Metric spaces as models for real-time concurrency

G.M. Reed and A.W. Roscoe[1]

Oxford University Computing Laboratory
7-11 Keble Road, Oxford OX1 3QD, U.K.

Abstract. *We propose a denotational model for real time concurrent systems, based on the failures model for CSP. The fixed point theory is based on the Banach fixed point theorem for complete metric spaces, since the introduction of time as a measure makes all recursive operators naturally contractive. This frees us from many of the constraints imposed by partial orders on the treatment of nondeterminism and divergence.*

1 Introduction

Real time has generally been considered to be too much an implementation matter to include in abstract models of concurrency. Most existing theories view of time is restricted to the relative order of events and to high level concepts such as 'eventually' and 'forever'. Nevertheless there are several reasons why it is desirable to have the ability to reason about real times. Most obviously, it is likely that anyone specifying a real system will wish to impose constraints on its running speed (and perhaps more detailed timing matters concerning its external communications). But perhaps more importantly from a theoretical point of view, there are several concepts commonly used in concurrent languages, such as interrupts and priority, which do not fit easily or at all into untimed models.

We therefore believe that there is a need for models of real-time parallel computation. But since it is likely to remain easier and cleaner, where possible, to do analysis in an untimed framework, it is important that a real-time model has well-understood links with an untimed theory. We have chosen to base our work on (extensions of) the theoretical version of CSP and to try to discover models that have links with that language's untimed theory. In an earlier paper [RR] we showed how the traces model [H1] could be expanded to include time. In this paper we give a timed version of the 'failures' model (including divergence) described in [BR,H2].

One of the main purposes of this paper is to show how real time gives a particularly natural measure for comparing processes: we can think of two processes as being t-alike if they are indistinguishable up to time t. This notion is easily formalised as a metric over the space of processes which provides a natural fixed point theory, seemingly with few of the disadvantages of the traditional ways of defining fixpoints in untimed models. In particular, we are able to deal with the problems of unbounded nondeterminism.

In the next section we present the model and the semantics of CSP. The difficulties of time mean that the model is quite complex and some of the semantic operators quite subtle; unfortunately time constraints for publication mean we cannot motivate or explain our definitions as thoroughly as we would have liked. (We aim to give a fuller presentation of our work in the near future.)

[1]The work reported in this paper was supported by the U.S. Office of Naval Research under grant N0014-85-G-0123.

Section 3 shows how we have treated nondeterminism and divergence and how the introduction of time frees us from difficulties found in the construction of untimed models. Finally we present our conclusions and outline some ways in which our work can be extended.

2. The timed failures-stability model for CSP

2.1 Objectives of Timed CSP

Our objective is the construction of a timed CSP model which provides a basis for the definition, specification, and verification of real-time processes with an adequate treatment of divergence and deadlock. Furthermore, we wish the model to be a "natural" extension of existing untimed models, and in particular, it should contain the timed equivalents of those CSP constructs modelled in [BHR,BR].

2.2 Abstract syntax for TCSP (Timed Communicating Sequential Processes)

We shall essentially extend the abstract syntax for untimed CSP from [BHR,BR] (with the addition of $\perp$, the diverging process which engages in no event visible to the environment). We use P, Q, R to range over syntactic processes; a, b over the alphabet Σ; X, Y over subsets of Σ; f over the set of finite-to-one functions from Σ to Σ; and F over "appropriate" compositions of our syntactic operators.

The basic requirement for analysing real-time programming languages is the ability to model time-outs and interrupts. This can be accomplished in CSP simply by the addition of a process *WAIT* t for each real number $t \geq 0$: the process which engages in no visible event to the environment and which terminates successfully after t units of time. Intuitively, *SKIP* should coincide with *WAIT* 0.

TCSP

$$\begin{aligned} P ::= \;& \perp \mid STOP \mid SKIP \mid WAIT\ t \mid (a \rightarrow P) \mid P \Box Q \mid P \sqcap Q \mid P \parallel Q \mid \\ & P\ {}_X\|_Y\ Q \mid P \mathbin{|||} Q \mid P; Q \mid P \setminus X \mid f^{-1}(P) \mid f(P) \mid \mu p.F(p) \end{aligned}$$

2.3 Timing Postulates

The following are our basic assumptions about timing in a distributed system.

(1) **A global clock.** We assume that all events recorded by processes within the system relate to a *conceptual* global clock.

(2) **A system delay constant.** We realistically postulate that a process can engage in only finitely many events in a bounded period of time. The structure of our timed models allows several parameters by which to ensure adherence with this postulate. In the current presentation, for simplicity we assume the existence of a single delay constant δ such that:

a) For each $a \in \Sigma$ and each process P, the process $(a \rightarrow P)$ is ready to engage in P only after a delay of time δ from participation in the event a.

b) A given recursive process is only ready to engage in an observable event after a delay of δ time from making a recursive call.

Inevitably, our semantics are influenced by these and other decisions about the implementation of the language. We imagine, however, that the semantics presented below could be modified to take account of different decisions, or even of a nondeterministic choice of possible implementations.

(3) **Hiding.** We wish $(a \rightarrow P)$ to denote the process that is willing at *any* time to engage in the event a and then to behave like the process P. Clearly, if $P = a \rightarrow P$, we then wish $P \setminus a = \perp$. However, consider $P = a \rightarrow STOP$ (the process that is willing to engage in a at any time ≥ 0 and then to deadlock). What do we wish $P \setminus a$ to denote?

