Real Time Agents

Luca Cardelli
Dept. of Computer Science
University of Edinburgh
JCMB The King's Buildings
Edinburgh EH9 3JZ, Scotland

Introduction

This paper is inspired by Milner's approach to synchronous processes, as reported in (Milner 82). The main differences are the use of a dense time domain and a dense-nondeterminism operator. Milner has shown that many of the characteristics of concurrent processes can be modelled and, more importantly, manipulated in an algebraic framework tailored to synchronous discrete interaction. Although much can be done in a discrete-time model by reducing the grain of discreteness to the desired level, we think it is interesting to see what can be gained in a dense-time framework and what additional difficulties arise.

At an appropriate level of abstraction there are entities which act and influence each other's behaviour through a continuous interaction. These entities are called here agents and their interactions are assumed to happen in real time (we use real numbers as a standard example of dense order). Agents progress by performing actions. Actions are denoted by the letters a,b,c and d, and the set of all the actions is A. Actions can be performed concurrently, so we denote by $a\cdot b$ (or simply ab) the simultaneous occurrence of the actions a and b. We also admit a neutral action 1, so that $(A,\cdot,1)$ is an abelian monoid.

Communication between agents can be modelled by requiring A to be a commutative group $(A,\cdot,1,^{-})$. A successful communication between two agents is represented by the matching of two complementary actions a and $\bar{a}$. The fact that $a\bar{a} = 1$ means that communication involves exactly two agents, that the respective communication capabilities are consumed during the process and that an external observer is unable to tell which communication took place (he can only observe 1). Note that communication here means simple synchronisation, without passage of values.

The central idea in real time agents is the explicit use of time information when expressing the behaviour of agents. Time is assumed to be dense, i.e. for every two instants t',t" it is always possible to find an instant t such that $t' < t < t''$. We shall formalise the idea of observing a real time system during intervals of time (i.e. not observing at time instants) and we want to rule out the possibility of observing zero-length actions. Hence the variables denoting time will range over a dense domain $\mathbb{K}$ (for Kronos) = $\mathbb{R}^+$, that is the set of strictly positive real numbers. The letters t,u,v will range over $\mathbb{K}$.

Deterministic Agents

We first examine agents which are deterministic, in the informal sense that every agent has a unique possible development in time. A formal property corresponding to the idea of determinism will be examined later.

We begin with a very simple set of operators to build agents. Our initial operator signature consists of: a constant $\mathbb{1}$ representing the neutral agent always performing the neutral action 1; a unary prefix operator $a[t]:$ which represents the act of performing the action a for an interval of time t; and the binary infix operator X representing the synchronous composition (coexistence) of two agents. An agent (denoted by p,q,r,s) is an expression over the signature $\Sigma^D = \{\mathbb{1}, a[t]:, X\}$ (where D stands for deterministic). The set of agents P^D is the free algebra over Σ^D.

Now we specify how our agents behave, by defining a set of binary relations $\xrightarrow[t]{a}$ (for $a \in A$ and $t \in \mathbb{K}$) over P^D. We read $p \xrightarrow[t]{a} q$ as "p moves to q performing a for an interval t", or "p takes t to move under a to q". The reduction rules for deterministic agents are as follows:

$(\mathbb{1} \rightarrow)$ $\quad \mathbb{1} \xrightarrow[t]{1} \mathbb{1}$

$(a[] \rightarrow)$ $\quad a[t]:p \xrightarrow[t]{a} p$

$(a[]a[] \rightarrow)$ $\quad a[t+u]:p \xrightarrow[t]{a} a[u]:p$

$(X \rightarrow)$ $\quad \dfrac{p \xrightarrow[t]{a} p' \qquad q \xrightarrow[t]{b} q'}{p \, X \, q \xrightarrow[t]{ab} p' \, X \, q'}$

Rule $(\mathbb{1} \rightarrow)$ asserts that $\mathbb{1}$ moves under 1 for an arbitrary interval t to produce $\mathbb{1}$ again. Rule $(a[] \rightarrow)$ says that $a[t]:p$ takes t to move under a to p, with $t>0$. Rule $(a[]a[] \rightarrow)$ has to do with the density of time; it says that after an interval t, $a[t+u]:p$ has only reached $a[u]:p$. Note that it is possible to split actions at arbitrary points, but this is done consistently so that the final outcome remains the same. Rule $(X \rightarrow)$ gives meaning to the coexistence of two agents: if p takes t to move under a to p' and q takes t to move under b to q', then p X q takes t (the same t) to move under a·b to p' X q'. Note that if q has form $b[t+u]:q''$, we can use $(a[]a[] \rightarrow)$ to get a t-derivation of q, so that we can use $(X \rightarrow)$.

