A fully abstract game semantics for finite nondeterminism

Russell Harmer
Imperial College, London
email: rsh2@doc.ic.ac.uk

Guy McCusker
University of Sussex
email: guym@cogs.susx.ac.uk

Abstract

A game semantics of finite nondeterminism is proposed. In this model, a strategy may make a choice between different moves in a given situation; moreover, strategies carry extra information about their possible divergent behaviour. A Cartesian closed category is built and a model of a simple, higher-order nondeterministic imperative language is given. This model is shown to be fully abstract, with respect to an equivalence based on both safety and liveness properties, by means of a factorization theorem which states that every nondeterministic strategy is the composite of a deterministic strategy with a nondeterministic oracle.

1 Introductory remarks

Nondeterminism, the notion that the behaviour of a computer system need not be completely determined by the behaviour of its environment, is a valuable abstraction in the analysis of programs. An unreliable hardware component, governed by laws of physics too complex to take into account, can be understood as a nondeterministic entity; multi-threaded programs in which resources are shared between threads may well be entirely deterministic if we know all the details of the operating system scheduling the threads, but are better considered as exhibiting nondeterminism; and a specification which may be satisfied by many different programs can fruitfully be viewed as a single nondeterministic process.

The importance of these manifestations of nondeterminism has long been recognized in semantics and various approaches have been proposed to account for such features. At one end of the spectrum, traditional denotational models embrace nondeterminism by modelling programs not as *functions* from input to output but as *relations*, typically realised by means of powerdomain constructions [20, 24]. In the study of concurrency, however, programs are modelled directly as processes which carry an in-built notion of nondeterminism: at a given point in time, there may be many things which a process "can do next" [12, 18].

Game semantics In recent years, the paradigm of *game semantics* has gained in popularity as a middle ground between domain-theoretic semantics and models of concurrency. Games provide models of programming languages enjoying much of the structure and elegance of domain-based models, but built out of processes which are recognisable descendants of those used in concurrency.

The main novelty of the approach is that an explicit distinction is made between the actions performed by a process itself, and those of its environment. The *intensional* flavour of game semantics has proved valuable: in addition to providing insight, it has allowed various *full abstraction* results to be obtained, showing that games models capture the behaviour of programs in a precise way. Furthermore, the paradigm is flexible enough to furnish fully abstract models of a wide variety of languages, ranging from purely functional languages [1, 4, 13, 16, 17, 19], to languages with control operators [15], mutable variables [3, 5], and even general references [2]. However, despite being grounded on mathematical structures usually associated with concurrency, all of this work has focussed on sequential, deterministic languages.

Contribution In this paper we present an account of nondeterminism in the setting of game semantics, and obtain a full abstraction result for a nondeterministic language.

Our starting point is Reynolds' higher-order imperative language Idealised Algol [21], and its fully abstract games model [5]. In this model, a term is interpreted as a strategy on a certain game, constrained in such a way that "what to do next" is completely determined by what has happened so far. Following a common pattern in game semantics [6], we extend the scope of the model by relaxing this constraint, obtaining a model which encompasses nondeterminism. This model is no longer fully abstract for Idealised Algol, of course, since that language is deterministic; but by adding a simple *erratic choice* operator to Idealised Algol, the balance is restored: our model is fully abstract with respect to a notion of equivalence which captures the "may-converge" predicate, so that two programs are considered equivalent if they can produce the same range of output val-

ues.

This notion of equivalence is, however, far from definitive since it fails to distinguish a program which may either produce the number 1 or go into an infinite loop from the program which always terminates with the output 1. In other words, it gives no account of *liveness* properties as embodied in the "must-converge" predicate. To address this issue, we add to our strategies some additional information in the form of CSP-style *divergences* [12]. Divergences tell us precisely those situations in which the environment may push a program "over the edge" into an infinite loop, allowing us to obtain a model which is fully abstract for a notion of equivalence taking both "may-converge" and "must-converge" predicates into account.

In common with other results in game semantics, the proofs of full abstraction for these extended models work by factorizing a strategy from the extended model into one from the smaller model, already known to be fully abstract for the smaller language, and a particular canonical element of the larger model. In this case, the factorization makes precise the idea that a nondeterministic process can be considered to be a deterministic process operating in tandem with a nondeterministic *oracle*, which "tosses a coin" whenever a nondeterministic choice is to be made.

Related work Closely related to our work is the paper of Hennessy & Ashcroft [11] which considers essentially the same language and notion of equivalence as ours, but without the imperative features. There it is shown that a model based on the Plotkin powerdomain is sound but fails to be fully abstract. In fact, we believe that our full abstraction result is the first of its kind. Nondeterministic strategies have also been considered in [8] for quite different reasons.

Work in progress The work reported here should only be considered a preliminary examination of nondeterminism in the games setting; several important questions remain unanswered. First, although deterministic game semantics handles pure functional languages very well, we do not know how to give a fully abstract model of a nondeterministic functional language such as that considered by Hennessy & Ashcroft: the obvious nondeterministic extension of the fully abstract model of PCF does not work for technical reasons [10]. The reason for this failure seems to be an inadequate representation of branching-time information, so it may be that moving to a tree-based formulation of game semantics will solve this problem. Second, our model of must-convergence can only handle finite nondeterminism. It is well known that countable nondeterminism, as exhibited by a program which may produce any integer but never diverges, raises problems with continuity [7] and makes the familiar operation of hiding in CSP troublesome [23]; these same problems arise in the games model as non-continuity and non-associativity of composition, so we are forced to restrict our attention to a class of finitely branching strategies; Roscoe's work on infinite traces in CSP provides some hope for a solution to this. Finally, although the study of nondeterminism is often seen as a stepping stone to concurrency, it is by no means clear that the ideas presented here lead to a good model of concurrent features. That said, it may be hoped that the distinctive approach of game semantics will bear fruit of its own; but that remains to be seen.

2 Erratic Idealized Algol

Our starting point is an Idealized Algol, *i.e.* PCF extended to allow the representation of typical imperative features, such as command sequencing, assignment and variable allocation. The types are defined inductively:

$$T \quad ::= \quad \mathtt{Nat} \mid \mathtt{Com} \mid \mathtt{Var} \mid T \to T,$$

as are the terms:

$$\begin{array}{rcl} M & ::= & x \mid \lambda x^T.\, M \mid M\, M \mid \mathbf{Y}_T\ M \mid \\ & & \mathtt{n} \mid \mathtt{skip} \mid \mathtt{succ}\, M \mid \mathtt{pred}\, M \mid \\ & & \mathtt{if0}\ M\ M\ M \mid \mathtt{seq}_T\ M\ M \mid \\ & & \mathtt{assign}\, M\ M \mid \mathtt{deref}\ M \mid \\ & & \mathtt{new}_T\ M \mid \mathtt{mkvar}\, M\ M. \end{array}$$

Here x ranges over an inexhaustible supply of variables and $\mathtt{n}$ over $n \in \mathbf{N}$; we tend to use v to range over variables of type $\mathtt{Var}$. We call the language IA.