By hiding, we remove external control. Hence, any time a process is willing to engage in an internal action, it is permitted to do so. Thus, we assume that each hidden event has taken place as soon as such event was possible. In the above example, we would wish:

$$(a \rightarrow STOP) \setminus a = WAIT\ \delta; STOP$$

(4) **Timed stability.** In untimed CSP, it is only necessary to know that a given process can or cannot diverge after engaging in a trace s; in the timed models, it is necessary to know (if the process cannot diverge after s) *when* it will again be ready to respond to the environment. This analysis leads us to consider the untimed divergence models [Ros,Brookes,OH,BR]) as providing discrete information for a given trace s ("0" cannot diverge, "∞" can diverge), and our corresponding timed model as providing continuous information ($\alpha \in [0, \infty]$ such that the process is guaranteed to be stable within α time after engaging in s). Our topological models will be based on this notion of *stability*, which is the dual of divergence.

We will model a timed CSP process as a specified set of ordered 3-tuples $(s, \alpha, \aleph)$, where s is a *timed trace* of the process, $\aleph$ is a timed refusal of the process (a subset of Σ in which the process can fail to engage over a specifed time interval), and α is the time at which the process is guaranteed to be stable after the "observation" of s and $\aleph$. If $(s, \alpha, \aleph)$ is in the process P and $\alpha < \infty$, then the next observable event in the life of the process following s may occur at any time on or after time α at the discretion of the environment, and the set of possible next events must be the same at *all* such times after stability. Clearly no event can *become* available after α.

We think of timed stability as a red light on the outside of a process which goes off when the process can make no more internal progress.

(5) **Termination and sequential composition.** The sequential composition operator treats the termination of its first argument very much as a hidden event. That is, we postulate in $P; Q$ the process P *must* terminate as soon as it can not refuse to do so. Thus, we assume that participation in the "hidden" event $\surd$ has taken place as soon as such participation was possible.

For example,

$$((a \rightarrow STOP) \Box WAIT\ 1); b \rightarrow STOP$$

is the process that is prepared to participate in the event a for the first unit of time and then deadlock, or to wait one unit of time and then participate in the event b and then deadlock; after time 1, it is no longer able to participate in the event a.

It is the above assumption about termination that allows us to model interrupts in Timed CSP.

2.4 Notation

The set (alphabet) of all communications (untimed events) will be denoted Σ. A *timed event* is an ordered pair (t, a), where a is a communication and $t \in [0, \infty)$ is the time at which it occurs. The set $[0,\infty) \times \Sigma$ of all timed events is denoted $T\Sigma$. The set of all *timed traces* is

$$(T\Sigma)^*_\leq \;=\; \{s \in T\Sigma^* \mid \text{if } (t,a) \text{ precedes } (t',a') \text{ in } s, \text{ then } t \leq t'\}.$$

If $s \in (T\Sigma)^*_\leq$, we define $\#s$ to be the length (i.e., number of events) of s and $\Sigma(s)$ to be the set of communications appearing in s (i.e., the second components of all its timed communications).

$begin(s)$ and $end(s)$ are respectively the earliest and latest times of any of the timed events in s. (For completeness we define $begin(\langle\rangle) = \infty$ and $end(\langle\rangle) = 0$.)

If $X \subseteq \Sigma$, $s\upharpoonright X$ is the maximal subsequence w of s such that $\Sigma(w) \subseteq X$; $s \setminus X = s\upharpoonright(\Sigma - X)$. If $t \in [0,\infty)$, $s\upharpoonright t$ is the subsequence of s consisting of all those events which occur no later than t. If $t \in [-begin(s),\infty)$ and $s = \langle(t_0,a_0),(t_1,a_1),\ldots,(t_n,a_n)\rangle$,

$$s+t \;=\; \langle(t_0+t,a_0),(t_1+t,a_1),\ldots,(t_n+t,a_n)\rangle\ .$$

If $s,t \in (T\Sigma)^*_\leq$, we define $s \cong t$ if, and only if, t is a permutation of s (i.e., events that happen at the same time can be re-ordered).

If $s,w \in (T\Sigma)^*_\leq$, $Tmerge(s,w)$ is defined to be the set of all traces in $(T\Sigma)^*_\leq$ obtained by interleaving s and w. (Note that this is a far more restricted set than in the untimed case, as the times of events must increase through the trace. In fact, $Tmerge(s,w)$ only contains more than one element when s and w record a pair of events at exactly the same time.)

Let $TSTAB = [0,\infty] = [0,\infty) \cup \{\infty\}$. This is the set of all "timed stability values".