This set of operational rules enjoys two fundamental properties:

Lemma 1 (Density Lemma) $p \xrightarrow[t+u]{a} r \;\Rightarrow\; \exists q.\; p \xrightarrow[t]{a} q,\; q \xrightarrow[u]{a} r$

Proof: Induction on the structure of the derivation of $p \xrightarrow[t+u]{a} r$ □

Lemma 2 (Persistency Lemma) $\forall p,t.\; \exists p_1,a_1,t_1 \ldots p_n,a_n,t_n.$

$$\textstyle\sum_i t_i = t \quad \text{and} \quad p \xrightarrow[t_1]{a_1} p_1 \ldots \xrightarrow[t_n]{a_n} p_n$$

Proof: Induction on the structure of p. The case p=p'Xp" needs the density lemma □

We shall abandon the persistency lemma later, but density is fundamental for all the different signatures we shall study. When adding a new operator to our signature, most of the results for the old signature extend to the new one, provided that density is preserved.

Agents will be observed by considering the sequences of actions they can perform. If the agents p and q are in the relation $p \xrightarrow[t]{a} q$, and q and r are in the relation $q \xrightarrow[u]{b} r$, then we can consider the composition of the relations $\xrightarrow[t]{a}$ and $\xrightarrow[u]{b}$ (denoted $\xrightarrow[t]{a} \circ \xrightarrow[u]{b}$) so that p and r are in the relation $p\ (\xrightarrow[t]{a} \circ \xrightarrow[u]{b})\ r$.

Definition 1 $\xrightarrow[t]{a} \circ \xrightarrow[u]{b} \;=\; \left\{(p,r) \,\middle|\, \exists q.\ (p,q) \in \xrightarrow[t]{a} \text{ and } (q,r) \in \xrightarrow[u]{b} \right\}$ □

We write $\xrightarrow[(t_1 \ldots t_n)]{(a_1 \ldots a_n)}$ for $\xrightarrow[t_1]{a_1} \circ \ldots \circ \xrightarrow[t_n]{a_n}$ $(n>0)$. Moreover a sequence of actions is denoted by $\hat{a} = (a_1 \ldots a_n)$ with $\#\hat{a} = n$, and a sequence of time intervals by $\hat{t} = (t_1 \ldots t_n)$ with $\#\hat{t} = n$ and $\sum\hat{t} = \sum_{1 \leq i \leq n} t_i$.

We want to observe actions in such a way that, for example, the sequences $\xrightarrow[(1,1)]{(a,a)}$ and $\xrightarrow[(2)]{(a)}$ are indistinguishable. This can be done by considering similar sequences in the following informal sense:

$\xrightarrow[(2,2,2,2)]{(a,b,b,b)}$ is similar to $\xrightarrow[(1,1,3,3)]{(a,a,b,b)}$; $\xrightarrow[(1,2)]{(a,b)}$ is not similar to $\xrightarrow[(2,1)]{(a,b)}$.

Definition 2 Similarity is the least equivalence relation, $\simeq$, between relations $\xrightarrow[\hat{t}]{\hat{a}}$ such that:

(i) If $a_1 = \ldots = a_n = b_1 = \ldots = b_m$ and $\sum\hat{t} = \sum\hat{u}$ then $\xrightarrow[\hat{t}]{\hat{a}} \simeq \xrightarrow[\hat{u}]{\hat{b}}$

(ii) If $\xrightarrow[\hat{t}']{\hat{a}'} \simeq \xrightarrow[\hat{u}']{\hat{b}'}$ and $\xrightarrow[\hat{t}'']{\hat{a}''} \simeq \xrightarrow[\hat{u}'']{\hat{b}''}$ then $\xrightarrow[\hat{t}']{\hat{a}'} \circ \xrightarrow[\hat{t}'']{\hat{a}''} \simeq \xrightarrow[\hat{u}']{\hat{b}'} \circ \xrightarrow[\hat{u}'']{\hat{b}''}$ □

We can also talk about sequences which are finer than other sequences:

Definition 3 $\xrightarrow[\hat{t}]{\hat{a}}$ is finer than $\xrightarrow[\hat{u}]{\hat{b}}$ when $\xrightarrow[\hat{t}]{\hat{a}} \leqslant \xrightarrow[\hat{u}]{\hat{b}}$, where $\leqslant$ is the least relation satisfying:

(i) $\xrightarrow[(t_1 \ldots t_n)]{(a \ldots a)} \leqslant \xrightarrow[\sum_i t_i]{a}$

(ii) If $\xrightarrow[\hat{t}']{\hat{a}'} \leqslant \xrightarrow[\hat{u}']{\hat{b}'}$ and $\xrightarrow[\hat{t}'']{\hat{a}''} \leqslant \xrightarrow[\hat{u}'']{\hat{b}''}$ then $\xrightarrow[\hat{t}']{\hat{a}'} \circ \xrightarrow[\hat{t}'']{\hat{a}''} \leqslant \xrightarrow[\hat{u}']{\hat{b}'} \circ \xrightarrow[\hat{u}'']{\hat{b}''}$ □

Theorem 1 $\leqslant$ is a partial order over the relations $\xrightarrow[\hat{t}]{\hat{a}}$. Moreover:

(i) If $\xrightarrow[\hat{t}]{\hat{a}} \leqslant \xrightarrow[\hat{u}]{\hat{b}}$ then $\xrightarrow[\hat{t}]{\hat{a}} \simeq \xrightarrow[\hat{u}]{\hat{b}}$

(ii) If $\xrightarrow[\hat{t}]{\hat{a}} \simeq \xrightarrow[(u)]{\hat{u}(b)}$ then $\xrightarrow[\hat{t}]{\hat{a}} \leqslant \xrightarrow[(u)]{\hat{u}(b)}$

(iii) The greatest lower bound of two similar sequences exists and is unique.

