FULL ABSTRACTION FOR A SIMPLE PARALLEL PROGRAMMING LANGUAGE

M.C.B. Hennessy
G.D. Plotkin
Department of Artificial Intelligence
University of Edinburgh
Hope Park Square Meadow Lane
EDINBURGH EH8 9NW Scotland

INTRODUCTION

In [Plo1] a powerdomain was defined which was intended as a kind of analogue of the powerset construction, but for (certain kinds) of cpos. For example the powerdomain $\mathcal{P}(S_\perp)$ of the flat cpo $S_\perp$, formed from a set S, is the set $\{X \subseteq S_\perp | (X \neq \emptyset)$ and $((\perp \in X)$ or X is finite$)\}$ with the Egli-Milner ordering :

$$X \sqsubseteq_{E-M} Y \equiv (\forall x \in X. \exists y \in Y.\ x \sqsubseteq y) \wedge (\forall y \in Y. \exists x \in X.\ x \sqsubseteq y).$$

This enabled nondeterminism to be modelled by an analogue of set-theoretic union and a denotational semantics for a simple language with parallelism was given, treating parallelism in terms of non-deterministic mergeing of uninterruptible actions. Expected identities such as the associativity and commutativity of the parallel combinator were true in this semantics.

Unfortunately, other reasonable identities do not hold, in particular the distributivities :

$$P\ ;\ (Q\ \underline{or}\ R) \equiv (P\ ;Q)\ \underline{or}\ (P\ ;\ R)$$

$$(Q\ \underline{or}\ R)\ ;\ P \equiv (Q\ ;\ P)\ \underline{or}\ (R\ ;\ P)$$

and so, with a suitable definition of behavior, the semantics will not be fully abstract [Mil 1] , [Plo 2]. Analysis of the problem leads us to the desire for a variant of the product of two powerdomains and a definition of union, $\cup$, on the new structure and a pairing function, $\otimes$, so that :

$$x \otimes (y \cup z) = (x \otimes y) \cup (x \otimes z)$$
$$(y \cup z) \otimes x = (y \otimes x) \cup (z \otimes x).$$

The ordinary product and pointwise union will not do as then the stronger equation :

$$(x \cup x') \otimes (y \cup y') = (x \otimes y) \cup (x' \otimes y')$$

holds and the corresponding equivalence for programs should be false.

In the present paper we further develop the idea of non-deterministic domains [Egl] [Hen] which are cpos with an associative, commutative, absorptive continuous binary function (called union). Their connection to cpos gives a definition of the powerdomain for all cpos, extending [Plo1][Smy1]; there is a tensor product which satisfies the above desire ; we can give a semantics using non-deterministic domains for a simple parallel programming language like that in [Plo1] which in at least one sense, is fully abstract. Interestingly most of the manipulation of sets explicit in [Plo1] disappears here as it is "built in" to the domains and their constructions.

2. THE PROGRAMMING LANGUAGE

Syntax Our language has three sets of syntactic items.

1. BExp - a given set of Boolean expressions, ranged over by the metavariable b.
2. Act - a given set of primitive actions, ranged over by a.
3. Stat - a set of statements, ranged over by s, and given by the grammar :

s::= a|(s;s)|(<u>if</u> b <u>then</u> s <u>else</u> s)|(<u>while</u> b <u>do</u> s)|(s <u>or</u> s)|(s <u>par</u> s)|(s <u>co</u> s).

It is not necessary here to assume anything about the structure of BExp or Act ; standard examples of elements would be "$x \geq y$" or "$x:=y+5$" for an arithmetic language. The statements provide a simple imperative language with parallelism, which will be treated in terms of interleaving of atomic actions, and with a somewhat strange " coroutine" facility which gives a very strict interleaving of the atomic actions.

Operational Semantics

We will use the set $T = \{tt, ff\}$ of truthvalues and a given set, S, of states, ranged over by σ. The behaviours of the Boolean expressions and the primitive actions are given, rather abstractly, by two functions :

1. $\mathcal{E} : BExp \to (S \to T)$,
2. $\mathcal{A} : Act \to (S \to T)$.

For the statements we axiomatise a relation $\to$: Str×Str where $Str =_{def} S \cup (Stat \times S)$; the relation $\langle s,\sigma\rangle \to \sigma'$ ($\langle s,\sigma\rangle \to \langle s',\sigma'\rangle$) is to mean that executing the first uninterruptible step of s, starting from σ, results in σ' with the termination of s (respectively, with s' being the remainder of s) ; no other relations are possible :

I1. $\langle a,\sigma\rangle \to \mathcal{A}[\![a]\!](\sigma)$.

II1. $\dfrac{\langle s_1,\sigma\rangle \to \sigma'}{\langle (s_1;s_2),\sigma\rangle \to \langle s_2,\sigma'\rangle}$, 2. $\dfrac{\langle s_1,\sigma\rangle \to \langle s_1',\sigma'\rangle}{\langle (s_1;s_2),\sigma\rangle \to \langle (s_1';s_2),\sigma'\rangle}$.

III.1 $\dfrac{\langle s_1,\sigma\rangle \to str}{\langle (\underline{if}\ b\ \underline{then}\ s_1\ \underline{else}\ s_2),\sigma\rangle \to str}$ $(\mathcal{E}[\![b]\!](\sigma) = tt\ ;\ str \in Str)$,

2. $\dfrac{\langle s_2,\sigma\rangle \to str}{\langle(\underline{if}\ b\ \underline{then}\ s_1\ \underline{else}\ s_2),\sigma\rangle \to str}$ $(\mathcal{E}[\![b]\!](\sigma) = \mathrm{ff}\ ;\ str \in Str)$.