Define:

$$\begin{array}{rcll} I\in TINT &=& \{\,[l(I),r(I)) \mid 0 \leq l(I) < r(I) < \infty\} & (Time\ Intervals)\\ T\in RTOK &=& \{I \times X \mid I \in TINT \wedge X \in P(\Sigma)\} & (Refusal\ Tokens)\\ \aleph\in RSET &=& \{\bigcup Z \mid Z \in p(RTOK)\} & (Refusal\ Sets) \end{array}$$

1) $\forall \aleph \in RSET$,

$$\begin{array}{rcl} \Sigma(\aleph) &=& \{a \in \Sigma \mid \exists t \in [0,\infty)\ such\ that\ (t,a) \in \aleph\}\\ I(\aleph) &=& \{t \in [0,\infty) \mid \exists a \in \Sigma\ such\ that\ (t,a) \in \aleph\}\\ begin(\aleph) &=& min(I(\aleph)),\ \forall \aleph \neq \emptyset\\ end(\aleph) &=& sup(I(\aleph)),\ \forall \aleph \neq \emptyset\\ begin(\aleph) &=& \infty, \quad for\ \aleph = \emptyset\\ end(\aleph) &=& 0, \quad for\ \aleph = \emptyset\\ \forall t \geq -begin(\aleph),\quad \aleph + t &=& \{(t'+t,a) \mid (t',a) \in \aleph\}. \end{array}$$

2) $\forall S \subseteq (T\Sigma)^*_\leq \times TSTAB \times RSET$,

$$\begin{array}{rcl} Traces(S) &=& \{s \mid \exists \alpha \in TSTAB,\ \aleph \in RSET\ such\ that\ (s,\alpha,\aleph) \in S\}\\ Stab(S) &=& \{(s,\alpha) \mid \exists \aleph \in RSET\ such\ that\ (s,\alpha,\aleph) \in S\}\\ Fail(S) &=& \{(s,\aleph) \mid \exists \alpha \in TSTAB\ such\ that\ (s,\alpha,\aleph) \in S\}\\ SUP(S) &=& \{(s,\alpha,\aleph) \mid (s,\aleph) \in Fail(S)\\ && \wedge\ \alpha = sup\{\beta \mid (s,\beta,\aleph) \in S\}\} \end{array}$$

If $\aleph \in RSET$ and $t \in [0,\infty)$, let $\aleph\upharpoonright t$ denote $\aleph \cap ([0,t) \times \Sigma)$.

2.5 The evaluation domain TM_{FS}

We formally define TM_{FS} to be those subsets S of $(T\Sigma)^*_\leq \times TSTAB \times RSET$ satisfying:

1. $\langle\rangle \in Traces(S)$
2. $(s.w, \aleph) \in Fail(S) \;\Rightarrow\; (s, \aleph \upharpoonright begin(w)) \in Fail(S)$
3. $(s, \alpha, \aleph), (s, \beta, \aleph) \in S \Rightarrow\; \alpha = \beta$
4. $(s, \alpha, \aleph) \in S \;\wedge\; s \cong w \;\Rightarrow\; (w, \alpha, \aleph) \in S$
5. $\forall t \in [0, \infty),\ \exists n(t) \in \mathbf{N}$ such that $\forall s \in Traces(S),\ (end(s) \leq t \Rightarrow \#s \leq n(t))$
6. $(s, \alpha, \aleph) \in S \;\Rightarrow\; end(s) \leq \alpha$
7. $(s, \alpha, \aleph) \in S \;\Rightarrow$ if $t > \alpha$, $t' \geq \alpha$, $a \in \Sigma$ and $w \in (T\Sigma)^*_\leq$ is such that $w = \langle(t, a)\rangle.w'$, then $(s.w, \alpha', \aleph') \in S \;\wedge\; \aleph \subseteq \aleph' \upharpoonright t \;\Rightarrow$
   $\exists \gamma \geq \alpha' + (t' - t).\ (s.(w + (t' - t)), \gamma, \aleph_1 \cup \aleph_2 \cup (\aleph_3 + (t' - t))) \in S$,
   where $\aleph_1 = \aleph' \upharpoonright \alpha$, $\aleph_2 = [\alpha, t') \times \Sigma(\aleph' \cap ([\alpha, t) \times \Sigma))$,
   and $\aleph_3 = \aleph' \cap ([t, \infty) \times \Sigma)$.
8. $(s, \alpha, \aleph) \in S \;\wedge\; (s.\langle(t, a)\rangle, \aleph) \in Fail(S) \;\wedge\; t > t' \geq \alpha \;\wedge\; t \geq end(\aleph) \;\Rightarrow (t', a) \notin \aleph$
9. $(s, \alpha, \aleph) \in S \wedge \aleph' \in RSET$ such that $\aleph' \subseteq \aleph$
   $\Rightarrow\; \exists \alpha' \geq \alpha$ such that $(s, \alpha', \aleph') \in S$
10. $(s.w, \alpha, \aleph) \in S \wedge \aleph' \in RSET$ is such that $end(s) \leq begin(\aleph') \wedge$
    $end(\aleph') \leq begin(w) \;\wedge\; (\forall (t, a) \in \aleph', (s.\langle(t, a)\rangle, \aleph \upharpoonright t) \notin Fail(S))$
    $\Rightarrow\; (s.w, \alpha, \aleph \cup \aleph') \in S$
11. $(s, \alpha, \aleph) \in S \Rightarrow$
    $\forall I \subseteq [\alpha, \infty),\ (s, \alpha, \aleph \cup (I \times \Sigma(\aleph \cap ([\alpha, \infty) \times \Sigma)))) \in S$

Though some of these axioms appear complex, each reflects a simple healthiness property. They are explained as follows.