Proof: Directly from the definitions □

The density lemma implies the following:

Lemma 3 (Refinement Lemma) If $p \xrightarrow[\hat{t}]{\hat{a}} q$ and $\xrightarrow[\hat{u}]{\hat{b}} \leqslant \xrightarrow[\hat{t}]{\hat{a}}$ then $p \xrightarrow[\hat{u}]{\hat{b}} q$ □

The following abbreviation will be used:

Definition 4 $p \xrightarrow[\hat{t}]{\hat{a}}{}^{s} q$ if there exists $\xrightarrow[\hat{t}']{\hat{a}'} \simeq \xrightarrow[\hat{t}]{\hat{a}}$ such that $p \xrightarrow[\hat{t}']{\hat{a}'} q$ □

Informally, the behaviour of agents is given by their reduction chain, and we want to regard as equivalent agents which have the "same" reduction chains (i.e. which perform the "same" actions) even if they are syntactically different as members of P^D. After having defined a congruence relation $\sim$ over P^D so that $p \sim q$ iff they perform the same actions, we can then take the equivalence class of p in $P^D/\sim$ as the semantics of p.

We are going to define the following equivalence: p is equivalent to q iff whenever p can reduce under a single action $\xrightarrow[t]{a}$ to p', then q can reduce by a <u>similar</u> sequence $\xrightarrow[t]{a}{}^{s}$ to some q' equivalent to p', and vice versa. This equivalence is called <u>smooth equivalence</u> because it ignores the "density" of individual actions and only considers their coarse result. We first define a formula $\mathbb{D}(\approx)$ parametrically in an arbitrary relation over P^D:

Definition 5 $\mathbb{D}(\approx)$ = $p \approx q$ iff $\forall a \in A, \forall t \in \mathbb{K}$.

both $p \xrightarrow[t]{a} p' \Rightarrow \exists q'.\ q \xrightarrow[t]{a}{}^{s} q'$ and $p' \approx q'$

and $q \xrightarrow[t]{a} q' \Rightarrow \exists p'.\ p \xrightarrow[t]{a}{}^{s} p'$ and $p' \approx q'$ □

Definition 6 Smooth equivalence ($\sim$) is the maximal fixpoint of the equation $\mathbb{D}(\approx) = \approx$ in the lattice of binary relations over P^D □

Theorem 2 (Park's Induction Principle (Park 81))

$p \sim q$ iff $\exists R \subseteq P^D \times P^D$. (i) $(p,q) \in R$

(ii) $R \subseteq \mathbb{D}(R)$ □

Condition (ii) can be written more explicitely as:

$$(p,q) \in R \quad \Rightarrow \quad \begin{array}{l} (ii') \ \forall p \xrightarrow[t]{a} p'.\ \exists (p',q') \in R.\ q \xrightarrow[t]{a}{}^{s} q' \\ (ii'') \ \forall q \xrightarrow[t]{a} q'.\ \exists (p',q') \in R.\ p \xrightarrow[t]{a}{}^{s} p' \end{array}$$

Theorem 3

(i) $\sim$ is an equivalence relation.

(ii) $\sim$ is a congruence with respect to $\Sigma^D = \{1\!1,\ a[t]:,\ X\}$.

(iii) $P^D/\sim$ is a Σ^D-algebra.

Proof: (i) is easily verified.

(ii) We have to show that for every Σ^D-context $C(x)$: $p \sim q \Rightarrow C(p) \sim C(q)$.

It is enough to show (using Park's induction) that:

(1) $p \sim q \Rightarrow a[t]:p \sim a[t]:q$

(2) $p \sim q \Rightarrow p\,X\,r \sim q\,X\,r$ and $r\,X\,p \sim r\,X\,q$

For (1) take $R = \{(a[t]:p,\ a[t]:q) \mid p \sim q\} \cup \sim$, and proceed by Park's induction and analysis of the structure of the derivations. For (2), similarly, take $R = \{(p\,X\,r,\ q\,X\,r) \mid p \sim q\} \cup \sim$ (and symmetrically in the second case). Note that the density lemma is required.

(iii) This is a standard algebraic result, based on (ii) $\square$

We can now investigate the equivalence ($\sim$) of agents. The following laws hold:

$(X1\!1)$	$p\,X\,1\!1 \sim p$	$(1[\,]1\!1)$	$1[t]:1\!1 \sim 1\!1$
(X)	$p\,X\,q \sim q\,X\,p$	$(a[\,]a[\,])$	$a[t]:a[u]:p \sim a[t+u]:p$
(XX)	$p\,X\,(q\,X\,r) \sim (p\,X\,q)\,X\,r$	$(a[\,]X)$	$a[t]:p\,X\,b[t]:q \sim ab[t]:(pXq)$

All the laws can be proved smoothly by Park's induction. Both the congruence property for X and the factorisation law (a[]X) depend only on the density lemma; whenever we modify our signature we need only to make sure that the density lemma still holds.