IV1. $\dfrac{\langle s,\sigma\rangle \to \sigma'}{\langle(\underline{while}\ b\ \underline{do}\ s),\sigma\rangle \to \langle(\underline{while}\ b\ \underline{do}\ s),\sigma'\rangle}$ $(\mathcal{E}[\![b]\!](\sigma) = \mathrm{tt})$,

2. $\dfrac{\langle s,\sigma\rangle \to \langle s',\sigma'\rangle}{\langle(\underline{while}\ b\ \underline{do}\ s),\sigma\rangle \to \langle(s';(\underline{while}\ b\ \underline{do}\ s)),\sigma'\rangle}$ $(\mathcal{E}[\![b]\!](\sigma) = \mathrm{tt})$,

3. $\langle(\underline{while}\ b\ \underline{do}\ s),\sigma\rangle \to \sigma$ $(\mathcal{E}[\![b]\!](\sigma) = \mathrm{ff})$.

V1. $\dfrac{\langle s_i,\sigma\rangle \to str}{\langle(s_1\ \underline{or}\ s_2),\sigma\rangle \to str}$ $(i = 1, 2\ ;\ str \in Str)$.

VI1. $\dfrac{\langle s_1,\sigma\rangle \to \sigma'}{\langle(s_1 \underline{par}\ s_2),\sigma\rangle \to \langle s_2,\sigma'\rangle}$, $\dfrac{\langle s_2,\sigma\rangle \to \sigma'}{\langle(s_1\ \underline{par}\ s_2),\sigma\rangle \to \langle s_1,\sigma'\rangle}$,

2. $\dfrac{\langle s_1,\sigma\rangle \to \langle s_1',\sigma'\rangle}{\langle(s_1\ \underline{par}\ s_2),\sigma\rangle \to \langle(s_1'\ \underline{par}\ s_2),\sigma'\rangle}$, $\dfrac{\langle s_2,\sigma\rangle \to \langle s_2',\sigma'\rangle}{\langle(s_1\ \underline{par}\ s_2),\sigma\rangle \to \langle(s_1\ \underline{par}\ s_2'),\sigma'\rangle}$.

VII1. $\dfrac{\langle s_1,\sigma\rangle \to \sigma'}{\langle(s_1 \underline{co}\ s_2),\sigma\rangle \to \sigma'}$, 2. $\dfrac{\langle s_1,\sigma\rangle \to \langle s_1',\sigma'\rangle}{\langle(s_1\ \underline{co}\ s_2),\sigma\rangle \to \langle(s_2\ \underline{co}\ s_1'),\sigma'\rangle}$

Now we can give a definition of the behavior of a statement in terms of a non-deterministic state transformation function :

3. $\mathcal{B} : Stat \to (S \to \mathcal{P}(S_\perp))$,

where : $\mathcal{B}[\![s]\!](\sigma) = \{\sigma' \in S \mid \langle s,\sigma\rangle \overset{*}{\to} \sigma'\} \cup$
$\{\perp \mid$ there is an infinite sequence $\langle s,\sigma\rangle \to .. \to \langle s_n,\sigma_n\rangle \to ..\}$.

As $\{str \mid \langle s,\sigma\rangle \to str\}$ is always finite and nonempty, Königs lemma shows $\mathcal{B}[\![s]\!](\sigma)$ is always finite or contains $\perp$ and so is in $\mathcal{P}(S_\perp)$ as required.

Note the flexibility of the method for specifying interruption points ; if we had wished conditionals to be interruptable after the test, instead of the present "test and set" capability, we would have written :

III'1. $\langle(\underline{if}\ b\ \underline{then}\ s_1\ \underline{else}\ s_2),\sigma\rangle \to \langle s_1,\sigma\rangle$ $(\mathcal{E}[\![b]\!](\sigma) = \mathrm{tt})$,

2. $\langle(\underline{if}\ b\ \underline{then}\ s_1\ \underline{else}\ s_2),\sigma\rangle \to \langle s_2,\sigma\rangle$ $(\mathcal{E}[\![b]\!](\sigma) = \mathrm{ff})$.

3. NON-DETERMINISTIC DOMAINS

We discuss the extra structure provided by the union function, the connections with powerdomains and useful constructions such as the tensor product.

Definition 3.1 : A complete partial order (cpo) is a partial order, $\langle D,\sqsubseteq\rangle$, with a least element, $\perp_D$, and lubs, $\bigsqcup_D x_n$, of increasing ω-chains ; a function $f: D \to E$ of partial orders is strict, monotonic or continuous according, respectively, as it preserves the least element, the order or the order and all existing lubs of increasing

ω-chains. We let $\underline{\underline{CPO}}$ be the category of cpo's and continuous functions, let $\underline{\underline{CPO}}_\perp$ be the subcategory of strict functions, and let $\underline{\underline{O}}$ be the category of partial orders with lubs of all increasing ω-chains and continuous functions.

The reason for considering the three categories, $\underline{\underline{CPO}}_\perp \subseteq \underline{\underline{CPO}} \subseteq \underline{\underline{O}}$ is that the main one of interest, $\underline{\underline{CPO}}$, lies between the more natural $\underline{\underline{CPO}}_\perp$ and $\underline{\underline{O}}$. All three have all small products given by Cartesian product ; $\underline{\underline{CPO}}_\perp$ and $\underline{\underline{O}}$ are small complete.

Definition 3.2 : A non-deterministic partial order (nd-po) is a structure $\langle D, \sqsubseteq, \cup \rangle$ where $\langle D, \sqsubseteq \rangle$ is a po and $\cup : D^2 \to D$ is a monotonic function (called union) where :

1. Associativity For all x, y, z in D, $(x \cup y) \cup z = x \cup (y \cup z)$.
2. Commutativity For all x,y in D, $(x \cup y) = (y \cup x)$.
3. Absorption For all x in D, $(x \cup x) = x$.

A function $f: D \to E$ of nd-po's is linear if it preserves union. We let $\underline{\underline{ND}}$ be the category of non-deterministic domains (nd-pos which are cpo's and have a continuous union) and continuous linear functions, let $\underline{\underline{ND}}_\perp$ be the subcategory of strict functions and let $\underline{\underline{NO}}$ be the category of nd-po's which are $\underline{\underline{O}}$-objects and which have a continous union and continuous linear functions.