We add a single term constructor to the language for the representation of erratic nondeterminism:

$$M \quad ::= \quad \cdots \mid M\ \mathtt{or}\ M.$$

We call the extended language *erratic* Idealized Algol, or EIA for short. The typing rule for $\mathtt{or}$ is the following. Note that we restrict nondeterminism to the natural number type; we'll see later that this entails no loss of generality.

$$\frac{\Gamma \vdash M : \mathtt{Nat} \qquad \Gamma \vdash N : \mathtt{Nat}}{\Gamma \vdash M\ \mathtt{or}\ N : \mathtt{Nat}}$$

The canonical forms of the language are:

$$V \quad ::= \quad \mathtt{skip} \mid \mathtt{n} \mid \lambda x.\, M \mid v \mid \mathtt{mkvar}\, M\ M.$$

A selection of the rules defining the operational semantics is given in Figure 1. In these rules, $\langle s, M\rangle$ is a *configuration* in which M is a term, all of whose free variables have type $\mathtt{Var}$, and s is a *store, i.e.* a function from the free variables of M to the natural numbers. If s is a store and v is a variable, we write $\langle s \mid v \mapsto n\rangle$ for the store s updated to map v to n. This may override a previous value for v or extend the domain of definition of s.

The relation $\langle s, M\rangle \Downarrow \langle s', V\rangle$ should be read as "starting from state s, the term M *may* converge to V, finishing in state s'". The predicate $\langle s, M\rangle \Downarrow^{\mathrm{must}}$ reads as "starting from state s, the term M *must* converge to something". If M is closed, we abbreviate these to $M \Downarrow V$ and $M \Downarrow^{\mathrm{must}}$.

Let M and N be closed terms of the same type. We write $M \simeq_{\mathrm{may}} N$ iff for all contexts $C[-]$ of ground type,

$$C[M] \Downarrow V \iff C[N] \Downarrow V$$

and say that M and N are ***may-equivalent***. We write $M \simeq_{m\&m} N$ iff for all contexts $C[-]$ of ground type,

$$\begin{array}{rcl} C[M] \Downarrow V & \iff & C[N] \Downarrow V \quad \wedge \\ C[M] \Downarrow^{\mathrm{must}} & \iff & C[N] \Downarrow^{\mathrm{must}} \end{array}$$

and say that M and N are ***M&M-equivalent***, an abbreviation for "may & must equivalent".

3 Game semantics

Game semantics models computation as an interaction between two agents, mediated by certain rules. This interaction is represented by sequences of "moves". First, a little notation: if s and t are strings on some alphabet Σ, we write $s \sqsubseteq t$ for the prefix ordering; the least element of this, *i.e.* the empty string, is denoted by ε; if $\Sigma' \subseteq \Sigma$, we write $s \restriction \Sigma'$ for the *restriction* of s to Σ', *i.e.* the subsequence obtained by removing all symbols of s that aren't from Σ.

3.1 Arenas

Roughly speaking, a game is played in an *arena* which sets out certain basic conditions and ground rules. Play proceeds between two protagonists, Opponent (the Environment) and Player (the System), who take turns to make moves.

Formally, an ***arena*** is a triple $\langle M_A, \lambda_A, \vdash_A\rangle$ where

- M_A is a countable set of ***moves***.
- $\lambda_A : M_A \to \{\mathsf{O}, \mathsf{P}\} \times \{\mathsf{Q}, \mathsf{A}\}$ is a ***labelling*** function designating each $m \in M_A$ as an Opponent move or a Player move and as a Question or an Answer. It's useful to define:

$$\begin{array}{rcl} \lambda_A^{\mathsf{OP}} & = & \mathsf{fst} \circ \lambda_A \\ \lambda_A^{\mathsf{QA}} & = & \mathsf{snd} \circ \lambda_A\,. \end{array}$$

  $\overline{\lambda}_A$ is the labelling function that reverses the roles of Opponent and Player, leaving Questions and Answers unchanged.

- $\vdash_A$ is an ***enabling*** relation on $M_A \times M_A$ satisfying
  - $a \vdash_A b \wedge a \neq b \Rightarrow \lambda_A^{\mathsf{OP}}(a) \neq \lambda_A^{\mathsf{OP}}(b)$.
  - $a \vdash_A a \Rightarrow \lambda_A(a) = \mathsf{OQ} \wedge (b \neq a \Rightarrow b \not\vdash_A a)$.
  - $a \vdash_A b \wedge \lambda_A^{\mathsf{QA}}(b) = \mathsf{A} \Rightarrow \lambda_A^{\mathsf{QA}}(a) = \mathsf{Q}$.

  If $m \vdash_A m$ then m is ***initial*** in A.

The simplest arena is the "empty arena" $\mathbf{1} = \langle \varnothing, \varnothing, \varnothing\rangle$. More interesting is the ***flat arena*** for natural numbers, written $\mathbf{N}$, which has one Opponent Question, q, and one Player Answer n for each natural number. This notion of flat arena evidently generalizes to any (countable) set so, in particular, we have an arena $\mathbf{B}$ for the Booleans. The flat arena for the singleton set will be called $\mathbf{C}$ because it forms our interpretation of the type of commands; we use the symbol a to denote the single answer in this arena.

A ***justified string*** s in arena A is a sequence of moves of A equipped with *justification pointers* where every non-initial occurrence m in s has a pointer to an earlier occurrence n in s such that $n \vdash_A m$. We say that n ***justifies*** m. An occurrence m is ***hereditarily justified*** by an initial occurrence i iff following pointers back from m eventually leads us to i.

A ***legal play*** s is a justified string where, if $s = s_1 a b s_2$ then $\lambda_A^{\mathsf{OP}}(a) \neq \lambda_A^{\mathsf{OP}}(b)$, *i.e.* it strictly alternates between Opponent moves and Player moves. We denote by L_A the set of all legal plays in arena A. The ***current thread*** of $sa \in L_A^{\mathsf{odd}}$, written $\lceil sa \rceil$, is the subsequence of sa consisting of all moves hereditarily justified by the hereditary justifier of a.

3.2 Compound arenas

We can combine arenas A and B in two different ways. The first forms their "product".