1. Every process has initially done nothing at all.
2. If a process has been observed to communicate $s.w$ while refusing $\aleph$ then at the time when the first event of w occurred the pair $(s, \aleph \upharpoonright begin(w))$ had been observed.
3. There is only one stability value for each trace/refusal pair: the least time by which we can guarantee stability after the given observation.
4. Traces which are equivalent (are the same except for the permutation of events happening at the same times) are interchangeable.
5. The process cannot perform an infinite number of events in a finite time.
6. The time of stability is always after the end of the trace.
7. After stability the same set of events is available at all times. Furthermore the behaviour of a process after such an event does not depend on the exact time at which it was executed. Thus the trace w and the corresponding part of the refusal may be translated so as to make

the first event of w now occur at time t'. The stability value γ correponding to the translated behaviour may, in general, be greater than the obvious value because the translated behaviour may in some circumstances be possible for other reasons.

8. A stable process cannot communicate an event which it has been seen to refuse since stability.
9. If a process has been observed to communicate s while refusing $\aleph$ then it can communicate the same trace while refusing any subset of $\aleph$. This simply reflects the fact that the environment might offer it less and so have less refused. However, because less has been observed, the stability value can, in general, be greater.
10. Any set of impossible events *must* be refused if offered. Such observation does not give any extra information to the observer, so the stability value is not affected.
11. Something that is refused at one time on or after stability is refused at all such times.

2.6 The complete metric on TM_{FS}

If $S \in TM_{FS}$ and $t \in [0, \infty)$, we define

$$\begin{aligned} S(t) &= \{(s, \alpha, \aleph) \in S \mid \alpha < t\} \\ &\quad \cup\{(s, \infty, \aleph \upharpoonright t) \mid end(s) \leq t \;\wedge\; \exists \alpha \geq t.\, (s, \alpha, \aleph) \in S\}. \end{aligned}$$

This is just a standard representation of the behaviour of S up to time t: S and T have identical behaviours up to t if and only if $S(t) = T(t)$.

The complete (ultra-)metric on TM_{FS} is defined:

$$d(S_1, S_2) \;=\; inf\{2^{-t} \mid S_1(t) = S_2(t)\}$$

The completeness of this metric is a consequence of the fact that, if one of the above axioms is not satisfied by some set S of behaviours, this will show up in some finite time t. Specifically, if $S(t) = T(t)$ then T does not satisfy the axiom either.

2.7 The semantic function $\mathcal{E}$

We now define the semantic function $\mathcal{E} : TCSP \to TM_{FS}$. The definitions of most of the operators are closely modelled on their untimed counterparts. The only new operator is $WAIT\ t$, which becomes stable and able to communicate $\surd$ at time t.

Except in the case of hiding the use of the SUP operator simply reflects the fact that there can only be one stability value record for each trace/refusal pair, and some such pairs can get into the sets for several different reasons.

The definition of hiding is surprisingly simple, but the way stability is handled is rather subtle. The fact that X can be refused thoughout the behaviour ensures that hidden events occur as soon as they can. The βs in the set are all times which are demonstrably lower bounds for the time of stability, and with thought it can be seen that applying the SUP operator gives exactly the correct stability value.

Each operator has the important property that the behaviour of $F(P)$ up to time t depends only the behaviour of P up to time t. It is this which makes all operators non-expanding. See the next section for more discussion of this.

$$\mathcal{E}[\![\bot]\!] = \{(\langle\rangle, \infty, \aleph) \mid \aleph \in RSET\}$$

$$\mathcal{E}[\![STOP]\!] = \{(\langle\rangle, 0, \aleph) \mid \aleph \in RSET\}$$

$$\begin{array}{rcl}\mathcal{E}[\![SKIP]\!] & = & \{(\langle\rangle, 0, \aleph) \mid \surd \notin \Sigma(\aleph)\} \\ & & \cup\{(\langle(t,\surd)\rangle, t, \aleph_1 \cup \aleph_2) \mid t \geq 0 \wedge (I(\aleph_1) \subseteq [0,t) \wedge \surd \notin \Sigma(\aleph_1)) \\ & & \quad \wedge\, I(\aleph_2) \subseteq [t,\infty)\}\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![WAIT\,t]\!] & = & \{(\langle\rangle, t, \aleph) \mid \aleph \cap ([t,\infty) \times \{\surd\}) = \emptyset\} \\ & & \cup\{(\langle(t',\surd)\rangle, t', \aleph_1 \cup \aleph_2 \cup \aleph_3) \mid t' \geq t \wedge I(\aleph_1) \subseteq [0,t) \\ & & \quad \wedge(I(\aleph_2) \subseteq [t,t') \wedge \surd \notin \Sigma(\aleph_2)) \wedge I(\aleph_3) \subseteq [t',\infty)\}\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![a \to P]\!] & = & \{(\langle\rangle, 0, \aleph) \mid a \notin \Sigma(\aleph)\} \\ & & \cup\{(\langle(t,a)\rangle.(s + (t+\delta)), \alpha + t + \delta, \aleph_1 \cup \aleph_2 \cup (\aleph_3 + (t+\delta))) \mid t \geq 0 \\ & & \quad \wedge(I(\aleph_1) \subseteq [0,t) \wedge a \notin \Sigma(\aleph_1)) \wedge I(\aleph_2) \subseteq [t, t+\delta) \wedge (s, \alpha, \aleph_3) \in \mathcal{E}[\![P]\!]\}\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![P \Box Q]\!] & = & SUP(\{(\langle\rangle, max\{\alpha_P, \alpha_Q\}, \aleph) \mid (\langle\rangle, \alpha_P, \aleph) \in \mathcal{E}[\![P]\!] \;\wedge\; (\langle\rangle, \alpha_Q, \aleph) \in \mathcal{E}[\![Q]\!]\} \\ & & \qquad \cup\{(s, \alpha, \aleph) \mid s \neq \langle\rangle \;\wedge\; (s, \alpha, \aleph) \in \mathcal{E}[\![P]\!] \cup \mathcal{E}[\![Q]\!] \\ & & \qquad\quad \wedge\; (\langle\rangle, \aleph \upharpoonright begin(s)) \in Fail(\mathcal{E}[\![P]\!]) \cap Fail(\mathcal{E}[\![Q]\!])\})\end{array}$$