Again, $\underline{\underline{ND}}_\perp \subseteq \underline{\underline{ND}} \subseteq \underline{\underline{NO}}$ and all three have all small products given by Cartesian product and $\underline{\underline{ND}}_\perp$ and $\underline{\underline{NO}}$ are small complete. Note that in any nd-po, D, we can define a "subset relation" by : $x \subseteq y$ iff $(x \cup y) = y$; this is a partial order and if D is $\underline{\underline{NO}}$ then $\subseteq$ is inductive in the sense that if $\langle x_n \rangle, \langle y_n \rangle$ are two increasing ω-chains with $x_n \subseteq y_n$ then $(\bigsqcup x_n) \subseteq (\bigsqcup y_n)$.

For constructions on $\underline{\underline{ND}}_\perp$ we use the Freyd Adjoint Functor Theorem (FAFT-see [Mac]) in conjunction with a useful lemma.

Definition 3.3 : An $\underline{\underline{O}}$ -category is one whose hom sets are equipped with a partial order so that they form an $\underline{\underline{O}}$-object and so that composition is continuous in each argument ; an $\underline{\underline{O}}$-functor $G:A \to X$ of $\underline{\underline{O}}$-categories is one which is continuous with respect to the order on the hom-sets. An $\underline{\underline{NO}}$-category is an $\underline{\underline{O}}$-category whose hom -sets are equipped with a binary function so that they form an $\underline{\underline{NO}}$-object and so that composition is linear in each argument ; an $\underline{\underline{NO}}$-functor $G:A \to X$ of $\underline{\underline{NO}}$-categories is an $\underline{\underline{O}}$-functor which is linear with respect to the union on the hom-sets.

Note that all the above categories are $\underline{\underline{O}}$-categories with respect to the natural pointwise ordering of morphisms ; further $\underline{\underline{NO}}$, $\underline{\underline{ND}}$ and $\underline{\underline{ND}}_\perp$ are all $\underline{\underline{NO}}$-categories with respect to the natural pointwise union. Any small product of $\underline{\underline{NO}}$-categories is an $\underline{\underline{NO}}$-category and so the product functor is an $\underline{\underline{NO}}$-functor (which is also strict on the hom -sets).

Definition 3.4 : Let $G:A \to X$ be an $\underline{\underline{O}}$-functor. Then $f:x \to Ga$ is a G-orderepi iff whenever $a \xrightarrow{g,\, g'} a'$ are such that $(Gg)f \sqsubseteq (Gg')f$ then $g \sqsubseteq g'$.

Lemma 3.5 : Let $G:A \to X$ be an $\underline{\underline{O}}$-functor such that every $f:x \to Ga$ factorises as

$x \xrightarrow{f'} Ga' \xrightarrow{Gg} Ga$ where f' is a G-orderepi. Then the left adjoint of G is also an $\underline{\underline{O}}$-functor and if G is an $\underline{\underline{NO}}$-functor its left adjoint is an $\underline{\underline{NO}}$-functor too.

<u>Powerdomains</u> : The evident forgetful functor $V_2:\underline{\underline{NO}} \to \underline{\underline{O}}$ has a left-adjoint $\mathcal{P}:\underline{\underline{O}} \to \underline{\underline{NO}}$ which is an $\underline{\underline{O}}$-functor ; further $\mathcal{P}$ cuts down to a left-adjoint to each of the forgetful functions $V_1:\underline{\underline{ND}} \to \underline{\underline{CPO}}$, V_o: $\underline{\underline{ND}}_\perp \to \underline{\underline{CPO}}_\perp$. The powerdomain construction in [Plo1][Smy1] is the restriction of $\mathcal{P}$: $\underline{\underline{CPO}} \to \underline{\underline{ND}}$ to the ω-algebraic case and then the unit map is singleton, $\{\!|\ \cdot\ |\!\}$: $D \to \mathcal{P}(D)$ and the "big union" $\biguplus$:$\mathcal{P}(\mathcal{P}(D)) \to \mathcal{P}(D)$ is the multiplication of the associated monad.

Other powerdomain constructions [Smy1][Mil1] can be treated similarly. Smyth's one can be obtained by adding the inequation :

4. $(x \cup y) \sqsubseteq x$.

If instead we add :

5. $(x \cup y) \sqsupseteq x$

we would obtain a construction involving the "other half" of the Egli-Milner ordering. Variations with an empty set [Mil1] are obtained by considering algebras, $\langle D, \sqsubseteq, \cup, \emptyset\rangle$ where $\emptyset$ is an element of D satisfying :

6. $(x \cup \emptyset) = x$.

Other possibilities are to consider a strict construction with the equation :

7. $(x \cup \perp) = \perp$

or to drop the absorption axiom to obtain a kind of multiset construction.

In all cases where we have the experiènce, all the neccessary auxiliary functions can be obtained from categorial considerations ; it is not clear however whether the required properties can be (conveniently) so obtained and we might need detailed constructions as in [Plo1][Smy1][Mil1], although they are not necessary in the present paper.

<u>Sums</u> : The category $\underline{\underline{ND}}_\perp$ has binary sums ; that is for any nd-domains, D_o, D_1 there is another (D_o+D_1) and strict continuous linear functions, in_i : $D_i \to (D_o+D_1)(i=0,1)$ such that for any other nd-domain, F, and strict continuous linear functions $f_i:D_i \to F(i=0,1)$ there is a unique such function $[f_o,f_1]:(D_o+D_1) \to F$ such that the following diagrams commute :

$$\begin{tikzcd}
D_i \arrow[d, "in_i"'] \arrow[dr, "f_i"] & \\
D_o+D_1 \arrow[r, "{[f_o,f_1]}"'] & F
\end{tikzcd} \qquad (i=0,1)$$

Further, [.,.] is strict continuous and linear on the hom-sets and so is $+ : \underline{\underline{ND}}^2 \to \underline{\underline{ND}}_\perp$ considered as a functor (it is an $\underline{\underline{NO}}$-functor).