The following results tell us that the above set of laws is rich and consistent:

Theorem 4 (Soundness) Let us denote by $\equiv$ the congruence defined by the set of laws $(X1\!1)$ ... $(a[\,]X)$. We say that p is convertible to q iff $p \equiv q$. Then:

$$p \equiv q \Rightarrow p \sim q$$

Proof: Induction on the derivation of $p \equiv q$, using the fact that $\sim$ is a congruence and the laws are valid $\square$

Theorem 5 (Normal Forms) Let $S_{i\leq n}\ a_i[t_i]:p$ abbreviate $a_1[t_1]:\ldots a_n[t_n]:p$ (for $n \geq 0$). An agent is in sequence form if it has the form $S_{i\leq n}\ a_i[t_i]:1\!1$.

An agent is in <u>normal form</u> if it is in sequence form $S_{i\leq n}\ a_i[t_i]: 1\!1$ with both $(n>0 \Rightarrow a_n \neq 1)$ and $(n \geq 2 \Rightarrow \forall i<n.\ a_i \neq a_{i+1})$. Then:

(i) Every agent is convertible to a sequence form.

(ii) Every sequence form is convertible to a normal form.

(iii) Every agent has a unique normal form.

Proof: Simple inductions on the structure of terms □

<u>Theorem 6</u> (Completeness)

$$p \sim q \quad \Rightarrow \quad p \equiv q$$

Proof: First prove that for p',q' in normal form, $p' \sim q' \Rightarrow p' \equiv q'$ by induction on the structure of p' and q' (this is easy because of the simple structure of normal forms: we even have $p' \sim q' \Rightarrow p' = q'$). In general, by the normal form theorem, p and q have respective normal forms p' and q' (so that $p \equiv p'$ and $q \equiv q'$). By soundness $p' \sim p \sim q \sim q'$. So by the first part of the proof $p' \equiv q'$. Hence $p \equiv p' \equiv q' \equiv q$ □

We said that our agents are deterministic: this can be stated formally in the following way:

<u>Theorem 7</u> (Determinism)

Vertical determinism: $p \xrightarrow[t]{a} q$ and $p \xrightarrow[u]{b} r$ implies $a = b$

Horizontal determinism:

(i) If $p \xrightarrow[\hat{t}]{\hat{a}} q$, $p \xrightarrow[\hat{u}]{\hat{b}} r$ and $\xrightarrow[\hat{t}]{\hat{a}} \simeq \xrightarrow[\hat{u}]{\hat{b}}$ then $q = r$

(ii) If $p \sim q$, $p \xrightarrow[\hat{t}]{\hat{a}} p'$, $q \xrightarrow[\hat{u}]{\hat{b}} q'$ and $\xrightarrow[\hat{t}]{\hat{a}} \simeq \xrightarrow[\hat{u}]{\hat{b}}$ then $p' \sim q'$

Proof: Structural induction on the left hand side of the arrows, plus in each case a simple lemma about the corresponding structure of the action and the right hand side of the arrow □

In this formal sense our agents are completely deterministic, and we can also see that it is possible to introduce two orthogonal kinds of nondeterminism. This will be done in the next section.

<u>Nondeterministic Agents</u>

Let us now extend our signature by the following operators. A constant 0 representing an agent with no actions; when a system reaches the state 0, a catastrophe occurs and time ceases to flow, hence 0 is called a disaster. A unary prefix operator $a(t)$: performing the action a for a positive interval of length <u>at most</u> t; we say that $a(t)$: introduces horizontal continuous nondeterminism in the sense

that arrows can be stretched horizontally according to the duration of a(t):.
A binary infix operator + representing the choice between two behaviours; we say that + introduces vertical discrete nondeterminism. We can imagine the behaviour of an agent as a (discontinuous) trajectory on the plane, with time on the x axis and the action monoid on the y axis; this explains the sense of the adjectives "horizontal" and "vertical".