- $M_{A\times B} = M_A + M_B$;
- $\lambda_{A\times B} = [\lambda_A, \lambda_B]$, the copairing in $\mathbf{Set}$;
- $n \vdash_{A\times B} m$ iff $n \vdash_A m$ or $n \vdash_B m$.

The second forms the "function space" from A to B.

- $M_{A\Rightarrow B} = M_A + M_B$;
- $\lambda_{A\Rightarrow B} = [\overline{\lambda}_A, \lambda_B]$;
- $n \vdash_{A\Rightarrow B} m$ iff $n \vdash_B m$ or ($n \neq m$ and $n \vdash_A m$) or ($n \vdash_B n$ and $m \vdash_A m$).

In each case, the two ***constituent*** arenas are placed side-by-side. In $A \times B$, there is no interaction between A and B. In $A \Rightarrow B$, A is subservient to B: the initial moves of A are not initial here; instead, they are enabled by the initial moves of B. This requires that the O/P roles be reversed in A: an Opponent move of A is a Player move of $A \Rightarrow B$ and vice versa. The empty arena $\mathbf{1}$ is the unit for the $\times$ constructor.

$$\frac{\langle s, N\rangle \Downarrow \langle s', \mathtt{n}\rangle \quad \langle s', M\rangle \Downarrow \langle s'', v\rangle}{\langle s, \mathtt{assign}\ M\ N\rangle \Downarrow \langle \langle s'' \mid v \mapsto n\rangle, \mathtt{skip}\rangle} \qquad \frac{\langle s, M\rangle \Downarrow \langle s', v\rangle}{\langle s, \mathtt{deref}\ M\rangle \Downarrow \langle s', \mathtt{n}\rangle}\ s'(v) = n$$

$$\frac{\langle s, M\rangle \Downarrow \langle s', \mathtt{skip}\rangle \quad \langle s', N\rangle \Downarrow \langle s'', V\rangle}{\langle s, \mathtt{seq}_T\ M\ N\rangle \Downarrow \langle s'', V\rangle} \qquad \frac{\langle \langle s \mid v \mapsto 0\rangle, M\rangle \Downarrow \langle \langle s' \mid v \mapsto n\rangle, V\rangle}{\langle s, \mathtt{new}_T\ \lambda v.\ M\rangle \Downarrow \langle s', V\rangle}$$

$$\frac{\langle s, M\rangle \Downarrow \langle s', V\rangle}{\langle s, M\ \mathtt{or}\ N\rangle \Downarrow \langle s', V\rangle} \qquad \frac{\langle s, N\rangle \Downarrow \langle s', V\rangle}{\langle s, M\ \mathtt{or}\ N\rangle \Downarrow \langle s', V\rangle} \qquad \frac{\langle s, M\rangle \Downarrow^{\mathrm{must}} \quad \langle s, N\rangle \Downarrow^{\mathrm{must}}}{\langle s, M\ \mathtt{or}\ N\rangle \Downarrow^{\mathrm{must}}}$$

$$\frac{\langle s, M\rangle \Downarrow^{\mathrm{must}} \qquad \forall \langle s', \lambda x.\ M'\rangle.\ \langle s, M\rangle \Downarrow \langle s', \lambda x.\ M'\rangle \Rightarrow \langle s', M'[N/x]\rangle \Downarrow^{\mathrm{must}}}{\langle s, MN\rangle \Downarrow^{\mathrm{must}}}$$

Figure 1. Operational semantics of EIA

3.3 Strategies

The usual definition of strategy is in terms of traces: a strategy for Player is a set of even-length legal plays saying what moves may be made by the system. As was hinted at in the introduction, if we take this definition, we end up with a model of EIA which is fully abstract with respect to may-equivalence. In order to capture M&M-equivalence, we add to the definition of strategy an additional component, its *divergences*. We shall only give details of the extended model; the model based solely on traces can be obtained (more or less) by deleting all mention of divergences and must-convergence from what follows.

A ***strategy*** σ on an arena A is a pair (T_σ, D_σ). The first component T_σ, known as the ***traces*** of σ, is a non-empty set of even-length legal plays of A satisfying

$$sab \in T_\sigma \Rightarrow s \in T_\sigma.$$

We write $\mathsf{dom}(\sigma)$ for the ***domain*** of σ, *i.e.* the set $\{sa \in L_A \mid \exists b.\, sab \in T_\sigma\}$ and $\mathsf{cc}(\sigma)$ for the ***contingency closure*** of σ, *i.e.* $T_\sigma \cup \mathsf{dom}(\sigma)$. Given $sa \in \mathsf{dom}(\sigma)$, let $\mathsf{rng}_\sigma(sa) = \{b \in M_A \mid sab \in T_\sigma\}$.

The second component D_σ is known as the ***divergences*** of σ; it's a set of odd-length legal plays of A satisfying

(d1) $(s \in T_\sigma \wedge sa \in L_A \wedge sa \notin \mathsf{dom}(\sigma)) \Rightarrow \exists d \in D_\sigma.\, d \sqsubseteq sa.$

(d2) $sa \in D_\sigma \Rightarrow s \in T_\sigma.$

(d3) $\mathsf{rng}_\sigma(sa)$ infinite $\Rightarrow \exists d \in D_\sigma.\, d \sqsubseteq sa.$

Axiom **(d1)** says that, if a strategy is confronted with a situation to which it has no response, *i.e.* $sa \notin \mathsf{dom}(\sigma)$, then this must be reflected by an appropriate divergence.

(d3) is the ***finite-branching*** condition: if, at some point, there is the possibility of infinite branching then there must be the possibility of divergence.

Representing divergence The existence of a divergence $sa \in D_\sigma$ records the fact that σ may diverge after playing the sequence sa. Of course, once a program has diverged, it remains divergent so, if $sa \in D_\sigma$, the existence of some other $satb \in D_\sigma$ is of little interest.

This can be represented in several ways. The choice made in CSP is to include all extensions of a divergence as divergences by convention; this has the consequence of forcing the traces of a process to include all possible behaviours after a divergence has been reached, running counter to the intuition of a divergent process having *no* observable behaviours. A second possibility is to record only the minimal divergences, denoted by $\mathsf{div}(\sigma)$.

Clearly, many choices of *representation* of divergent behaviour are available. Rather than fixing on any particular representation, we identify those strategies which intuitively record the same behaviour by means of an equivalence relation; see Section 3.5 below.

3.4 Composition of strategies

Let u be a finite string of moves from arenas A, B and C with "justification pointers" from all moves except those initial in C. We define $u \upharpoonright B, C$ to be the subsequence of u where we delete all moves from A and all pointers to A; $u \upharpoonright A, B$ is defined similarly. We define $u \upharpoonright A, C$ by removing all moves from B and all pointers to B with one exception: if $a \in A$ points to $b \in B$ which, in turn, points to $c \in C$ then a points to c in $u \upharpoonright A, C$.