$$\mathcal{E}[\![P \sqcap Q]\!] = SUP(\mathcal{E}[\![P]\!] \cup \mathcal{E}[\![Q]\!])$$

$$\begin{array}{rcl}\mathcal{E}[\![P \| Q]\!] & = & SUP(\{(s, max\{\alpha_P, \alpha_Q\}, \aleph_P \cup \aleph_Q) \mid (s, \alpha_P, \aleph_P) \in \mathcal{E}[\![P]\!] \\ & & \qquad \wedge(s, \alpha_Q, \aleph_Q) \in \mathcal{E}[\![Q]\!]\})\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![P\; {}_X\|_Y Q]\!] & = & \{(s, max\{\alpha_P, \alpha_Q\}, \aleph_P \cup \aleph_Q \cup \aleph_Z) \mid \exists (s_P, \alpha_P, \aleph_P) \in \mathcal{E}[\![P]\!], \\ & & (s_Q, \alpha_Q, \aleph_Q) \in \mathcal{E}[\![Q]\!] \; with \; \Sigma(\aleph_P) \subseteq X \wedge \Sigma(\aleph_Q) \subseteq Y \; such \; that \\ & & s \in (s_P \; {}_X\|_Y s_Q) \wedge \Sigma(\aleph_Z) \subseteq (\Sigma - (X \cup Y))\} \\ & & \text{where} \\ & & v \; {}_X\|_Y w = \{s \in (T\Sigma)^*_{\leq} \mid s \upharpoonright (X \cup Y) = s \;\wedge\; s \upharpoonright X = v \;\wedge\; s \upharpoonright Y = w\}\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![P \,|||\, Q]\!] & = & SUP(\{(s, max\{\alpha_P, \alpha_Q\}, \aleph) \mid \exists (u, \alpha_P, \aleph) \in \mathcal{E}[\![P]\!], \; (v, \alpha_Q, \aleph) \in \mathcal{E}[\![Q]\!] \\ & & \qquad such \; that \; s \in Tmerge(u, v)\})\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![P ; Q]\!] & = & SUP(\{(s, \alpha, \aleph) \mid \surd \notin \Sigma(s) \;\wedge\; \forall I \in TINT \\ & & \qquad\quad (s, \alpha, \aleph \cup (I \times \{\surd\})) \in \mathcal{E}[\![P]\!]\} \\ & & \qquad \cup\{(s.(w + t), \alpha + t, \aleph_1 \cup (\aleph_2 + t)) \mid \; \surd \notin \Sigma(s) \;\wedge\; end(\aleph_1) \leq t \\ & & \qquad\quad \wedge\; (s.\langle(t,\surd)\rangle, \aleph_1 \cup ([0,t) \times \{\surd\})) \in Fail(\mathcal{E}[\![P]\!]) \\ & & \qquad\quad \wedge\; (w, \alpha, \aleph_2) \in \mathcal{E}[\![Q]\!]\})\end{array}$$

$$\begin{array}{rcl}\mathcal{E}[\![P \setminus X]\!] & = & SUP\{s \setminus X, \beta, \aleph) \mid \; \exists \alpha \geq \beta \geq end(s). \\ & & \qquad\qquad (s, \alpha, \aleph \cup ([0, max\{\beta, end(\aleph)\}) \times X)) \in \mathcal{E}[\![P]\!]\}\end{array}$$

$$\mathcal{E}[\![f^{-1}(P)]\!] = \{(s, \alpha, \aleph) \mid (f(s), \alpha, f(\aleph)) \in \mathcal{E}[\![P]\!]\}$$

$$\mathcal{E}[\![f(P)]\!] = SUP(\{(f(s), \alpha, \aleph) \mid (s, \alpha, f^{-1}(\aleph)) \in \mathcal{E}[\![P]\!]\})$$

$\mathcal{E}[\![\mu p.F(p)]\!]$ = The unique fixed point of the contraction mapping $\hat{C}(Q) = C(WAIT\delta; Q)$, where C is the mapping on TM_{FS} represented by F.

3. Remarks on the model

3.1 Hiding and recursion

To illustrate the intuitive appeal of topological limits in the analysis of CSP processes, consider the following untimed example.

$$\begin{array}{rclrcl} Q &=& b \rightarrow Q & P_0 &=& Q \\ P &=& a \rightarrow P & \forall n \geq 1, P_n &=& a \rightarrow P_{n-1} \end{array}$$

Recall that, by $P = a \rightarrow P$, we mean $P = \mu p.F(p)$ where $F(R) = a \rightarrow R$ is an appropriate mapping on our semantic domain.

