not the case that $\alpha(g) = \emptyset \circ g$ for all g. Hence $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ is not dense in $\mathbf{FQ}^{\mathrm{op}}$.

We do not know of a straightforward way to remedy this situation in the non-enriched case. Fortunately, as should be clear from the above counterexample, the failure of density results from a discontinuity in the components of the natural transformation α, which can be eliminated by working with **Cpo**-enriched categories rather than ordinary categories. Since we intended all along to move to the **Cpo** setting, so as to recover a semantics of recursion, this is harmless, though the picture is perhaps more delicate than one would like.

We let **Cpo** denote the category of directed-complete partial orders, possibly without bottom element, and continuous functions. Recall that a **Cpo**-category has hom-cpos rather than hom-sets, with composition being a continuous function on hom-cpos. The categories $\mathbf{FQ}$, $\mathbf{FQ}^{\mathrm{op}}$ and $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ can be seen as a **Cpo**-categories: given sets X and Y, $\mathbf{FQ}(X,Y)$ is the collection of relations between X and Y^*, ordered by inclusion. It is straightforward to verify that composition is continuous.

The **Cpo**-functor-category $\mathbf{Cpo}^{\mathbf{FQ}}$ has as its objects all **Cpo**-functors from **FQ** to **Cpo**, that is, functors whose actions on morphisms are continuous. Maps between **Cpo**-functors boil down to ordinary natural transformations between the underlying ordinary functors; but note that the components of such natural transformations are maps in **Cpo** and therefore continuous functions. In this setting, $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ is a dense subcategory of $\mathbf{FQ}^{\mathrm{op}}$.

Theorem 5.1 $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ *is dense in* $\mathbf{FQ}^{\mathrm{op}}$. *Dense here means that the functor from* $\mathbf{FQ}^{\mathrm{op}}$ *to* $\mathbf{Cpo}^{\mathbf{FQ}_{\mathrm{f}}}$ *taking an object* X *to* $\mathbf{FQ}^{\mathrm{op}}(J-, X) : \mathbf{FQ}_{\mathrm{f}} \to \mathbf{Cpo}$ *is* *fully faithful as a* **Cpo***-functor, i.e. that each of its actions on hom-cpos is an order isomorphism.*

Proof. We must show that the action of the functor in question on hom-cpos is injective, surjective and reflects order.

For injectivity, observe that if $f, g : X \to Y$ are distinct maps of $\mathbf{FQ}^{\mathrm{op}}$, there must be some pair $(s, y) \in X^* \times Y$ that appears in f but not g (or vice versa). We must show that the natural transformations $\mathbf{FQ}^{\mathrm{op}}(J-, f)$ and $\mathbf{FQ}^{\mathrm{op}}(J-, g)$ are distinct. Let $A \subseteq X$ be any finite subset that contains all the elements of s. Instantiating the natural transformations at A and applying to the morphism $\mathsf{in}_{A,X}$ gives the two maps $f \circ \mathsf{in}_{A,X}$ and $g \circ \mathsf{in}_{A,X}$, and one of these contains the pair (s, y) while the other does not, as required. A similar argument shows that the functor's action on hom-cpos reflects order.

More interesting is the question of surjectivity. Let $\alpha : \mathbf{FQ}^{\mathrm{op}}(J-, X) \to \mathbf{FQ}^{\mathrm{op}}(J-, Y)$ be any natural transformation. Just as in the non-enriched case, there is a unique $f : X \to Y$ such that $\alpha(\mathsf{in}_{A,X}) = f \circ \mathsf{in}_{A,X}$ for all finite $A \subseteq X$: concretely, this f consists of all those pairs (s, y) such that $(s, y) \in \alpha(\mathsf{in}_{A,X})$ for some A. It remains to show that $\alpha = \mathbf{FQ}^{\mathrm{op}}(J-, f)$, that is, that $\alpha(g) = f \circ g$ for *any* finite set A and map $g : A \to X$.

But for any finite $B \subseteq X$, applying the naturality square for α

$$\begin{tikzcd}
\mathbf{FQ}^{\mathrm{op}}(JB, X) \arrow[r, "\alpha_B"] \arrow[d, "\mathbf{FQ}^{\mathrm{op}}(\mathsf{out}_{X,B} \circ g, X)"'] & \mathbf{FQ}^{\mathrm{op}}(JB, Y) \arrow[d, "\mathbf{FQ}^{\mathrm{op}}(\mathsf{out}_{X,B} \circ g, Y)"] \\
\mathbf{FQ}^{\mathrm{op}}(JA, X) \arrow[r, "\alpha_A"'] & \mathbf{FQ}^{\mathrm{op}}(JA, Y)
\end{tikzcd}$$

to the element $\mathsf{in}_{B,X}$ of $\mathbf{FQ}^{\mathrm{op}}(JB, X)$ means that $\alpha_A(\mathsf{in}_{B,X} \circ \mathsf{out}_{X,B} \circ g) = \alpha_B(\mathsf{in}_{B,X}) \circ \mathsf{out}_{X,B} \circ g = f \circ \mathsf{in}_{B,X} \circ \mathsf{out}_{X,B} \circ g$. The set of maps

$$\{\mathsf{in}_{B,X} \circ \mathsf{out}_{X,B} \circ g : A \to X \mid B \subseteq X, B \text{ finite}\}$$

is directed, with supremum g. Hence by continuity of α_A, $\alpha_A(g) = f \circ g$ as required. □