The operational semantics is as follows. There are no axioms for 0. The agent a(t):p takes time $v \leq t$ to move under a to p, and a(t+u):p takes time $v \leq t$ to move under a to p + a(u):p. Hence a(t):p can choose at any move to shorten its life span by some amount; moreover at any point in time it can stop its a-action and start executing p. As for +, if p takes t to move under a to p', then p+q may move under a to p' taking time t, or else if q takes u to move under b to q', then p+q may move under b to q' taking time u.

$$(a()\rightarrow)\quad a(t):p \xrightarrow[v]{a} p \qquad v \leq t$$

$$(a()a()\rightarrow)\quad a(t+u):p \xrightarrow[v]{a} p+a(u):p \qquad v \leq t$$

$$(+\rightarrow)\quad \frac{p \xrightarrow[t]{a} p'}{p+q \xrightarrow[t]{a} p'} \qquad \frac{q \xrightarrow[u]{b} q'}{p+q \xrightarrow[u]{b} q'}$$

Applying the same definition of smooth equivalence to the new extended signature $\sum$ (freely generating the new set of agents P), we obtain the following laws:

(+0)	$p + 0 \sim p$	(a()+)	$a(t+u):p \sim a(t+u):p + a(t):p$
(+p)	$p + p \sim p$	(a()a())	$a(t+u):p \sim a(t):(p + a(u):p)$
(+)	$p + q \sim q + p$	(X0)	$p \text{ X } 0 \sim 0$
(++)	$p + (q + r) \sim (p + q) + r$	(X+)	$p \text{ X } (q + r) \sim (p\text{X}q) + (p\text{X}r)$
(1()11)	$1(t): 11 \sim 11$		

The density lemma is still valid (we must abandon the persistency lemma because of 0) and $\sim$ is a congruence. However the set of laws above is not complete, we lack the distributivity of a(t): over X and laws relating a(t): to a[t]:.

Laws relating a(t): and X are called <u>factorisation</u> <u>theorems</u> (the restriction operator $\lceil B$ used below is explained in the next section; the laws (FT2) and (FT4) hold also with all the $\lceil B$ elided):

(FT1) $(a(t):p \text{ X } b(t):q)\lceil B \sim 0$ if $ab \notin B$

(FT2) $(a(t):p \text{ X } b(t):q)\lceil B \sim (ab(t):(p \text{ X } q))\lceil B$

if either $\forall u<t.\ (p\text{X}(q+b(u):q))\lceil B \sim (p\text{X}q)\lceil B$
or $\forall u<t.\ \exists v\leq u.\ (p\text{X}(q+b(u):q))\lceil B \sim (p\text{X}q+a(v):(p\text{X}b(v):q))\lceil B$
and either $\forall u<t.\ ((p+a(u):p)\text{X}q)\lceil B \sim (p\text{X}q)\lceil B$

$\quad\text{or}\ \forall u<t.\ \exists v\leq u.\ ((p+a(u):p)Xq)\lceil B \sim (pXq+a(v):(pXb(v):q))\lceil B$
$\text{and either}\ \forall u<t.\ ((p+a(u):p)X(q+b(u):q))\lceil B \sim (pXq)\lceil B$
$\quad\text{or}\ \forall u<t.\ \exists v\leq u.\ ((p+a(u):p)X(q+b(u):q))\lceil B \sim (pXq+a(v):pXb(v):q)\lceil B$
$\text{and either}\ \forall u<t.\ (pXq+a(u):pXb(u):q)\lceil B \sim (pXq)\lceil B$
$\quad\text{or}\ \forall u<t.\ \exists v\leq u.\ (pXq+a(u):p+b(u):q)\lceil B \sim ((p+a(v):p)X(q+b(v):q))\lceil B$

(FT3) $(a(t):p\ X\ b[t]:q)\lceil B \sim 0 \qquad \text{if}\ \ ab \notin B$

(FT4) $(a(t):p\ X\ b[t]:q)\lceil B \sim (ab[t]:(pXq))\lceil B$

$\text{if}\ \ \forall u<t.\ (a(u):p\ X\ b[u]:q)\lceil B \sim (p\ X\ b[u]:q)\lceil B$
$\text{and}\ \forall u<t.\ \exists v\leq u.\ (a(u):p\ X\ b[u]:q)\lceil B \sim ((p+a(v):p)\ X\ b[u]:q)\lceil B$

These laws constitute a major departure from the equational style we have observed so far, and may be an indication that we have not chosen the best possible set of primitive operators. On the other hand they seem to reflect rather faithfully the complex relationships between a synchronous deterministic world (1l, a[t]:, X) and an asynchronous nondeterministic one (0, a(t):, +), and we could not devise a simpler formulation. The factorisation theorems can usually be much simplified in practical situations (e.g. replacing "$\forall u<t$" by "$\forall u$"), and they turn out to be very useful in proving <u>equational</u> laws of interesting derived operators, as we shall see later.

<u>Communication</u>

The <u>restriction</u> operator $\lceil B$, for $B \subseteq A$ and $1 \in B$ is used to extract a subset of the possible actions of an agent, inhibiting the rest of the actions.

$$(\lceil \rightarrow) \qquad \frac{p \xrightarrow[t]{a} q}{p\lceil B \xrightarrow[t]{a} q\lceil B} \qquad \text{if}\ \ a \in B$$

Thus $p\lceil B$ can only perform actions which are in B, and this can force some communication event inside p. The action 1 is never inhibited by definition; it represents the possible anonymous occurrence of a communication event inside p.

The <u>delabelling</u> operator $p\backslash\alpha$ is a particular case of restriction. We assume here that A is generated by a set of atomic actions $\alpha, \beta, \gamma \ldots$. Then $p\backslash\alpha$ is the restriction of p to the set of all the actions of A not containing α or $\bar{\alpha}$ as prime factors.