Such a string u is a ***legal interaction*** of A, B and C iff $u \upharpoonright A, B \in L_{A \Rightarrow B}$, $u \upharpoonright B, C \in L_{B \Rightarrow C}$ *and* $u \upharpoonright A, C \in L_{A \Rightarrow C}$. The set of all legal interactions of A, B and C is written $\mathsf{int}(A, B, C)$.

Given $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$, we define $T_\sigma \parallel T_\tau$ to be the set of $u \in \mathsf{int}(A, B, C)$ such that

$$u \upharpoonright A, B \in T_\sigma \wedge u \upharpoonright B, C \in T_\tau.$$

We then set $T_{\sigma;\tau} = \{s \upharpoonright A, C \mid s \in T_\sigma \parallel T_\tau\}$, *i.e.* the usual "parallel composition plus hiding". We therefore have that $T_{(\sigma;\tau);\upsilon} = T_{\sigma;(\tau;\upsilon)}$.

A divergence in $\sigma\,;\tau$ can arise in two ways. The first is when one or other of σ and τ diverges; the other is when infinite chattering happens in B. For the first, we define $D_\sigma \lightning D_\tau$ to be the set of $u \in \mathsf{int}(A, B, C)$ such that

$$\begin{array}{l}(u \upharpoonright A, B \in T_\sigma \ \wedge\ u \upharpoonright B, C \in D_\tau) \quad \vee \\ (u \upharpoonright A, B \in D_\sigma \ \wedge\ u \upharpoonright B, C \in T_\tau).\end{array}$$

In other words, u consists of a trace of σ interacting with a divergence of τ, or vice versa.

If u_∞ is an infinite sequence of moves from A, B and C equipped with "justification pointers" then it's an ***infinite interaction*** iff for all finite prefixes $u' \sqsubseteq^{\mathsf{fin}} u_\infty$, $u' \in \mathsf{int}(A, B, C)$. We write $\mathsf{int}_\infty(A, B, C)$ for the set of all such infinite interactions. We now define $T_\sigma \lightning T_\tau$ to be the set of all $u_\infty \in \mathsf{int}_\infty(A, B, C)$ such that $u_\infty \upharpoonright A, C \in L_{A \Rightarrow C}$, *i.e.* $u_\infty \upharpoonright A, C$ is finite, and, for all $u' \sqsubseteq^{\mathsf{fin}} u_\infty$,

$$u' \upharpoonright A, B \in \mathsf{cc}(\sigma) \wedge u' \upharpoonright B, C \in \mathsf{cc}(\tau).$$

We complete the definition of $\sigma\,;\tau$ by specifying $D_{\sigma;\tau}$ to be

$$\{u \upharpoonright A, C \mid u \in D_\sigma \lightning D_\tau \vee u \in T_\sigma \lightning T_\tau\}.$$

Proposition If $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ then $\sigma\,;\tau : A \Rightarrow C$.

Proof This is largely routine. The tricky point is to check that $\sigma\,;\tau$ is finite-branching. The rough idea here is that, if $\sigma\,;\tau$ has a point of infinite branching then there are two possible reasons. Either, one or other of σ and τ has a point of infinite branching, in which case we're guaranteed a divergence since σ and τ are both finite-branching. Or, neither σ nor τ has any point of infinite branching and, by applying König's lemma, we find an infinite interaction between σ and τ giving rise to a divergence by livelock. ■

Identities & diagonals Given an arena A, we define Id_A to be $(\mathsf{id}_A, \varnothing)$ where id_A is the usual ***copycat***:

$$\{s \in L_{A_\ell \Rightarrow A_r} \mid \forall s' \sqsubseteq^{\mathsf{even}} s.\ s' \upharpoonright A_\ell = s' \upharpoonright A_r\}.$$

We also have a strategy $\Delta_A : A \Rightarrow (A \times A)$ with no divergences and with traces all $s \in L^{\mathsf{even}}_{A \Rightarrow (A_\ell \times A_r)}$ such that

$$\forall s' \sqsubseteq^{\mathsf{even}} s.\ s' \upharpoonright \ell \in \mathsf{id}_A \wedge s' \upharpoonright r \in \mathsf{id}_A$$

where $s' \upharpoonright \ell$ means the subsequence of s' consisting of all moves hereditarily justified by an initial occurrence in A_ℓ, and $s' \upharpoonright r$ is defined similarly.

3.5 Ordering strategies

Let σ and τ be strategies on some arena A; what does it mean for τ to be better than σ? Well, it should certainly be the case that every trace of σ is also a trace of τ. This can be equivalently stated as $\forall s \in T_\sigma.\ \exists t \in T_\tau.\ s \sqsubseteq t$, bringing out the similarity to the (lower half of the) Egli-Milner ordering.

For the divergences, the analogy with the upper half of the Egli-Milner ordering suggests: $\forall e \in D_\tau.\ \exists d \in D_\sigma.\ d \sqsubseteq e$. In other words, whenever τ is in a position where it may (already) have diverged, σ should also be so. But, it should also be the case that, if τ wishes to "improve" on σ, perhaps by adding more traces, then σ must already have reached a divergence. Otherwise, τ risks causing divergence where there was previously no such possibility.

This is best illustrated with a small example; consider the strategies $\mathtt{tt} = (\{\varepsilon, \mathsf{q}\mathtt{tt}\}, \varnothing)$ and $\top_{\mathbf{B}} = (\{\varepsilon, \mathsf{q}\mathtt{tt}, \mathsf{qff}\}, \varnothing)$ on $\mathbf{1} \Rightarrow \mathbf{B}$ and the "half-baked identity" on $\mathbf{B} \Rightarrow \mathbf{B}$ given by $\mathsf{hbi} = (\{\varepsilon, \mathsf{qq}, \mathsf{qq}\mathtt{tt}\mathtt{tt}\}, \{\mathsf{qqff}\})$. It's clear that, with the above sketch of an ordering, $\mathtt{tt}$ is less than $\top_{\mathbf{B}}$. However, $\mathtt{tt}\,;\mathsf{hbi} = (\{\varepsilon, \mathsf{q}\mathtt{tt}\}, \varnothing)$ and $\top_{\mathbf{B}}\,;\mathsf{hbi} = (\{\varepsilon, \mathsf{q}\mathtt{tt}\}, \{\mathsf{q}\})$ so that $\mathtt{tt}\,;\mathsf{hbi}$ is *not* less than $\top_{\mathbf{B}}\,;\mathsf{hbi}$. This means that composition isn't even monotone with respect to the proposed ordering: by adding an extra trace qff, $\top_{\mathbf{B}}$ can be "caught out" by a strategy like hbi. We must therefore modify the ordering so that that $\mathtt{tt}$ is *not* less than $\top_{\mathbf{B}}$.