Lifting : The forgetful functor $V_\perp : \underline{\underline{ND}}_\perp \to \underline{\underline{NO}}$ has a left adjoint $(.)_\perp : \underline{\underline{NO}} \to \underline{\underline{ND}}_\perp$; that is for any $\underline{\underline{NO}}$ object, D, there is an nd-domain, $(D)_\perp$, and a continuous linear up : $D \to (D)_\perp$ such that for any other continuous linear f: $D \to E$ there is a unique strict continuous linear lift(f) : $(D)_\perp \to E$ such that the following diagram commutes:

$$\begin{tikzcd}
D \arrow[d, "up"'] \arrow[dr, "f"] & \\
D_\perp \arrow[r, "lift(f)"'] & E
\end{tikzcd}$$

Further lift(.) is continuous and linear as is $(.)_\perp$ on the hom-sets. Finally, we note that $(.)_\perp$ cuts down to a left adjoint to the forgetful functor from $\underline{\underline{ND}}_\perp$ to $\underline{\underline{ND}}$.

Tensor products : The point of the tensor product is to reduce multilinear functions to linear ones.

Definition 3.6 : Let A,B,C be nd-domains. A continuous function $f : A \times B \to C$ is bistrict iff for all b in B $f(\perp,b) = \perp$ and for all a in A, $f(a,\perp) = \perp$; it is bilinear iff for all a, a' in A and b in B, $f(a \cup a',b) = f(a,b) \cup f(a',b)$ and for all a in A, b, b' in B, $f(a,b \cup b') = f(a,b) \cup f(a,b')$. (N.B. We are not assuming f either strict or linear). We let Bislin (A,B;C) be the set of bistrict, bilinear continuous functions from A×B to C equipped with the pointwise order and union (and so an nd-domain).
Note the bistrict functions are strict and linear binary functions are bilinear.

Any nd-domains A,B have a tensor product $A \otimes B$; that is, there is a bistrict, bilinear continuous $\otimes : A\times B \to A\otimes B$ which is universal in the sense that for any bistrict, bilinear f: $A\times B \to C$ there is a unique strict, linear continuous slin (f): $A\otimes B \to C$ such that the following diagram commutes :

$$\begin{tikzcd}
A\times B \arrow[d, "\otimes"'] \arrow[dr, "f"] & \\
A\otimes B \arrow[r, "slin(f)"'] & C
\end{tikzcd}$$

Further slin : Bislin $(A,B;C) \cong \mathrm{Hom}_{\underline{\underline{ND}}_\perp}(A\otimes B,C)$ is an isomorphism of nd-domains. We extend $\otimes$ to a functor $\otimes : \underline{\underline{ND}}_\perp \times \underline{\underline{ND}}_\perp \to \underline{\underline{ND}}_\perp$ by requiring that the following diagram always commutes :

$$\begin{tikzcd} A\times B \arrow[r, "f\times g"] \arrow[d, "\otimes"'] & A'\times B' \arrow[d, "\otimes"] \\ A\otimes B \arrow[r, "f\otimes g"] & A'\otimes B' \end{tikzcd}$$

Then we find that $\otimes$ is continous, linear and bistrict on the hom-sets.

Domain Equations : All the theory in [Smy2] applies to $\underline{\underline{ND}}_\perp$ and as constructions we can use $\times$, powers, $\mathcal{P} \circ V_\circ$, $+$, $(.)_\perp \circ V_\perp$, $\otimes$ (and others not mentioned here). In the present work we only need to solve the one equation :

$$\alpha\colon R \cong (\mathcal{P}(S_\perp) + (\mathcal{P}(S_\perp) \otimes (R)_\perp))^S$$

(where we have omitted the $V_\perp$). This gives us the nd-domain, R, of resumptions, which have the same motivation as the corresponding cpo in [Plo1]. Below we shall treat the isomorphism as an equality, omitting to write α or α^{-1}.

4. DENOTATIONAL SEMANTICS

We present a useful abbreviation. Suppose a,b,c are different variables of types $\mathcal{P}(S_\perp)$, $\mathcal{P}(S_\perp)$, R, respectively, and ——— , --- , ... , are expressions of types $(\mathcal{P}(S_\perp) + (\mathcal{P}(S_\perp) \otimes R_\perp))$, D, D respectively where --- is strict and linear in a, and .... is strict in b and linear in b and c ; then the expression, e, where :

$$e = (\underline{\text{cases}} \text{ ——— } \underline{\text{first}}\ a : \text{----}\ \underline{\text{second}}\ b, c, : \ldots)$$

is of type D and abbreviates :

$$[\lambda a \in \mathcal{P}(S_\perp).\ \text{----}\ ,\ \text{slin}\ (\lambda b \in \mathcal{P}(S_\perp),\ d \in R_\perp.\ \text{lift}(\lambda c \in R.\ \ldots)(d))]\ (\text{———})$$

where d is a new variable of type $R_\perp$ not free in .... . There are two "evaluation" values for e :

$$(\underline{\text{cases}}\ \text{in}_0(a)\ \underline{\text{first}}\ a : \text{----}\ \underline{\text{second}}\ b,c : \ldots) = \text{---},$$

$$(\underline{\text{cases}}\ \text{in}_1(b\otimes \text{up}(c))\ \underline{\text{first}}\ a : \text{---}\ \underline{\text{second}}\ b,c : \ldots) = \ldots .$$

From now on we will often omit in_0, in_1, up and $\{.\}$ when they are clear from the context. If ——— , ----, ... are all continuous in a variable, x , so is e ; if ——— is strict in x or else if both ---- and .... are strict in x then e is strict in x ; if ——— is linear in x, but x does not occur free in ---- , ... or else if both ---- and .... are linear in x, but x does not occur free in ——— then e is linear in x.

We now consider various useful combinators.

Sequence : The sequence combinator is the least continuous $*\colon R\times R \to R$ such that :

$$r_1 * r_2 = \langle \underline{\text{cases}}\ r_1(\sigma)\ \underline{\text{first}}\ a : a\otimes r_2\ \underline{\text{second}}\ b,c : b\otimes(c * r_2)\rangle\ \sigma \in S$$

The sequence combinator is bilinear and left-strict.