We also need a way of renaming actions, so that we can easily set up communication channels. The most general form of renaming is called a <u>morphism</u> $p\{\phi\}$ where ϕ: $A \rightarrow A$ is a monoid homomorphism:

$$(\{\phi\}\rightarrow) \qquad \frac{p \xrightarrow[t]{a} p'}{p\{\phi\} \xrightarrow[t]{\phi(a)} p'\{\phi\}}$$

We write $\{\alpha_i/\beta_i\}$ for the unique monoid morphism renaming the generators β_i to α_i and leaving the other generators unchanged.

We omit the laws for restriction and morphism, because they are not significantly different from those of (Milner 82).

Recursion

A recursive definition facility will now be introduced in our language. Its general form for a single recursive definition is:

$x \Leftarrow r$

where x is a variable and r is a context, i.e. a term possibly containing variables. We have the operational rule:

$$(\Leftarrow) \qquad \frac{r \xrightarrow[t]{a} p}{x \xrightarrow[t]{a} p}$$

To satisfy a definition like $x \Leftarrow 11 + a[t]:x$, it is sufficient to find a p such that $p \sim 11 + a[t]:p$ because all our laws are valid up to equivalence. In fact it is easy to show that $(\Leftarrow)$ implies $x \sim p$. But we still need to specify which particular x we want, when several of them are available, like in the definition $x \Leftarrow x$. To avoid this problem we restrict our admissible definitions to those having a unique solution up to equivalence; thus there is no doubt about which x we mean. In general we use sets of definitions, to take mutual recursion into account.

Definition 7 A definition set is a set of pairs $\{(x_i, r_i)\}$, written $\{x_i \Leftarrow r_i\}$ or $\hat{x} \Leftarrow \hat{r}$, where x_i are variables and r_i are contexts. A 1-step expansion of a definition set $\hat{x} \Leftarrow \hat{r}$ is obtained by replacing $x_i \Leftarrow r_i$ by $x_i \Leftarrow r_i\{r_j/x_j\}$ (for some i and j) in $\hat{x} \Leftarrow \hat{r}$. A finite expansion $\hat{x} \Leftarrow \hat{r}'$ of $\hat{x} \Leftarrow \hat{r}$ is an expansion obtained by a finite number of 1-step expansions $\square$

Definition 8 A variable x is guarded in a context r if all the occurrences of x are in subterms of r of the form a[t]:r' or a(t):r'. A context r is guarded if all its variables are guarded. A definition set $\{x_i \Leftarrow r_i\}$ is guarded if there is a finite expansion $\{x_i \Leftarrow r'_i\}$ such that each r'_i is guarded $\square$

In order to have unique solutions for our definition sets, we need to exclude definition sets which expand indefinitely but only approach a finite limit (i.e. such that the duration of their infinite chains of actions is finite). Definition sets which can expand for the same duration as their solutions are persistent.

Definition 9 A definition set $\{x_i \Leftarrow r_i\}$ is persistent if whenever $\hat{p} \sim \hat{r}\{\hat{p}/\hat{x}\}$ then for all j, $p_j \xrightarrow[t]{a} q_j$ implies that there exists a finite expansion r^o_j of r_j such that $r^o_j \xrightarrow[t]{a}{}^s r'_j$ with $r'_j\{\hat{p}/\hat{x}\} \sim q_j$ □

Every persistent definition set is guarded, and every finite guarded definition set is persistent, but there are infinite guarded definition sets which are not persistent (e.g. $\{Z_n \Leftarrow 1[n]:Z_{n/2} \mid n \in \mathbb{K}\}$).

Theorem 10 (Recursion Theorem)

Every persistent definition set $\hat{x} \Leftarrow \hat{r}$ has a unique solution up to $\sim$, i.e.:

$$p_i \sim r_i\{\hat{p}/\hat{x}\} \quad \text{and} \quad q_i \sim r_i\{\hat{q}/\hat{x}\} \quad \Rightarrow \quad p_i \sim q_i$$

Proof: Let $\approx = \{(C\{\hat{p}/\hat{x}\}, C\{\hat{q}/\hat{x}\}) \mid C \text{ is a context}\}$. By Park's induction:

(i) $p_i \approx q_i$ (take $C = x_i$)

(ii) $C\{\hat{p}/\hat{x}\} \xrightarrow[t]{a} P$ may hold because either $C \xrightarrow[t]{a} C'$ with $P = C'\{\hat{p}/\hat{x}\}$ (then also $C\{\hat{q}/\hat{x}\} \xrightarrow[t]{a} Q = C'\{\hat{q}/\hat{x}\}$, and $Q \approx P$), or x_j is not guarded in C and $p_j \xrightarrow[t]{a} P$. In the latter case, $\hat{r}$ is persistent and there is a finite expansion r^o_j with $r^o_j \xrightarrow[t]{a}{}^s r'_j$ and $r'_j\{\hat{p}/\hat{x}\} \sim P$. Then also $r^o_j\{\hat{q}/\hat{x}\} \xrightarrow[t]{a}{}^s r'_j\{\hat{q}/\hat{x}\}$, and since $q_j \sim r_j\{\hat{q}/\hat{x}\} \sim r^o_j\{\hat{q}/\hat{x}\}$, we have $q_j \xrightarrow[t]{a}{}^s Q \sim r'_j\{\hat{q}/\hat{x}\}$. Hence $C\{\hat{q}/\hat{x}\} \xrightarrow[t]{a}{}^s Q$ with $Q \approx P$ □