The situation at which we have arrived is satisfactory, but somewhat delicate. The functor category $\mathbf{Cpo}^{\mathbf{FQ}_{\mathrm{f}}}$ is locally small and Cartesian closed, and contains $\mathbf{FQ}^{\mathrm{op}}$ as a full subcategory, so we are in a position to give semantics to Idealised Algol as explained by O'Hearn and Reddy. The move to **Cpo**-categories was on the agenda from the outset, because of the desire to model recursion using fixed points. However, **Cpo**-enrichment plays a double role, because $\mathbf{FQ}_{\mathrm{f}}^{\mathrm{op}}$ fails to be dense without it; this makes the construction less flexible, and the fact that we have used directed completeness rather than ω-completeness in the density argument raises difficulties, because **Cpo** is not locally presentable while $\omega\mathbf{Cpo}$ is. Nevertheless, we have identified a small category of “worlds”, namely $\mathbf{FQ}_\mathrm{f}$, whose functor category provides a model of Idealised Algol.

Theorem 5.2 *The model of Idealised Algol in* $\mathbf{Cpo}^{\mathbf{FQ}_\mathrm{f}}$ *is fully abstract up to order two.*

Proof. The proof follows that of O’Hearn and Reddy. Some small alterations are needed because $\mathbf{FQ}^\mathrm{op}$ lacks the coherence structure of Reddy’s original object spaces model, so contains some nondeterministic elements. However, adding nondeterminism to Idealised Algol is a conservative extension (see [10] for instance) so this makes no difference to the result. $\square$

6 Failure of full abstraction at order 3

We now give a counterexample to full abstraction, that is to say, a pair of terms which are contextually equivalent in Idealised Algol but have distinct denotations in the functor-category model. The type of these two terms are of third order, so we have a sharp result: the model is fully abstract to order two, but no further.

The terms are of the form $\lambda f : (\mathtt{comm} \to \mathtt{comm}) \to \mathtt{comm}.M_i$ where M_1 and M_2 are defined by:

$$\begin{aligned} M_1 &= \mathtt{new}(\lambda x.f(\lambda c.x := 0; c; x :=!x + 1; c; \mathtt{if}\ !x \geq 2\ \mathtt{then}\ \Omega\ \mathtt{else}\ x := 0) \\ M_2 &= f(\lambda c.c; c) \end{aligned}$$

Here we make use of some syntactic sugar, using infix operators ; and := in place of the functions `seq` and `assign`, ! for `deref`, and standard arithmetic and boolean operations which are readily definable.

Lemma 6.1 *The terms* $\lambda f.M_1$ *and* $\lambda f.M_2$ *are contextually equivalent.*

Proof. Equivalence of these terms is established by analysing their denotations in the fully abstract game semantics [2]. We shall not detail the games model here, but give the equivalence proof for those readers familiar with the model.

The game denotation of the term

$$f(\lambda c.x := 0; c; x :=!x + 1; c; \mathtt{if}\ !x \geq 2\ \mathtt{then}\ \Omega\ \mathtt{else}\ x := 0)$$

is a strategy on the game

$$((\mathtt{comm}_4 \to \mathtt{comm}_3) \to \mathtt{comm}_2) \times \mathtt{var} \to \mathtt{comm}_1.$$

Subscripts on the `comm` games are used to identify the four occurrences, and we tag their run and done moves similarly. The strategy plays as follows:

- when O plays run_1, play run_2
- when O plays done_2, play done_1

- when O plays run_3, play $\mathsf{write}(0)$, and when O then plays ok, play run_4. We refer to this occurrence of run_4 as a *first invocation* of the move.
- when O plays done_4 in response to a first invocation of run_4, play read; O then plays some value n, and P plays $\mathsf{write}(n+1)$; O then plays ok, and P plays run_4 — this is a *second invocation.*
- when O plays done_4 in response to a second invocation of run_4, play read; when O plays a value n, play $\mathsf{write}(0)$ if $n \in \{0,1\}$, otherwise make no response; when O plays ok, play done_3.