Parallelism : The parallelism combinator is the least continuous $*: R\times R \to R$ such that

$$r_1 \parallel r_2 = \langle \underline{\text{cases}}\ r_1(\sigma)\ \underline{\text{first}}\ a{:}a\otimes r_2\ \underline{\text{second}}\ b,c{:}b\otimes(c \parallel r_2)\rangle \quad \sigma \in S$$

$$\cup\langle \underline{\text{cases}}\ r_2(\sigma)\ \underline{\text{first}}\ a{:}a\otimes r,\ \underline{\text{second}}\ b,c{:}b\otimes(r \parallel c)\rangle \quad \sigma \in S.$$

It is bilinear.

Coroutine : The coroutine combinator is the least continuous co : $R\times R \to R$ such that

$$r_1 \text{ co } r_2 = \langle \underline{\text{cases}}\ r_1(\sigma)\ \underline{\text{first}}\ a{:}a\ \underline{\text{second}}\ b,c{:}b\otimes(r_2 \text{ co } c)\rangle \quad \sigma \in S$$

It is bilinear and left-strict.

The denotational semantics of our language is given by a function $\mathcal{V}$:Stat $\to$ R defined by structural induction on statements :

I. $\mathcal{V}[\![a]\!] = \langle \mathcal{A}[\![a]\!](\sigma)\rangle\ \sigma \in S$

II. $\mathcal{V}[\![s_1 ; s_2]\!] = \mathcal{V}[\![s_1]\!] * \mathcal{V}[\![s_2]\!]$

III. $\mathcal{V}[\![\underline{\text{if}}\ b\ \underline{\text{then}}\ s_1\ \underline{\text{else}}\ s_2]\!] = \langle \underline{\text{if}}\ \mathcal{E}[\![b]\!](\sigma)\ \underline{\text{then}}\ \mathcal{V}[\![s_1]\!]_\sigma\ \underline{\text{else}}\ \mathcal{V}[\![s_2]\!]_\sigma\rangle \quad \sigma \in S$

IV. $\mathcal{V}[\![\underline{\text{while}}\ b\ \underline{\text{do}}\ s]\!] = Y(\lambda r \in R.\ \langle \underline{\text{if}}\ \mathcal{E}[\![b]\!](\sigma)\ \underline{\text{then}}\ (\mathcal{V}[\![s]\!] * r)_\sigma\ \underline{\text{else}}\ \sigma\rangle\ \sigma \in S)$

V. $\mathcal{V}[\![s_1\ \underline{\text{or}}\ s_2]\!] = \mathcal{V}[\![s_1]\!] \cup \mathcal{V}[\![s_2]\!]$

VI. $\mathcal{V}[\![s_1\ \underline{\text{par}}\ s_2]\!] = \mathcal{V}[\![s_1]\!] \parallel \mathcal{V}[\![s_2]\!]$

VII. $\mathcal{V}[\![s_1\ \underline{\text{co}}\ s_2]\!] = \mathcal{V}[\![s_1]\!] \text{ co } \mathcal{V}[\![s_2]\!]$.

Again the method is flexible for specifying interruption points ; if we had conditionals to be interruptable after the test we would have written :

III'. $\mathcal{V}[\![\underline{\text{if}}\ b\ \underline{\text{then}}\ s_1\ \underline{\text{else}}\ s_2]\!] = \langle \underline{\text{if}}\ \mathcal{E}[\![b]\!](\sigma)\ \underline{\text{then}}\ \sigma \otimes \mathcal{V}[\![s_1]\!]\ \underline{\text{else}}\ \sigma \otimes \mathcal{V}[\![s_2]\!]\rangle$

$\sigma \in S$.

5. RELATIONS BETWEEN THE TWO SEMANTICS

We begin by showing that the denotational semantics, $\mathcal{V}$, can be derived from the operational semantics as "the least model of $\to$". Specifically we can regard (Stat $\to$ R) as an nd-domain - the power R^{Stat} - and define a continuous map $\psi : R^{\text{Stat}} \to R^{\text{Stat}}$ by :

$$\psi(\xi)\ [\![s]\!]_\sigma = \cup\{\sigma' \mid \langle s,\sigma\rangle \to \sigma'\} \cup$$

$$\cup\{\sigma' \otimes \xi[\![s']\!] \mid \langle s,\sigma\rangle \to \langle s',\sigma'\rangle\}$$

The definition makes sense as, by the properties of $\to$, at least one of the sets on the right is non-empty and both are finite.

So putting $\mathcal{W}_n = \psi^n(\perp)$, the least fixed-point of ψ is $\mathcal{W} =_{\text{def}} \bigsqcup_{n\geq 0} \mathcal{W}_n$.

Lemma 5.1 $\mathcal{V} = \mathcal{W}$.

Proof (Outline) One proves that $\mathcal{W}$ satisfies the equations defining $\mathcal{V}$.
For example to show $\mathcal{W}[\![s_1 \underline{\text{par}}\ s_2]\!] = \mathcal{W}[\![s_1]\!] \,||\, \mathcal{W}[\![s_2]\!]$ one proves by induction on n that $\mathcal{W}_n [\![s_1 \underline{\text{par}}\ s_2]\!] \sqsubseteq \mathcal{W}[\![s_1]\!] \,||\, \mathcal{W}[\![s_2]\!]$ and similarly that $\mathcal{W}[\![s_1]\!] \,||_n\, \mathcal{W}[\![s_2]\!] \sqsubseteq \mathcal{W}[\![s_1 \underline{\text{par}}\ s_2]\!]$ where $||_n$ is the nth approximant to $||$. ◪