Indefinite Actions and Delays

We now use recursion and nondeterministic guards to define actions of indefinite duration in time (a.p):

$$a.p \Leftarrow a(1):(p + a.p)$$

The particular choice of unit delay above makes no difference, as we have:

$a(t):(p + a.p)$	$\sim a(t):(p + a.p + a.p)$	by (+p)
	$\sim a(t):(p + a.p + a(1):(p + a.p))$	by definition of a.p
	$\sim a(t+1):(p + a.p)$	by (a()+)
	$\sim a(1):(p + a.p + a(t):(p + a.p))$	by (a()+)

$$a.p \sim a(1):(p + a.p) \sim a(1):(p + a.p + a.p)$$

Hence $a.p \sim a(t):(p + a.p)$ for any t, by the recursion theorem.

Moreover a.p enjoys the laws:

(1.0)	$1.0 \sim \mathbb{1}$	(a.)	$a.p \sim a.(p + a.p)$
(1.𝟙)	$1.\mathbb{1} \sim \mathbb{1}$	(a.Xb.)	$a.p \times b.q \sim ab.(p \times q + a.p \times q + p \times b.q)$

Note the importance of the law (a.Xb.); it allows us to equationally factorise actions in horizontally nondeterministic agents, which we could not do for the "a(t):" operator. The first three laws can be proved easily by the recursion theorem. Law (a.Xb.) is proved using the factorisation theorems, thereby demonstrating some of their power:

$$\begin{aligned} a.p \times b.q &\sim a(1):(p + a.p) \times b(1):(q + b.q) && \text{by definition of a.p} \\ &\sim ab(1):((p + a.p) \times (q + b.q)) && (*) \\ &\sim ab(1):(p \times q + a.p \times q + p \times b.q + a.p \times b.q) && \text{by } (\times +) \end{aligned}$$

$$\begin{aligned} & ab.(p \times q + a.p \times q + p \times b.q) \\ & \quad ab(1):(p \times q + a.p \times q + p \times b.q + ab.(p \times q + a.p \times q + p \times b.q)) \end{aligned}$$

Hence $a.p \times b.q \sim ab.(p \times q + a.p \times q + p \times b.q)$ by the recursion theorem. The step leading to (*) uses a factorisation theorem (FT2); the four hypotheses of the theorem can be verified as follows (using the fact that $a.p \sim a(t):(p + a.p)$ and $b.q \sim b(t):(q + b.q)$):

$$\begin{aligned} &(1) \quad (p + a.p) \times (q + b.q + b(t):(q + b.q)) \sim (p + a.p) \times (q + b.q) \\ &(2) \quad (p + a.p + a(t):(p + a.p)) \times (q + b.q) \sim (p + a.p) \times (q + b.q) \\ &(3) \quad (p+a.p+a(t):(p+a.p)) \times (q+b.q+b(t):(q+b.q)) \sim (p+a.p) \times (q + b.q) \\ &(4) \quad (p+a.p) \times (q+b.q) + a(t):(p+a.p) \times b(t):(q+b.q) \sim (p+a.p) \times (q+b.q) \end{aligned}$$

A closely related operator to a.p is <u>indefinite</u> <u>delay</u>:

$$d_a p \Leftarrow p + a.p$$

where the agent p may be activated immediately, or delayed indefinitely by an action a. The following laws can all be easily derived from the properties of a.p:

$$\begin{aligned} & d_a 0 \sim \mathbb{K}_a \qquad \text{where } \mathbb{K}_a \Leftarrow a[1]: \mathbb{K}_a \\ & d_a \mathbb{K}_a \sim \mathbb{K}_a \\ & d_a(d_a p) \sim d_a p \\ & d_a p \times d_b q \sim d_{ab}(d_a p \times d_b q) \\ & d_a p \times d_b q \sim d_{ab}(d_a p \times q + p \times d_b q) \end{aligned}$$

<u>An Asynchronous Rising Edge Counter</u>

We now discuss an example of application of non deterministic guards. Suppose we have a boolean signal represented as $tt[t_1]:ff[t_2]:tt[t_3]:ff[t_4]: \ldots$, where the length of the intervals t_i is completely arbitrary. The problem consists in counting the number of rising edges (i.e. transitions from ff to tt) which have occurred in the signal at any given time. It is pretty well evident that there can be no solution using deterministic guards, as any proposal would be bound to fail on some input waveforms.