If we strip out all the `var` moves from these plays, we obtain the strategy for the term $f(\lambda c.c;c)$. Therefore, to establish equivalence we need only show that if O's play in `var` is a cell-trace, then O's answer n to the read in the final paragraph above is always 0 or 1. This is done by induction on the length of a play according to the above strategy: we show that

- when a first invocation of run_4 is played, the last write carried value 0
- when a second invocation of run_4 is played, the last write carried value 1
- when done_4 is played in response to a first invocation, the last write carried value 0
- when done_4 is played in response to a second invocation, the last write carried value 0 or 1
- when done_3 is played, the last write carried value 0.

The interesting cases concern the done_4 moves. When done_4 is played, the previous move must have been either run_4, so the last write was either 0 (for a first invocation) or 1 (for a second invocation); or done_3, in which case the last write was 0. Thanks to the bracketing discipline for question and answer moves in the games model, these are the only two possibilities, so the proof is complete. $\square$

Lemma 6.2 *The denotations of $\lambda f.M_1$ and $\lambda f.M_2$ are distinct in the model in $\mathbf{Cpo}^{\mathbf{FQ}_\mathrm{f}}$.*

Proof. The terms will be distinguished by applying them to a particular element of the denotation of $(\mathtt{comm} \to \mathtt{comm}) \to \mathtt{comm})$. This is most readily described as a natural transformation $\alpha : [\![\mathtt{comm} \to \mathtt{comm}]\!] \longrightarrow [\![\mathtt{comm}]\!]$. Its component at X is the map

$$\alpha_X : \mathbf{FQ}_\mathrm{f}^{\mathrm{op}}(X + \mathtt{comm}, \mathtt{comm}) \longrightarrow \mathbf{FQ}_\mathrm{f}^{\mathrm{op}}(X, \mathtt{comm})$$

which takes a map g to the relation

$$\{(s_1 s_1' s_2 s_2' s_3 s_3', *) \mid (s_1 * s_2 * s_3, *) \in g, (s_1' * s_2' * s_3', *) \in g\}.$$

It is straightforward to verify that this is a natural transformation.

The denotation of the argument term

$$\lambda c.x := 0; c; x := !x + 1; c; \mathtt{if}\ !x \geq 2\ \mathtt{then}\ \Omega\ \mathtt{else}\ x := 0$$

is the natural transformation $[\![\texttt{var}]\!] \to [\![\texttt{comm} \to \texttt{comm}]\!]$ whose component at X is given by the map $\mathbf{FQ}^{\mathrm{op}}(X, \texttt{var}) \to \mathbf{FQ}^{\mathrm{op}}(X + \texttt{comm}, \texttt{comm})$ which takes a map h to the set of pairs $(s_1 * s_2 s_3 * s_4, *)$ where h contains $(s_1, \mathsf{write}(0))$, $(s_2, \mathsf{read}(m))$, $(s_3, \mathsf{write}(m+1))$ and (s_4, n) for some $m \in \mathbf{N}$ and $n \in \{0,1\}$.

Composing these two gives us the denotation of

$$f(\lambda c.x := 0; c; x :=!x + 1; c; \texttt{if } !x \geq 2 \texttt{ then } \Omega \texttt{ else } x := 0)$$

when f is bound to α. It is the natural transformation $[\![\texttt{var}]\!] \to [\![\texttt{comm}]\!]$ whose component at X is the map $\mathbf{FQ}^{\mathrm{op}}(X, \texttt{var}) \to \mathbf{FQ}^{\mathrm{op}}(X, \texttt{comm})$ taking h to the relation consisting of all pairs $(s_1 s_1' s_2 s_3 s_2' s_3' s_4 s_4', *)$ where h contains the pairs $(s_1, \mathsf{write}(0))$, $(s_1', \mathsf{write}(0))$, $(s_2, \mathsf{read}(m))$, $(s_2', \mathsf{read}(m'))$, $(s_3, \mathsf{write}(m+1))$, $(s_3', \mathsf{write}(m'+1))$, (s_4, n) and (s_4', n') for some $m, m' \in \mathbf{N}$ and $n, n' \in \{0,1\}$.

By our density result, this natural transformation must correspond to an element of $\mathbf{FQ}^{\mathrm{op}}(\texttt{var}, \texttt{comm})$, and the appropriate element is the relation containing all pairs

$$(\mathsf{write}(0)\mathsf{write}(0)\mathsf{read}(m)\mathsf{write}(m+1)\mathsf{read}(m')\mathsf{write}(m'+1)\mathsf{read}(n)\mathsf{read}(n'), *)$$

where $m, m' \in \mathbf{N}$ and $n, n' \in \{0,1\}$. This relation is the denotation of

$$\lambda x.f(\lambda c.x := 0; c; x :=!x + 1; c; \texttt{if } !x \geq 2 \texttt{ then } \Omega \texttt{ else } x := 0).$$

Finally, applying `new` to this results in the empty set, because none of the sequences of `var`-actions in the above relation is a cell-trace. We therefore conclude that applying $[\![\lambda f.M_1]\!]$ to α results in the empty relation.