Next we recast the definition of $\mathcal{B}$:Stat $\rightarrow$ (S $\rightarrow \mathcal{P}(S_\perp)$) in the same style defining $\Phi : (\mathcal{P}(S_\perp)^S)^{Stat} \rightarrow (\mathcal{P}(S_\perp)^S)^{Stat}$ by :

$$\Phi(\mathcal{E})[\![s]\!]_\sigma = \cup\{\sigma' \mid <s,\sigma> \rightarrow \sigma'\} \;\cup$$
$$\cup\{\mathcal{E}[\![s']\!]_{\sigma'} \mid <s,\sigma> \rightarrow <s',\sigma'>\}$$

Lémma 5.2 $\mathcal{B} = Y(\Phi)$

Proof Put $\mathcal{E}_n = \Phi^n(\perp)$ and then $\mathcal{E} =_{def} Y(\Phi) = \bigsqcup_n \mathcal{E}_n$.
One proves by induction on the length of the derivation that $<s,\sigma> \overset{*}{\rightarrow} \sigma'$ implies $\sigma' \in \mathcal{E}[\![s]\!]_\sigma$ and by induction on n that $\sigma' \in \mathcal{E}_n[\![s]\!]_\sigma$ implies $<s,\sigma> \overset{*}{\rightarrow} \sigma'$; next one shows that if $<s,\sigma> = <s_o,\sigma_o> \rightarrow \ldots \rightarrow <s_m,\sigma_m> \rightarrow \ldots$ is an infinite derivation sequence then, by induction on n, $\perp \in \mathcal{E}_n[\![s]\!]_\sigma$ and finally one shows by induction on n that if for all s,σ we have $\perp \in \mathcal{E}_n[\![s]\!]_\sigma$ then there is a derivation sequence of length n from $<s,\sigma>$ and then applies Konig's lemma. ◪

Now let β be the least continuous function from R to $(\mathcal{P}(S_\perp))^S$ such that :

$\beta(r) = <\underline{\text{cases}}\ r(\sigma)\ \underline{\text{first}}\ a{:}a\ \underline{\text{second}}\ b,c : Ext(\beta(c))(b)> \quad \sigma \in S.$

Here Ext $:\mathcal{P}(S_\perp)^S \rightarrow (\mathcal{P}(S_\perp) \rightarrow \mathcal{P}(S_\perp))$ is defined so that Ext(f) is the unique strict continuous linear extension of f: $Ext(f)(X) = \{\sigma : \exists x \in X.\ \sigma \in \bar{f}(x)\}$ where $\bar{f}: S_\perp \rightarrow \mathcal{P}(S_\perp)$ is the unique strict extension of f.

Lemma 5.3 The continuous function $\beta^{Stat} : R^{Stat} \rightarrow (\mathcal{P}(S_\perp)^S)^{Stat}$ is strict and the following diagram commutes :

$$\begin{tikzcd}
R^{Stat} \arrow[r, "\Psi"] \arrow[d, "\beta^{Stat}"'] & R^{Stat} \arrow[d, "\beta^{Stat}"] \\
(\mathcal{P}(S_\perp)^S)^{Stat} \arrow[r, "\Phi"] & (\mathcal{P}(S_\perp)^S)^{Stat}
\end{tikzcd}$$

Proof Straightforward calculation. ◪

Theorem 5.5 $\mathcal{B} = \beta \circ \mathcal{V}$

Proof
$\mathcal{B} = Y(\Phi)$ (lemma 5.2)
$= \beta^{Stat}(Y\Psi)$ (lemma 5.3)
$= \beta \circ \mathcal{W}$ (by definition)
$= \beta \circ \mathcal{W}$ (lemma 5.1). ◪

Thus the semantics, $\mathcal{V}$, determines the behavior, $\mathcal{B}$ (via β).

Full abstraction

Given a measure of behaviour, such as $\mathcal{B}$, and relations between behaviours, such as $=$, $\sqsubseteq$, $\subseteq$ on $(\mathcal{P}(S_\perp))^S$, we can define corresponding substitutive behavioural relations, $\approx$, $\underset{\sim}{\sqsubset}$, $\underset{\sim}{\subset}$. First, a context is a statement, $C[\cdot, \ldots, \cdot]$ with several "holes" which can be filled by any statements $s_1, \ldots, s_n$ to give a statement, $C[s_1, \ldots, s_n]$; a formal definition is obtained by adding the production $s ::= [\ ]$ to those given above for Stat. Now the relations $\approx$, $\underset{\sim}{\sqsubset}$, $\underset{\sim}{\subset}$ on Stat are defined by :

$$\begin{array}{llll} s_1 \approx s_2 & \text{iff} & \forall C[.].\ \mathcal{B}[\![C[s_1]]\!] & = \mathcal{B}[\![C[s_2]]\!] \\ s_1 \underset{\sim}{\sqsubset} s_2 & \text{iff} & \forall C[.].\ \mathcal{B}[\![C[s_1]]\!] & \sqsubseteq \mathcal{B}[\![C[s_2]]\!] \\ s_1 \underset{\sim}{\subset} s_2 & \text{iff} & \forall C[.].\ \mathcal{B}[\![C[s_1]]\!] & \subseteq \mathcal{B}[\![C[s_2]]\!] \end{array}$$

Clearly $s_1 \approx s_2$ iff $(s_1 \underset{\sim}{\sqsubset} s_2 \underset{\sim}{\sqsubset} s_1)$ iff $(s_1 \underset{\sim}{\subset} s_2 \underset{\sim}{\subset} s_1)$.

Proposition 5.5

1. $\mathcal{V}[\![s_1]\!] = \mathcal{V}[\![s_2]\!] \Rightarrow s_1 \approx s_2$
2. $\mathcal{V}[\![s_1]\!] \sqsubseteq \mathcal{V}[\![s_2]\!] \Rightarrow s_1 \underset{\sim}{\sqsubset} s_2$
3. $\mathcal{V}[\![s_1]\!] \underset{\sim}{\subset} \mathcal{V}[\![s_2]\!] \Rightarrow s_1 \underset{\sim}{\subset} s_2$

Proof 1. $\mathcal{V}[\![s_1]\!] = \mathcal{V}[\![s_2]\!] \Rightarrow \forall C[.].\ \mathcal{V}[\![C[s_1]]\!] = \mathcal{V}[\![C[s_2]]\!]$ (by definition of $\mathcal{V}$) $\Rightarrow s_1 \approx s_2$ (by theorem 5.4).