Clearly, an observer looking at behaviours on traces of length $\leq n$ cannot distinguish between P and P_n. Hence it seems intuitive that $\lim_{n\rightarrow\infty} P_n = P$. Indeed this is the case under a complete metric structure. However, with the standard complete partial order structure for the (untimed) failures model, $\lim_{n\rightarrow\infty} P_n$ does not exist. When we move to the timed CSP models, this situation becomes critical. If an observer looking at a record of all traces completed in n units of time cannot distinguish between P_n and P, we would certainly expect $\lim_{n\rightarrow\infty} P_n = P$. In particular, we wish $\lim_{t\rightarrow\infty}(\mathit{WAIT}\, t) = \perp \neq \mathit{STOP}$.

Several authors, for example [N,Ros,BZ,GR,Rou], have considered untimed models of concurrency as metric spaces. The metrics have generally been based on equivalence up to a certain number of steps or communications in much the same way as ours has been based on indistinguishability up to a certain time. However the fact that hiding deletes communications means that a model with a metric of *visible* actions will have a discontinuous hiding operator. For example, in a topological traces model,

$$\lim_{n\rightarrow\infty}(P_n \setminus a) = Q \neq (P \setminus a) = \{\langle\rangle\} = \mathit{STOP}$$

Also, recursions are not defined unless they represent contraction maps: something which is by no means automatic, especially when hiding is involved. For example,

$$\mu p.a \rightarrow (p\backslash a) \quad \text{is undefined.}$$

Some authors have chosen to retain hidden actions in their models (often synchronisation trees). This avoids the above problems, but leads to models which are insufficiently abstract for many purposes (indeed, semantics of this type are often termed operational).

We had none of these problems in constructing the present model because time cannot be hidden, and yet we would expect an observer (with a clock) to be able to observe it. There is no operator (even hiding) whose behaviour up to time t depends on the behaviours of its operands after t: no reasonable operator can be expected to see into the future. Thus every operator represents at worst a nonexpansive function of the metric space. (Note that a nonexpansive function is always continuous.)

Consider P, Q, and P_n as defined above in the timed failures-stability model (with the appropriate change in P_n to reflect the delay induced by each recursive call in P).

$$\begin{array}{rclrcl} Q &=& b \rightarrow Q & P_0 &=& Q \\ P &=& a \rightarrow P & \forall n \geq 1,\ P_n &=& a \rightarrow (\mathit{WAIT}\ \delta\ ;\ P_{n-1}) \end{array}$$

Now, $\lim_{n\rightarrow\infty} P_n = P$ and $\lim_{n\rightarrow\infty}(P_n \setminus a) = P \setminus a = \perp \neq \mathit{STOP}$.

We make every recursion into a contraction by observing that, realistically, a recursion will always take a small amount of time to unwind. For example:

$$\begin{array}{rcl} \mu p.p &=& \mathit{fix}(\hat{C}), \quad \text{where } \hat{C}(Q) = \mathit{WAIT}\ \delta; Q \\ &=& \perp \end{array}$$

3.2 Divergence

The timed failures-stability model (like the timed stability model of [RR]) differs from previous CSP models relevant to divergence in that $(s, \infty) \in \mathit{Stab}(P)$ does not imply that $(s.w, \infty) \in \mathit{Stab}(P)$ for all traces w. That is, just because a process *may* diverge after engaging in a given trace, it does not mean that some time later after extending the trace, the process might not again become stable. For example, let $P = a \rightarrow P$ and consider the process $R = (P \setminus a) \sqcap (b \rightarrow (b \rightarrow \mathit{STOP}))$. Both $(\langle\,\rangle, \infty)$ and $(\langle(t, b)\rangle, t + \delta) \in \mathit{Stab}(R)$ (for any $t \geq 0$). This process can diverge on the empty trace; however, once we observe a b, we know that we are safe. Although it is possible to modify our model to conform to the untimed models in this regard, we choose to allow the finer distinction of CSP processes made possible by the topological structure of our evaluation domain.

These distinctions were not made in the failures-divergence model [BR,H2] because of the use of least fixed points to define recursions. In any model where the partial order is based on nondeterminism or definedness ($P \sqsubseteq Q$ iff Q is more deterministic than P), the least fixed point of $\mu p.p$ (operationally, a simply diverging process) is the most nondeterministic process. We are thus forced to identify the diverging process with one that can do anything (including diverge). This is closely related to the philosophy of the Smyth powerdomain (the powerdomain of dæmonic nondeterminism).

The failures-divergence model's axioms essentially state that we cannot specify anything about a process' behaviour after the possibility of divergence. This is tantamount to saying that we will never be prepared to accept, for any practical purpose, a process that can diverge. Some authors have disliked being forced to take this very strict view, and would have prefered a theory more like that of the timed model. This has lead them to use alternative fixed point theories such as *optimal* fixed points [Broy] (usually using more than one partial order). By using a metric space we have been able to achieve the same effect more easily.

3.3 Infinite hiding

The reader will note that the axiom of bounded nondeterminism from the failures model in [BR] is not included in our model:

$$(\forall Y \in p(X),\ (s, Y) \in S) \ \Rightarrow\ (s, X) \in S$$

or, in a timed context,

$$\begin{aligned} I \in \mathit{TINT} \ \wedge\ X \in P(\Sigma)\ \mathit{such\ that}\ (\forall Y \in p(X),\ \exists \alpha\ \mathit{such\ that}\ (s, \alpha, \aleph \cup (I \times Y)) \in S) \\ \Rightarrow\ \exists \alpha'\ \mathit{such\ that}\ (s, \alpha', \aleph \cup (I \times X))) \in S \end{aligned}$$

Operationally, a process can be said to be boundedly nondeterministic if, at each point, it has only finitely many internal choices which may affect its future behaviour. It has generally proved much easier to model boundedly nondeterministic processes than to consider the more general case. In partial orders one often takes limits by intersecting the nondeterministic choices that can be made in an increasingly deterministic sequence of processes. Where processes have an infinite number of options it is possible to construct such an infinite sequence with empty intersection. (For example, suppose P_i is a process which nondeterministically communicates any integer $j \geq i$ on its first step.) It becomes necessary to introduce compactness assumptions, such as the axiom above, which are not natural in every circumstance.