We define the ***lower***, ***upper*** and ***convex*** ordering on our strategies as follows.

$$\begin{array}{lll}\sigma \leqslant^\flat_A \tau & \text{iff} & T_\sigma \subseteq T_\tau \\ \sigma \leqslant^\sharp_A \tau & \text{iff} & (\forall e \in D_\tau.\ \exists d \in D_\sigma.\ d \sqsubseteq e) \wedge \\ & & (sab \in T_\tau \wedge sab \notin T_\sigma) \Rightarrow \\ & & \exists d \in D_\sigma.\ d \sqsubseteq sab \\ \sigma \leqslant^\natural_A \tau & \text{iff} & \sigma \leqslant^\flat_A \tau \wedge \sigma \leqslant^\sharp_A \tau.\end{array}$$

This ordering is highly reminiscent of Roscoe's alternative order for the failures model of CSP [22] where, as here, the crucial idea is that the convergent behaviour of σ is *exactly* copied by τ. The key condition is the second conjunct of the upper ordering; this is what guarantees $\mathtt{tt} \not\leqslant^\natural_{\mathbf{B}} \top_{\mathbf{B}}$.

The relation $\leqslant^\natural_A$ is a preorder on the set of strategies for A. We set $\sigma =^\natural_A \tau$ iff $\sigma \leqslant^\natural_A \tau$ and $\tau \leqslant^\natural \sigma$. For standard reasons, $\leqslant^\natural_A$ lifts to be a partial order on the set, written $\mathsf{Str}(A)$, of strategies for A quotiented by $=^\natural_A$. One useful property of $=^\natural$ is that $\sigma =^\natural \tau$ if, and only if, $T_\sigma = T_\tau$ and $\mathsf{div}(\sigma) = \mathsf{div}(\tau)$, *i.e.* the divergent behaviour of a strategy is entirely determined by its minimal divergences. So, for example, $(\mathsf{id}_{\mathbf{B}}, \{\mathsf{q}\}) =^\natural (\mathsf{id}_{\mathbf{B}}, \{\mathsf{q}, \mathsf{qq}\mathtt{tt}, \mathsf{qqff}\})$. In a sense then, we can see $\mathsf{div}(\sigma)$ as mapping out the "boundary" between the reliable and the unreliable part of σ.

Proposition If $\sigma, \sigma' : A \Rightarrow B$ and $\tau, \tau' : B \Rightarrow C$ satisfy $\sigma \leqslant^\natural \sigma'$ and $\tau \leqslant^\natural \tau'$ then $\sigma\,;\tau \leqslant^\natural \sigma'\,;\tau'$.

Proof It suffices to show monotonicity in each argument separately; consider the proof that $\sigma ; \tau \leqslant^\natural \sigma' ; \tau$. The argument breaks down into a multitude of cases. The most interesting case is where $d \in D_{\sigma';\tau}$ is witnessed by $u \in D_{\sigma'} \not\downarrow D_\tau$ such that $u \upharpoonright A, B \in T_{\sigma'}$, $u \upharpoonright B, C \in D_\tau$ and $u \upharpoonright A, B \notin T_\sigma$. In this case, applying $\sigma \leqslant^\natural \sigma'$, we can find a prefix $sa \sqsubseteq^{\mathsf{odd}} u \upharpoonright A, B$ such that $sa \in D_\sigma$ so that truncating u after the occurrence a witnesses some prefix of d in $D_{\sigma;\tau}$. This is exactly where we catch the above-mentioned problem. ■

The basic category $\mathcal{G}$ We write $\langle\sigma\rangle$ for the $=^\natural$-equivalence class containing σ. We can now lift the definition of composition to such equivalence classes:

$$\langle\sigma\rangle ; \langle\tau\rangle =_{\mathrm{df}} \langle\sigma ; \tau\rangle.$$

We have a category where objects are arenas and an arrow $f : A \to B$ is a $=^\natural$-equivalence class of strategies on $A \Rightarrow B$. Associativity of composition follows from:

Proposition If $\sigma : A \Rightarrow B$, $\tau : B \Rightarrow C$ and $\upsilon : C \Rightarrow D$ then $(\sigma ; \tau) ; \upsilon =^\natural \sigma ; (\tau ; \upsilon)$.

Proof The interesting case is where a divergence in $(\sigma ; \tau) ; \upsilon$ is caused by an infinite chatter in C. We therefore have an infinite, increasing sequence of traces of $\sigma ; \tau$ conspiring with an infinite, increasing sequence of traces of υ, producing a divergence by livelock. Now, if this sequence of traces of $\sigma ; \tau$ is caused by an infinite, increasing sequence of *interactions* of σ and τ then we're okay. But this isn't necessarily the case: it's possible to have an infinite number of finite interactions witnessing the increasing sequence of traces but where the interactions do not themselves form an increasing sequence. In this case, König's lemma guarantees a point where one or other of σ and τ branches infinitely and an appeal to the finite-branching axiom completes the proof. ■

$\mathcal{G}$ as an SMCC The $\times$ and $\Rightarrow$ arena constructors lift to be bifunctors on this category. Furthermore, $\times$ provides a symmetric monoidal structure and, for any $\sigma : (A \times B) \Rightarrow C$, we can relabel the moves yielding $\Lambda(\sigma) : A \Rightarrow (B \Rightarrow C)$. This (natural) isomorphism gives us a symmetric monoidal closed category. Note also that the diagonal maps Δ_A give every object the structure of a comonoid.

3.6 A Cartesian closed category

We now consider the important subclass of *single-threaded* strategies (called *thread-independent* in [2]). First of all, we need some new notation: if $sab, ta \in L_A$ such that $\lceil sa \rceil = \lceil ta \rceil$ and the justifier of b in sab occurs in $\lceil sa \rceil$ then we write $\mathsf{Match}(sab, ta)$ for the (unique) $tab \in L_A$ such that $\lceil sa \rceil \cdot b = \lceil ta \rceil \cdot b$, *i.e.* the justifier of b is the same in $\mathsf{Match}(sab, ta)$ as it is in sab.