2. $\mathcal{V}[\![s_1]\!] \sqsubseteq \mathcal{V}[\![s_2]\!] \Rightarrow \forall C[.].\ \mathcal{V}[\![C[s_1]]\!] \sqsubseteq \mathcal{V}[\![C[s_2]]\!]$ (by the continuity of $*$, $\cup$, $\|$, co and the definition of $\mathcal{V}$) $\Rightarrow s_1 \underset{\sim}{\sqsubset} s_2$ (by the continuity of β and theorem 5.4).

3. As 2, but using the linearity of $\cup$, the bilinearity of $*$, $\|$, co and the easily proved monotonicity in $\subseteq$ of the conditional and while constructs and β. ⊠

The rest of the section establishes, under certain reasonable assumptions, the converse of these implications, therely obtaining three full abstraction results. The assumptions are :

1. S is infinite but denumerable.

2. For each σ in S there is an element K_σ of A such that for all σ' in S, $\mathcal{A}[\![K_\sigma]\!](\sigma') = \sigma$ (a next instruction).

3. For each σ in S there is an element is_σ of BExp such that for all σ' in S, $\mathcal{E}[\![is_\sigma]\!](\sigma') = tt$ iff $(\sigma = \sigma')$.

Under these assumptions we have :

Lemma 5.6 If $\mathcal{V}[\![s_1]\!] \neq \mathcal{V}[\![s_2]\!]$ then there is a context, $C[.]$ and a state σ so that $\bot \notin \mathcal{B}[\![C[s_1]]\!](\sigma) \cup \mathcal{B}[\![C[s_2]]\!](\sigma)$ and $\mathcal{B}[\![C[s_1]]\!](\sigma) \neq \mathcal{B}[\![C[s_2]]\!](\sigma)$.

which is enough to show ;

Theorem 5.7. 1. $\mathcal{V}[\![s_1]\!] = \mathcal{V}[\![s_2]\!] \Leftrightarrow s_1 \approx s_2 \Leftrightarrow$

$\mathcal{V}[\![s_1]\!] \subseteq \mathcal{V}[\![s_2]\!] \Leftrightarrow s_1 \lessapprox s_2$

2. $\mathcal{V}[\![s_1]\!] \subseteq \mathcal{V}[\![s_2]\!] \Leftrightarrow s_1 \lessapprox s_2$

Proof 1 The lemma implies that $\mathcal{V}[\![s_1]\!] \neq \mathcal{V}[\![s_2]\!]$ implies $s_1 \not\lessapprox s_2$.
So $\mathcal{V}[\![s_1]\!] = \mathcal{V}[\![s_2]\!] \Rightarrow \mathcal{V}[\![s_1]\!] \subseteq \mathcal{V}[\![s_2]\!] \Rightarrow s_1 \lessapprox s_2 \Rightarrow \mathcal{V}[\![s_1]\!] = \mathcal{V}[\![s_2]\!]$ and the rest of 1 is immediate.

2. First $s_1 \lessapprox s_2 \Rightarrow (s_1 \text{ or } s_2) \approx s_2$. For let $C[.]$ be a context ; then $\mathcal{B}[\![C[s_1 \text{ or } s_2]]\!] \subseteq \mathcal{B}[\![C[s_2 \text{ or } s_2]]\!]$ (as $s_1 \lessapprox s_2$) $= \mathcal{B}[\![C[s_2]]\!]$ (as $\mathcal{V}[\![s_2 \text{ or } s_2]\!] = \mathcal{V}[\![s_2]\!]$ shows $s_2 \text{ or } s_2 \approx s_2$) and conversely as $\mathcal{V}[\![s_2]\!] \subseteq \mathcal{V}[\![s_1 \text{ or } s_2]\!]$, we have $\mathcal{B}[\![C[s_2]]\!] \subseteq \mathcal{B}[\![C[s_1 \text{ or } s_2]]\!]$. But then by 1, $\mathcal{V}[\![s_1]\!] \subseteq \mathcal{V}[\![s_1 \text{ or } s_2]\!] = \mathcal{V}[\![s_2]\!]$. ⊠

We now outline the proof of the lemma. First for any pair $<s,\sigma>$ put :

$A_{s,\sigma} = \{\sigma' \mid <s,\sigma> \to \sigma'\}$

$B_{s,\sigma} = \{\sigma' \mid \exists s'. <s,\sigma> \to <s',\sigma'>\}$

and for any σ' in $B_{s,\sigma}$:

$st[s,\sigma,\sigma'] = (s'_1 \text{ or } (\ldots(s'_{n-1} \text{ or } s'_n)\ldots))$

where $\{s'_1,\ldots,s'_n\} = \{s' \mid <s,\sigma> \to <s',\sigma'>\}$ and clearly $n \neq 0$.

We say $<s,\sigma>$ is of types 1,2,3 according as $B_{s,\sigma}$, $A_{s,\sigma}$ or neither is $\emptyset$.