Since the convergence in our metric space is independent of nondeterminism we have the choice whether to have such an axiom or not.

‘Non-compact’ unbounded nondeterminism can enter models of concurrency through declining to ignore what might happen after divergence and also through assumptions such as fairness. But the clearest source of unbounded nondeterminism is infinite hiding, where an infinite set of external choices are made internal. It has usually been necessary to exclude $P\backslash X$ (X infinite) from CSP because of this problem.

For example, let Q denote the process which is prepared to input any odd integer n and then to participate in the event b and then to output $n+1$. (See [H2] for the obvious generalization to our syntax.)

$$Q \ = \ n : Odds \rightarrow (b \rightarrow (n+1 \rightarrow STOP))$$

Now, Q is certainly definable in the complete partial order failures-divergence model as well as the timed model. However, $Q \setminus X$ is not allowed in the failures-divergence model, since $\{(b, X) \mid X \in p(Evens)\} \subseteq Q \setminus Odds$, but $(b, Evens) \notin Q \setminus Odds$.

Given that in the timed model we have chosen not to ignore what might happen after possible divergence, it might appear that it is possible to obtain unbounded nondeterminism from only finite hiding. For example, consider $P_0\backslash a$, where

$$\begin{array}{rcl} P_n & = & (a \rightarrow P_{n+1} \\ & & \Box \\ & & b \rightarrow (n \rightarrow STOP)) \end{array}$$

Such a process can be defined in the topological models by use of infinite mutual recursion (easily added to the semantics). After communicating a b, this process can apparently choose to communicate any natural number, but cannot refuse them all. It would be problematic in an untimed model, but is not in our one because only finitely much nondeterminism is exhibited up to any finite time (as only finitely many hidden ‘a’s will have occurred).

It will be consistent to bring in an axiom of bounded nondeterminism if, and only if, we do not want to model any operator which, like infinite hiding, has the potential of introducing infinitely many choices in a finite time.

3.4 Choice between waiting and participation

Let us postulate the effect of the environment being given the choice of participating in a given process or of waiting. For example, $P = ((a \rightarrow STOP) \Box WAIT\ 1)$ offers the environment the initial choice of participating in the event a or of terminating successfully after 1 second. Again, what do we wish $P \setminus a$ to denote? Since we have assumed that the hidden event takes place as soon as possible, we would expect (under the assumption that $\delta < 1$):

$$((a \rightarrow STOP) \Box WAIT\ 1) \setminus a \ = \ WAIT\ \delta; STOP$$

Similarly, we would wish:

$$\begin{array}{rcl} (((a \rightarrow SKIP) \Box WAIT\,1)\ ;\ b \rightarrow STOP) \setminus a & = & WAIT\,\delta\ ;\ b \rightarrow STOP \\ (((a \rightarrow STOP) \Box WAIT\,1)\ ;\ b \rightarrow STOP) \setminus a & = & WAIT\,\delta\ ;\ STOP \end{array}$$

Note from examination of for the above processes, it is clear that to achieve our intuitive semantics, when defining $P \setminus X$ in the context of $\Box$, we must be able to exclude some traces in P

from consideration based on information about their possible refusals prior to stability. In fact, the situation is even more complicated. Consider:

$$\begin{aligned} P_1 &= ((a \rightarrow STOP)\Box(b \rightarrow STOP)) \sqcap (a \rightarrow c \rightarrow STOP) \\ P_2 &= ((a \rightarrow c \rightarrow STOP)\Box(b \rightarrow STOP)) \sqcap (a \rightarrow STOP) \end{aligned}$$

Such processes would seem free of our current concern since they do not involve either hiding or delays. However, let

$$Q = (WAIT\,1\Box(b \rightarrow STOP))\ ;\ a \rightarrow c \rightarrow STOP$$

Operationally, as indicated in our assumption (5) from section 2.3, we would expect:

$$(P_1 \parallel Q) \setminus b \neq (P_2 \parallel Q) \setminus b$$

In particular, we would expect:

$$\langle(1,a)(1+\delta,c)\rangle \in Traces((P_1\|Q) \setminus b)\ \ but\ \ \langle(1,a)(1+\delta,c)\rangle \notin Traces((P_2\|Q) \setminus b)$$

Hence, in our timed failures-stability model, we must distinguish between processes such as P_1 and P_2. Note that it is impossible to make such a distinction based on what a process can refuse *after* a given trace has been achieved. Hence, it is necessary not only to record refusals on a given trace prior to stability, but also to record what refusals were involved in the state changes which led to the final state witnessed by the trace. This seems to be a crucial issue in achieving a successful semantics for real-time parallel languages. The distributivity of the hiding operator over $\sqcap$ depends on the subtle resolution of this issue.