A strategy σ on arena A is ***single-threaded*** iff

- for all $sab \in T_\sigma$, the justifier of b occurs in $\lceil sa \rceil$.
- if $sab, t \in T_\sigma$, $ta \in L_A$ and $\lceil sa \rceil = \lceil ta \rceil$ then $\mathsf{Match}(sab, ta) \in T_\sigma$.
- if $sa \in D_\sigma$ then $\exists tb \sqsubseteq sa.\ \lceil tb \rceil \in D_\sigma$.
- if $\lceil sa \rceil \in D_\sigma$ then $\exists tb \sqsubseteq sa.\ tb \in D_\sigma$.

In other words, the response of such a strategy to some $sa \in L_A^{\mathsf{odd}}$ depends only on $\lceil sa \rceil$. The last two conditions extend this idea to divergences: what we want is for any divergence of σ to have been caused by the play in a single thread, *i.e.* if $sa \in D_\sigma$ then some prefix $tb \sqsubseteq sa$ satisfies $\lceil tb \rceil \in D_\sigma$ and, if we've reached a point $s \in T_\sigma$ and Opponent now plays the move a then, if $\lceil sa \rceil \in D_\sigma$ then this should make sa, or some prefix thereof, a divergence too.

Proposition A strategy $\sigma : A \Rightarrow B$ is single-threaded if, and only if, it's a comonoid homomorphism; *i.e.* $\sigma ; \Delta_B = \Delta_A ; (\sigma \times \sigma)$.

Proof The (long) proof proceeds via a detailed analysis of the kinds of interactions possible in $\sigma ; \Delta_B$ and in $\Delta_A ; (\sigma \times \sigma)$. For more details, see the first author's forthcoming thesis [10]. ■

It follows from this (and the bifunctoriality of $\times$) that the single-threaded strategies form a subcategory $\mathcal{C}$. Furthermore, the strategies Δ_A are single-threaded, so form a natural transformation on this subcategory. It follows for general reasons that $\times$ is a categorical product on $\mathcal{C}$, thanks to the validity of certain other equations [9]. Since the closed structure also restricts to $\mathcal{C}$, we conclude that $\mathcal{C}$ is a Cartesian closed category.

We've already noted that $\mathsf{Str}(A)$ is partially ordered by $\leqslant^\natural_A$ and that composition is monotone with respect to $\leqslant^\natural$. These results can be strengthened to show that all homsets are algebraic CPOs; a strategy σ is compact if, and only if, the set $\{sab \in T_\sigma \mid sab = \lceil sa \rceil \cdot b\}$ is finite. Moreover, composition is continuous with respect to $\leqslant^\natural$. In other words, $\mathcal{C}$ is CPO-enriched.

Constraining $\mathcal{C}$ There are several further constraints that may be placed on the strategies in $\mathcal{C}$. We'll be interested in three such conditions, two of which rely on the following definition of the ***Player view*** of a non-empty legal play.

$$\begin{aligned}
\ulcorner sm \urcorner &= m, && \text{if } m \text{ is initial;}\\
\ulcorner sntm \urcorner &= \ulcorner s \urcorner nm, && \text{if } m \text{ is an O-move and}\\
& && \quad n \text{ justifies } m;\\
\ulcorner sm \urcorner &= \ulcorner s \urcorner m, && \text{if } m \text{ is a P-move.}
\end{aligned}$$

A strategy σ satisfies ***Player visibility*** iff for all $sab \in T_\sigma$, the justifier of b occurs in $\ulcorner sa \urcorner$.

We say that σ satisfies ***Player bracketing*** iff for all $sab \in T_\sigma$ such that $\lambda_A^{\mathsf{QA}}(b) = \mathsf{A}$, b answers (*i.e.* is justified by) the *pending question* of $\ulcorner sa \urcorner$, *i.e.* the most recently asked but, as yet, unanswered Question in $\ulcorner sa \urcorner$.

Both of these conditions can be shown to be preserved by composition and, from now on, we'll only consider strategies satisfying both of them. That is to say, we work in the sub-CCC $\mathcal{C}_{vb}$ of $\mathcal{C}$ consisting of strategies satisfying the above constraints.

The third constraint of interest is ***determinism***: σ is deterministic iff, for all $sab, sac \in T_\sigma$, $sab = sac$ and if $sa \in D_\sigma$ then $sa \notin \mathsf{dom}(\sigma)$. Clearly, in the subcategory of deterministic strategies, the divergences play no role, so this category is equivalent to the one that harbours a fully abstract model of (deterministic) IA [5].

3.7 Deterministic factorization

Our full abstraction result for EIA hinges on a definability theorem stating that every compact strategy is the denotation of a term of EIA. We obtain this result by means of a factorization: we show that any strategy can be considered as the composition of a generic nondeterministic "oracle" with a deterministic strategy. Since we already know that all compact, deterministic strategies are definable in IA, this yields the desired result.

The factorization of a strategy σ on an arena A proceeds via an encoding of Player's moves (in A) as numbers: at each point $sa \in \mathsf{dom}(\sigma)$, the factorized strategy $\mathsf{Det}(\sigma)$ "consults the oracle" which (possibly) produces a number which the strategy interprets as a Player move—and then plays that move if appropriate. One important point to note is that, since we're factorizing from an arbitrary strategy to a deterministic one, the oracle strategy must be capable of causing divergence but also, if necessary, must be able to avoid doing this.

The following bit of useful notation and terminology helps make this more precise: if $sa \in \mathsf{dom}(\sigma)$ then we say that it's a ***reliable point*** of σ, written $sa \in \mathsf{rp}(\sigma)$, iff for all $d \sqsubseteq^{\mathsf{odd}} sa$, $d \notin D_\sigma$. So, the factorization must respect the reliable points of σ while ensuring that its unreliable points remain that way.

Proposition If σ is a strategy on arena A then there exists a *deterministic* strategy $\mathsf{Det}(\sigma)$ on $(\mathbf{N} \Rightarrow \mathbf{N}) \Rightarrow A$ and a strategy oracle for $\mathbf{1} \Rightarrow (\mathbf{N} \Rightarrow \mathbf{N})$ such that $\sigma =^\natural \mathsf{oracle}\,;\, \mathsf{Det}(\sigma)$. Furthermore, if σ is compact then $\mathsf{Det}(\sigma)$ can be taken compact.

Proof Let $\mathsf{code}_A(-)$ be an injective function from the Player moves of A to $\mathbf{N}$. Suppose $sa \in \mathsf{dom}(\sigma)$. There are two cases, depending on whether $sa \in \mathsf{rp}(\sigma)$ or not.