On the other hand, a similar and much simpler calculation shows that applying $[\![\lambda f.M_2]\!]$ to α gives the relation $\{(\varepsilon, *)\}$ (the denotation of `skip`), which is non-empty, completing the proof. $\square$

Corollary 6.3 *The model of Idealised Algol in* $\mathbf{Cpo}^{\mathbf{FQ}_{\mathrm{f}}}$ *is not fully abstract for terms of order 3.*

7 Future work

Taking our lead from O'Hearn and Reddy, we have shown how a small and simple category may be used as a category of worlds in a functor-category model of Idealised Algol. As O'Hearn and Reddy remarked, these "worlds" are of a rather different character to others in the literature, which generally correspond to the possible shapes of the store. We suggest that worlds in our model be understood as describing possible *observations* one can make of programs. Perhaps there are other notions of world following this idea which give rise to interesting and useful models, and connections with presheaf models of concurrency [4].

Though we claim that our simplified notion of object space gives rise to essentially the same model as O'Hearn and Reddy's approach, it is nevertheless natural to ask whether one can find a small dense subcategory of Reddy's original category of object-spaces. We believe that a similar approach will work, though we have not studied it in detail. Perhaps more interesting is to ask whether the fully abstract games models can be reconciled with the functor category approach: is there a fruitful way of viewing the games model as a functor category?

References

[1] Abramsky, S. and G. McCusker, *Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions (extended abstract)*, in: *Proceedings of 1996 Workshop on Linear Logic*, Electronic notes in Theoretical Computer Science **3** (1996), pp. 2–14.

[2] Abramsky, S. and G. McCusker, *Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions*, in: O'Hearn and Tennent [13] pp. 297–329 of volume 2.

[3] Bucciarelli, A., T. Ehrhard and G. Manzonetto, *A relational model of a parallel and non-deterministic λ-calculus*, in: *LFCS '09: Proceedings of the 2009 International Symposium on Logical Foundations of Computer Science* (2009), pp. 107–121.

[4] Cattani, G. L. and G. Winskel, *Presheaf models for concurrency*, in: *Computer Science Logic: 10th International Workshop Proceedings*, Lecture Notes in Computer Science **1258** (1997), pp. 58–75.

[5] Hyland, M., M. Nagayama, J. Power and G. Rosolini, *A category-theoretic formulation of engeler-style models of the untyped λ-calculus*, in: *Proc. MCFSIT 2004, Electronic Notes in Theoretical Computer Science volume 161*, 2006, pp. 43–57.

[6] Kelly, G. M., "Basic Concepts of Enriched Category Theory," Cambridge University Press, 1982.

[7] Mac Lane, S., "Categories for the Working Mathematician," Springer-Verlag, Berlin, 1971.

[8] McCusker, G., *A fully abstract relational model of syntactic control of interference*, in: *Proceedings, Computer Science Logic (CSL) 2002*, Lecture Notes in Computer Science **2471** (2002), pp. 247–261.

[9] McCusker, G., *On the semantics of the bad variable constructor in Algol-like languages*, in: S. Brookes and P. Panangaden, editors, *Proceedings, Nineteenth Conference on the Mathematical Foundations of Programming Semantics, Montreal 2003*, Electronic Notes in Theoretical Computer Science (2003).

[10] McCusker, G., *A graph model of imperative computation*, Logical Methods in Computer Science **6** (2010), pp. 1–35, doi:10.2168/LMCS-6(1:2)2010.

[11] O'Hearn, P. W., A. J. Power, M. Takeyama and R. D. Tennent, *Syntactic control of interference revisited*, Theoretical Computer Science **228** (1999), pp. 211–252.

[12] O'Hearn, P. W. and U. Reddy, *Objects, interference and the Yoneda embedding*, in: M. Main and S. Brookes, editors, *Mathematical Foundations of Programming Semantics: Proceedings of 11th International Conference*, Electronic Notes in Theoretical Computer Science (1995), pp. 487–514.

[13] O'Hearn, P. W. and R. D. Tennent, editors, "Algol-like Languages," Birkhaüser, 1997.

[14] Reddy, U. S., *Global state considered unnecessary: Object-based semantics for interference-free imperative programs*, Lisp and Symbolic Computation **9** (1996), pp. 7–76.

[15] Reynolds, J. C., *Syntactic control of interference*, in: *Conf. Record 5th ACM Symposium on Principles of Programming Languages*, 1978, pp. 39–46.

[16] Reynolds, J. C., *The essence of Algol*, in: *Proceedings of the 1981 International Symposium on Algorithmic Languages* (1981), pp. 345–372.