We have the useful formulae :

$$\mathcal{W}_{n+1}[\![s]\!]_\sigma = \cup A_{s,\sigma} \cup \bigcup_{\sigma' \in B_{s,\sigma}} \sigma' \otimes \mathcal{W}_n[\![st[s,\sigma,\sigma']]\!]$$

$$\mathcal{V}[\![s]\!]_\sigma = \cup A_{s,\sigma} \cup \bigcup_{\sigma' \in B_{s,\sigma}} \sigma' \otimes \mathcal{V}[\![st[s,\sigma,\sigma']]\!]$$

$$\mathcal{B}[\![s]\!]_\sigma = \cup A_{s,\sigma} \cup \bigcup_{\sigma' \in B_{s,\sigma}} \mathcal{B}[\![st[s,\sigma,\sigma']]\!](\sigma')$$

Define the relation $\sim$ between such pairs by :

$<s,\sigma> \sim <s',\sigma'>$ iff $(A_{s,\sigma} = A_{s',\sigma'})$ and $(B_{s,\sigma} = B_{s',\sigma'})$

Lemma 5.8 If $\mathcal{W}_n[\![s_1]\!] \neq \mathcal{W}_n[\![s_2]\!]$ then there are statements $s_i = s_i^0,\ldots,s_i^m$ $(m \geq 0)$ states $\sigma^0,\ldots,\sigma^m$ and states $\bar{\sigma}^j$ in $B_{s_1^j,\sigma^j}$ $(j < m)$ such that $s_i^{j+1} = st[s_i^j,\sigma^j,\bar{\sigma}^j]$ $(j < m)$, $<s_1^j,\sigma^j> \sim <s_2^j,\sigma^j>$ $(j < m)$ but $<s_1^m,\sigma^m> \not\sim <s_1^m,\sigma^m>$.

This sets up the path we want to follow to extract a difference. For assuming $\mathcal{V}[\![s_1]\!] \neq \mathcal{V}[\![s_2]\!]$ we have $\mathcal{W}_n[\![s_1]\!] \neq \mathcal{W}_n[\![s_2]\!]$ for some n. And we apply the lemma to obtain $s_i^j (j < m ; i=1,2)$, σ^j $(j \leq m)$, $\bar{\sigma}^j$ $(j < m)$.

Now for a state $\surd$ and a statement P^m (to be chosen later) we define statements $P^j (0 \leq j < m)$ by :

$$P^j = (\underline{\text{if}}\ \text{is}_{\bar{\sigma}^j}\ \underline{\text{then}}\ (K_{\sigma^{(j+1)}} ; P^{j+1})\ \underline{\text{else}}\ K_{\surd})$$

and for a statement Q' (to be chosen later) we set :

$$Q = (\underline{\text{if}}\ \text{is}_{\sigma^1}\ \underline{\text{then}}\ K_{\surd}\ \underline{\text{else}}$$
$$(\underline{\text{if}}\ \text{is}_{\sigma^2}\ \underline{\text{then}}\ K_{\surd}\ \underline{\text{else}}$$
$$\cdots\cdot$$
$$(\underline{\text{if}}\ \text{is}_{\sigma^m}\ \underline{\text{then}}\ K_{\surd}\ \underline{\text{else}}\ Q')\ldots))$$

and then calculate the following two formulae :

$$\mathcal{B}[\![s_i^0\ \underline{\text{co}}\ P^0]\!](\sigma^0) = (\bigcup_{j<m} A^j) \cup C \cup \mathcal{B}[\![s_i^m\ \underline{\text{co}}\ P^m]\!](\sigma^m)$$

(where $C \subseteq \{\surd\}$)

$$\mathcal{B}[\![(s_i^0 ; Q)\ \underline{\text{co}}\ P^0]\!](\sigma^0) = C \cup \bigcup\ \mathcal{B}[\![(s_i^m ; Q)\ \underline{\text{co}}\ P^m]\!](\sigma^m)$$

(where $C \subseteq \{\surd\}$)

Then the proof is completed by considering various cases based on the types of $<s_1^m,\sigma^m>$, $<s_2^m,\sigma^m>$ respectively and using one of the contexts $([\,]\ \underline{\text{co}}\ P^0)$ or $(([\,] ; Q)\ \underline{\text{co}}\ P^0)$ and choices of $\surd$, P^m, Q' as appropriate for the case at hand.

6. DISCUSSION

We make a few critical remarks to obtain some perspective on the above results. First the notion of behaviour chosen is inappropriate for languages for writing continuously interacting programs expressly written not to terminate. One should study our language with the addition of some I/O instructions and a different notion of behaviour. Again the coroutine instruction is somewhat peculiar and its rôle is somewhat similar to that of the "parallel or" in [Plo2] ; without it our semantics would, we conjecture, not be fully abstract, as we would have :

$$(x: = x) ; (x: = x) \approx (x: = x)$$

$$(x: = g(f(x))) \lesssim (x: = f(x) : x: = g(x))$$

We could also study definability questions, as in [Plo2] , and look for proof rules for $\approx$,$\lesssim$,$\lesssim$ using the semantics. Most importantly, our language is hardly a good model of communicating processes, and we feel it is rather important to study many other models of parallelism ([Bri], [Hoa] [Mil] and others) before claiming that the semantics of parallelism is understood.

ACKNOWLEDGEMENTS

We thank the Science Research Council and Orsay University, Paris for their support.

REFERENCES

[Bri] Brinch Hansen, P. (1978), Distributed Processes : A Concurrent Programming Concept. Comm. ACM 21, 11, pp. 934-940.

[Egl] Egli, H. (1975) : A Mathematical Model for Nondeterministic Computation .

[Hen] Hennessy, M.C.B. and Ashcroft, E.A. (1979) : A Mathematical Semantics for a Nondeterministic Typed λ-calculus. To appear.

[Hoa] Hoare, C.A.R. (1978) : Communicating Sequential Processes. Comm. ACM 21, 8, pp. 666-677.

[Mac] Mac Lane, S. (1971) : Categories for the Working Mathematician. Berlin, Springer Verlag .

[Mil1] Milne, G.J. and Milner, R. (1977) : Concurrent processes and their syntax. To appear in the J.A.C.M.

[Mil2] Milner, R. (1977) : Fully Abstract Models of Typed λ-calculi. T.C.S.

[Plo1] Plotkin, G.D. (1976) : A Powerdomain Construction. SIAM Journal on Computing 5, 3, pp. 452-487.

[Plo2] Plotkin , G.D. (1977) : LCF Considered as a Programming Language. T.C.S. 5, pp. 223-255.

[Smy1] Smyth, M. (1978) : Powerdomains. J.C.S.S. 16, 1.

[Smy2] Smyth, M. and Plotkin, G.D. (1978) : The Category Theoretic Solution of Recursive Domain Equations. University of Edinburgh : D.A.I. Research Report N° 60.