If so, by the finite-branching condition, the range of σ at sa must be finite, *i.e.* $\mathsf{rng}_\sigma(sa) = \{b_1, \ldots, b_n\}$. Let m be the maximum of $\{\mathsf{code}_A(b_1), \ldots, \mathsf{code}_A(b_n)\}$ and assume that $\mathsf{code}_A(b_1) < \mathsf{code}_A(b_2) < \ldots < \mathsf{code}_A(b_n)$. $\mathsf{Det}(\sigma)$ proceeds as follows, taking care to avoid introducing any possibility of divergence.

$$
\begin{array}{ccccc}
(\mathbf{N} & \Rightarrow & \mathbf{N}) & \Rightarrow & A \\
 & & \vdots & & \\
 & & & & a \\
 & & \mathsf{q} & & \\
\mathsf{q} & & & & \\
\mathsf{m} & & & & \\
 & & \mathsf{j} & & \\
 & & & & b_i
\end{array}
$$

where, if $i = 1$ then $0 \leq j \leq \mathsf{code}_A(b_1)$ or, for $1 < i \leq m$, $\mathsf{code}_A(b_{i-1}) < j \leq \mathsf{code}_A(b_i)$.

In other words, $\mathsf{Det}(\sigma)$ "consults the oracle" by playing q. The oracle (to be defined below) responds by asking "how much nondeterminism do you want to factorize?" to which $\mathsf{Det}(\sigma)$ responds with m. The oracle will then *reliably* provide some number between 0 and m. $\mathsf{Det}(\sigma)$ divides up the numbers between 0 and m by: if we get j such that $0 \leq j \leq \mathsf{code}_A(b_1)$, we play b_1; if $\mathsf{code}_A(b_1) < j \leq \mathsf{code}_A(b_2)$, we play b_2, etc.

The other case is if $sa \notin \mathsf{rp}(\sigma)$. In this case, we don't need to take any care to avoid undesired divergences as σ is, by now, already unreliable. By convention, we supply the oracle with input 0. The oracle's response to this will be to supply any natural number but with the (necessary—by finite-branching) possibility of divergence. The strategy continues in the following fashion.

$$
\begin{array}{ccccc}
(\mathbf{N} & \Rightarrow & \mathbf{N}) & \Rightarrow & A \\
 & & \vdots & & \\
 & & & & a \\
 & & \mathsf{q} & & \\
\mathsf{q} & & & & \\
\mathsf{0} & & & & \\
 & \mathsf{code}(b_i) & & & \\
 & & & & b_i
\end{array}
$$

So, not only the choice of move, but also any possibility of divergence has been delegated to the oracle. It's exactly this that allows $\mathsf{Det}(\sigma)$ to simulate σ while being deterministic.

To complete the proof, we must now describe the oracle strategy. Its trace set is the smallest set containing

$$\{\varepsilon, \mathsf{qq}\} \cup \{\mathsf{qq0}n \mid n \in \mathbf{N}\} \cup \bigcup_{i \in \mathbf{N} - \{0\}} \{\mathsf{qq}i n \mid n \leq i\}$$

and satisfying the axioms for single-threadedness. If $sa \in L_{\mathbf{N} \Rightarrow \mathbf{N}}$ such that s is a trace and $\lceil sa \rceil = s' \mathsf{qq0}$ for some s' then sa is a divergence of the oracle.

It's clear that $\mathsf{Det}(\sigma)$ is indeed a deterministic strategy and that if σ is compact then so is $\mathsf{Det}(\sigma)$. It's also easy to check that $\mathsf{oracle} \,;\, \mathsf{Det}(\sigma) =^{\natural} \sigma$: the essential point is that the cautious case of the factorization, where $sa \in \mathsf{rp}(\sigma)$, ensures that no divergence can arise in $\mathsf{oracle} \,;\, \mathsf{Det}(\sigma)$ until σ is unreliable, *i.e.* σ and $\mathsf{oracle} \,;\, \mathsf{Det}(\sigma)$ share the same set of minimal divergences. ■

4 Full abstraction

4.1 The model

The interpretation of the constructs of IA is defined in the same way as in [5]. We shall just give the semantics of the novel operator `or`. Recall that the base types are modelled as flat arenas: $[\![\mathtt{Com}]\!] = \mathbf{C}$ and $[\![\mathtt{Nat}]\!] = \mathbf{N}$.

Given $\Gamma \vdash M : \mathtt{Nat}$ and $\Gamma \vdash N : \mathtt{Nat}$, we define:

$$[\![M \,\mathtt{or}\, N]\!] = \langle [\![M]\!], [\![N]\!] \rangle \,;\, \mathsf{choice}$$

where the two typical plays of choice are:

$$\begin{array}{ccccc} (\mathbf{N} & \times & \mathbf{N}) & \Rightarrow & \mathbf{N} \\ & & & & \mathsf{q} \\ \mathsf{q} & & & & \\ \mathsf{n} & & & & \\ & & & & \mathsf{n} \\ \hline & & & & \mathsf{q} \\ & & \mathsf{q} & & \\ & & \mathsf{m} & & \\ & & & & \mathsf{m}. \end{array}$$

4.2 Adequacy & soundness

The following sequence of results shows that our model is sound with respect to M&M-equivalence. Let M and N be closed terms of type `Com`.

Lemma If $M \Downarrow \mathtt{skip}$ then $\mathsf{qa} \in T_{[\![M]\!]}$. If $M \Downarrow^{\mathrm{must}}$ then $D_{[\![M]\!]} = \varnothing$.

This *consistency* result shows that our model respects both may and must convergence. It is proved by a straightforward induction. Its converse, *adequacy*, also holds.

Proposition If $\mathsf{qa} \in T_{[\![M]\!]}$ then $M \Downarrow \mathtt{skip}$. If $D_{[\![M]\!]} = \varnothing$ then $M \Downarrow^{\mathrm{must}}$.

As usual for such results, the proof is involved and makes use of a computability predicate. Our proof bears a strong resemblance to that of Hennessy & Ashcroft [11], augmented with a treatment of state as in [5].

Putting these results together, we obtain *soundness* in the usual way:

Theorem If M and N are closed terms of type T such that $[\![M]\!] = [\![N]\!]$ then $M \simeq_{m\&m} N$.