The counter has two states: Low_n and $High_n$, and n is increased at any passage from Low to High (for simplicity n is not supplied as an explicit output).

$$Low_n \Leftarrow ff(1):Low_n + tt(1):High_{n+1}$$
$$High_n \Leftarrow ff(1):Low_n + tt(1):High_n$$

Note how the guards tt and ff are programmed to last as long as their corresponding non synchronised inputs. Again, we first prove that the "1"s used in the definition are not significative using the technique shown in the previous section:

$$Low_n \sim ff(t):Low_n + tt(1):High_{n+1} \sim ff(1):Low_n + tt(t):High_{n+1}$$
$$High_n \sim ff(t):Low_n + tt(1):High_n \sim ff(1):Low_n + tt(t):High_n$$

The following equivalences state the correctness of the counter; they can be proved using (FT3) and (FT4):

$$(Low_n \times \overline{ff}[t]:p)\backslash ff \sim 1[t]:(Low_n \times p)\backslash ff$$
$$(High_n \times \overline{tt}[t]:p)\backslash tt \sim 1[t]:(High_n \times p)\backslash tt$$
$$(Low_n \times \overline{tt}[t]:p)\backslash tt \sim 1[t]:(High_{n+1} \times p)\backslash tt$$
$$(High_n \times \overline{ff}[t]:p)\backslash ff \sim 1[t]:(Low_n \times p)\backslash ff$$

Descriptive Operators

Some operators can be introduced just to talk about the properties of agents. In order to talk about synchrony we can introduce synchronisation operators Γ_t, designed to "impose" a clock of period t on an otherwise unsynchronised agent.

$$(\Gamma_t \rightarrow) \quad \frac{p \xrightarrow[t]{a} q}{\Gamma_t p \xrightarrow[t]{a} \Gamma_t q} \qquad\qquad (\Gamma_{t+u} \rightarrow) \quad \frac{\Gamma_t p \xrightarrow[u+v]{a} q}{\Gamma_t p \xrightarrow[u]{a} a[v]:q}$$

Rule $(\Gamma_t \rightarrow)$ says that $\Gamma_t p$ can perform "t-ticks" only if p can, i.e. p must be synchronisable to a clock of period t, otherwise $\Gamma_t p$ will stop. Rule $(\Gamma_{t+u} \rightarrow)$ is introduced to preserve the density lemma.

Definition 10 An agent is t-synchronous if $p \sim \Gamma_t p$.
An agent is non-synchronous if it is not t-synchronous for any t □

The definition of t-synchrony intends to capture the idea that all the "significant changes" (i.e. transitions from an a-action to a different b-action) in a t-synchronous agent occur at instants which are divisors of t. For example $p \Leftarrow a[2]:b[2]:p$ is 2-synchronous, 1-synchronous etc., but it is not 3-synchronous, 4-synchronous etc. because p cannot produce any action longer than 2.

An example of a non-synchronous agent is provided by a "bouncing ball" agent $p_n \Leftarrow a[1/n]:b[1/n]:p_{n+1}$ which changes its output at a faster and faster rate.

If we eliminate the nondeterministic guard "a(t):" from our signature, and we replace "a[t]:" by "a[1]:" (abbreviated "a:"), then all the agents which can be expressed are 1-synchronous. The set of 1-synchronous agents correspond exactly to the Synchronous CCS calculus (Milner 82), in the sense that the same set of laws holds.

Finally we can try to characterise some form of asynchronous behaviour by the following operator, which stretches by arbitrary amounts all the actions of an agent:

$$(\Delta\rightarrow) \qquad \frac{p \xrightarrow[t]{a} q}{\Delta p \xrightarrow[t+u]{a} \Delta q}$$

Definition 11 An agent is asynchronous if $p \sim \Delta p$ □

Note that this definition allows us to make a subtle distinction between non-synchronous or non t-synchronous agents (which are deterministic) and asynchronous ones (which are completely nondeterministic) and that many other behaviours lie in between.

Acknowledgements

I would like to thank Matthew Hennessy, Robin Milner and Gordon Plotkin for helpful discussions and comments.

References

(Cardelli 80) L.Cardelli: "Analog processes". Proc. 9th Symposium on Mathematical Foundations of Computer Science. Lecture Notes in C.S. n.88, Springer-Verlag.

(Hennessy 80) M.Hennessy, R.Milner: "On observing nondeterminism and concurrency". Proc. ICALP 80. Lecture Notes in C.S. n.85, Springer-Verlag.

(MacQueen 79) D.MacQueen: "Models for distributed computing". Report n.351, IRIA.

(Milner 80) R.Milner: "A calculus of communicating systems". Lecture Notes in C.S. n.92, Springer-Verlag.

(Milner 82) R.Milner: "Calculi for synchrony and asynchrony". Internal report (to appear), Dept of Computer Science, University of Edinburgh.

(Plotkin 81) G.D.Plotkin: "A structural approach to operational semantics". Report DAIMI FN-19, Dept. of Computer Science, University of Aarhus.

(Park 81) D.M.Park: "Concurrency and automata on infinite seouences". Proc. GI Conference.