4.3 Definability

We now wish to show that every compact element of our model is the interpretation of some term of EIA. Thanks to the factorization theorem proved above, it suffices to give a term defining the strategy oracle. The term $\mho : \mathtt{Nat}$, defined to be

$$\mathbf{Y}_{\mathtt{Nat}}\,(\lambda x.\, \mathtt{0 \ or \ (succ}\ x))$$

may converge to any numeral and may also diverge. The term $\mho_n : \mathtt{Nat} \to \mathtt{Nat}$, defined as

$$\mathbf{Y}_{\mathtt{Nat}\to\mathtt{Nat}}\,(\lambda f.\, \lambda x.\, \mathtt{if0}\ x\ \mathtt{0}\ (\mathtt{0\,or\,(succ}\, f(\mathtt{pred}\, x))))$$

consumes a numeral `n` and must converge, producing any numeral between `0` and `n`. Putting these together, we now define the oracle strategy by

$$\lambda x.\, \mathtt{new}\ v := x\ \mathtt{in}\,(\mathtt{if0}\ !v\ \mho\ (\mho_n(!v)))$$

using a little standard syntactic sugar. As an immediate consequence, we have:

Theorem If A and B are arenas interpreting types of EIA and $\sigma : A \Rightarrow B$ is a compact arrow in $\mathcal{C}_{vb}$, then $\sigma = [\![M]\!]$ for some term M of EIA.

Remark Since our term defining the oracle strategy only makes use of nondeterministic choice between terms of type `Nat`, we deduce from the above factorization theorem that the restriction of the nondeterministic `or` operation in this way entails no loss of generality: nondeterminism can be "pushed down" to base types, *e.g.* if M, N have type $T_1 \to \ldots \to T_n \to \mathtt{Nat}$, we can define $M \,\mathtt{or}\, N$ by

$$\lambda \vec{x}.\, (M\vec{x})\,\mathtt{or}\,(N\vec{x}).$$

4.4 Full Abstraction

As usual, the passage from definability to full abstraction makes use of a certain quotient of the model, the *intrinsic quotient*. We define an equivalence relation on each homset of $\mathcal{C}_{vb}$ as follows.

Given $f, g : A \Rightarrow B$, write $f \sim g$ iff for all "test morphisms" $\alpha : [A \Rightarrow B] \Rightarrow \mathbf{C}$,

$$`f' \,;\, \alpha = `g' \,;\, \alpha$$

where $`f' : \mathbf{1} \Rightarrow [A \Rightarrow B]$ is the *name* of f, defined by currying in the obvious way. The quotient of $\mathcal{C}_{vb}$ by this equivalence relation is again a Cartesian closed category, and can readily be seen to give rise to a model of EIA which inherits the soundness property of the model in $\mathcal{C}_{vb}$. Moreover, the definability theorem for $\mathcal{C}_{vb}$ gives rise to a *completeness* result for the quotiented model.

Theorem (full abstraction) Suppose M and N are closed terms of type T. If $M \simeq_{m\&m} N$ then $[\![M]\!] \sim [\![N]\!]$.

Proof We prove the contrapositive. Suppose $[\![M]\!] \not\sim [\![N]\!]$, so that for some α, $[\![M]\!] ; \alpha \neq [\![N]\!] ; \alpha$. This α can clearly be taken to be compact, and by definability, $\alpha = [\![x : T \vdash P : \mathtt{Com}]\!]$. By consistency and adequacy, it follows that the context P distinguishes M from N. ■

References

[1] S. Abramsky, R. Jagadeesan, and P. Malacaria. Full abstraction for PCF. To appear in Information and Computation, 199?

[2] S. Abramsky, K. Honda, and G. McCusker. A fully abstract game semantics for general references. In *Proceedings, Thirteenth Annual IEEE Symposium on Logic in Computer Science*, 1998.

[3] S. Abramsky and G. McCusker. Full abstraction for Idealized Algol with passive expressions. To appear in Theoretical Computer Science, 199?

[4] S. Abramsky and G. McCusker. Games and full abstraction for the lazy λ-calculus. In *Proceedings, Tenth Annual IEEE Symposium on Logic in Computer Science*, pages 234–243. IEEE Computer Society Press, 1995.

[5] S. Abramsky and G. McCusker. Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions. In P. W. O'Hearn and R. D. Tennent, editors, *Algol-like Languages*, pages 297–329 of volume 2. Birkhaüser, 1997.

[6] S. Abramsky and G. McCusker. Game semantics. Lecture notes to accompany Samson Abramsky's lectures at the 1997 Marktoberdorf summer school. To appear, 1998.

[7] K. Apt and G. Plotkin. Countable nondeterminism and random assignment. *Journal of the ACM*, 33(4):724–767, October 1986.

[8] P. Baillot, V. Danos, T. Ehrhard, and L. Régnier. Believe it or not, AJM's games model is a model of classical linear logic. In *Proceedings, Twelth Annual IEEE Symposium on Logic in Computer Science* [14].

[9] P. J. Freyd, P. W. O'Hearn, A. J. Power, M. Takeyama, and R. D. Tennent. Bireflectivity. In *Mathematical Foundations of Programming Semantics, Eleventh Annual Conference, Tulane University, New Orleans, LA, March 29 - April 1, 1995*. Elsevier, 1995. Electronic Notes in Theoretical Computer Science 1.

[10] R. S. Harmer. *Games and full abstraction for nondeterministic languages*. PhD thesis, University of London, 1999. In preparation.

[11] M. Hennessy and E. Ashcroft. A mathematical semantics for a nondeterministic typed λ-calculus. *Theoretical Computer Science*, 11:227–245, 1980.

[12] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[13] J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II and III. To appear in Information and Computation, 199?

[14] IEEE Computer Society Press. *Proceedings, Twelth Annual IEEE Symposium on Logic in Computer Science*, 1997.

[15] J. Laird. Full abstraction for functional languages with control. In *Proceedings, Twelth Annual IEEE Symposium on Logic in Computer Science* [14].

[16] G. McCusker. *Games and Full Abstraction for a Functional Metalanguage with Recursive Types*. PhD thesis, Department of Computing, Imperial College, University of London, 1996.

[17] G. McCusker. Games and full abstraction for FPC. In *Proceedings, Eleventh Annual IEEE Symposium on Logic in Computer Science*, pages 174–183. IEEE Computer Society Press, 1996.

[18] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[19] H. Nickau. Hereditarily sequential functionals. In *Proceedings of the Symposium on Logical Foundations of Computer Science: Logic at St. Petersburg*, Lecture notes in Computer Science. Springer, 1994.

[20] G. Plotkin. A powerdomain construction. *SIAM Journal on Computing*, 5:452–487, 1976.

[21] J. C. Reynolds. The essence of Algol. In *Proceedings of the 1981 International Symposium on Algorithmic Languages*, pages 345–372. North-Holland, 1981.

[22] A. W. Roscoe. An alternative order for the failures model. In 'Two papers on CSP', Technical monograph PRG-67, Oxford University Computing Laboratory, 1989.

[23] A. W. Roscoe. Unbounded non-determinism in CSP. *Journal of Logic and Computation*, 3(2):131–172, April 1993.

[24] M. B. Smyth. Powerdomains. *Journal of Computer and Systems Sciences*, 16(1):23–36, February 1978.