It is the fact that we record refusals throughout a process' history that has allowed us to dispense with the 'hatted' events of [RR]. (There, $\hat{a}$ represented the communication of an event a at the instant when it became available. These communications were essential for the correct definition of hiding and sequential composition.) For it is now apparent that an event has just become available if it communicated immediately after it has been refused.

4. Conclusions

We have seen that using time as the basis of a metric space allows one to be freed from the constraints of complete partial orders without losing generality or abstractness. The reader should compare the model presented here to timed models for concurrency based on complete partial orders in [J,KSRGA,Bo]. The main difficulty that remains (under both structures) is that we still cannot describe properties (such as fairness) which are only detectable over infinite time spans.

The ideas behind our model are conceptually reasonably straightforward: a process is just modelled by the records of experiments that an observer can carry out on it (communications accepted and communications refused). The fact that refusals must be recorded all the way through a trace is, as was explained in 3.4, a consequence of the way timed processes interact: in some sense they can perform more delicate experiments on each other than untimed processes. Refusals only after traces are no longer properly compositional. Other authors [J,Bo] have remarked on this and suggested or introduced similar solutions (based on partial orders rather than metric spaces). Phillips [P] has studied the corresponding untimed congruence.

The reader may note that the *traces* of $P\backslash a$ depend in a crucial way on the (timed) *failures* of P. (This is clearly illustrated by the second example of 3.4.) An immediate consequence of this is that the traces of a CSP program modelled in the timed traces model [RR] may be a strict superset of the traces that can be extracted from the failures model. Because the earlier did not contain enough information to accurately predict the possible traces of $P\backslash a$, it was necessary to give an upper bound. This is unlike the case with untimed CSP, where the traces predicted by the failures semantics are always precisely those predicted by the trace semantics. (The problems with hiding in the timed trace semantics arise from its failure to distinguish $P \sqcap Q$ and $P \Box Q$: in reality the traces of $(P \Box Q)\backslash a$ may be a strict subset of those of $(P \sqcap Q)\backslash a$.)

We believe that a version of our present model with stablity omitted is the simplest equivalence which is a full and natural congruence with respect to all the usual CSP operators. This congruence does *not* exist in the untimed case, since there consideration of divergence is necessary if one is to consider failures.

Thus our inclusion of stability in the present paper has been in some sense optional. Aside from the well-known arguments for wishing to distinguish a deadlocked process from a diverging one, there is another good reason for our inclusion of it. It is our declared aim to build a hierarchy of models, both timed and untimed, with well-understood links between them. Stability allows natural links to be formed with the failures-divergence model, since the liveness properties predicted by that model can be inferred from the time of stability on.

We expect to be able to exploit this link by using reasoning in two simpler models (the timed stablilty model of [RR] and the failures-divergence model) to infer total correctness properties in the timed failures-stablity model (where we expect detailed computations to be rather complex). Such reasoning can be expected to be sufficient when proving simple properties of processes that do not depend on the details of timed interaction to achieve 'untimed' correctness.

The facts that our model is a complete metric space and all recursions are contraction mappings makes it a natural vehicle for correctness proofs using the form of recursion induction described in [Ros,RR]. (A predicate that represents a non-empty closed subset and which is preserved by a recursion must contain the unique fixed point.) The introduction of stability seems to enhance the range of useful predicates which represent closed sets, since it (to a limited extent) allows us to look into the future.

The semantics we gave for CSP is by no means the only possible one that is reasonable, for any such semantics must make specific timing assumptions about the language and its implementation. We assumed that all events take exactly δ, while in practice each event a might take its own duration $\delta(a)$ or even a time chosen nondeterministically from some interval. In the last case our new-found ability to cope with unbounded nondeterminism would be essential. We also assumed that none of the operators except recursion consumed any time by running (i.e, there was never any setting-up or "overhead" time). Also both the parallel operators we gave were *true* parallel operators, in that the time taken by the two operands was not summed: one might well need time-sliced pseudo-parallel operators in applications. In a particular application one will have to decide on the "right" timed semantics, but there should never be any problem in accomodating it in the model TM_{FS}. This is a topic for further research.

As indicated earlier, we intend shortly to give a fuller presentation of the timed failures-stability model and its CSP semantics. Some of the above issues will be investigated further, and also others such as full abstraction and the nondeterminism partial order.

5. References

[Bo] A. Boucher, *A time-based model for occam,* Oxford University D.Phil. thesis 1986.

[Brookes] S.D. Brookes, *A model for communicating sequential processes,* Oxford University D.Phil. thesis 1983.

[Broy] M. Broy, *Fixed point theory for communication and concurency*, TC2 Working Conference on Formal Description of Programming Concepts II, Garmisch, 1982.

[BHR] S.D. Brookes, C.A.R. Hoare and A.W. Roscoe, *A theory of communicating sequential processes*, JACM 31 (1894), 560-599.

[BR] S.D. Brookes and A.W. Roscoe, *An improved failures model for communicating processes*, Proceedings of the Pittsburgh Seminar on Concurrency, Springer LNCS 197 (1985).

[BZ] J.W. de Bakker and J.I. Zucker, *Processes and the denotational semantics of concurrency*, Information and Control 54 (1982), 70-120.

[GR] W.G. Golson and W.C. Rounds, *Connections between two theories of concurrency: metric spaces and synchronisation trees*, Information and Control 57 (1983), 102-124.

[H1] C.A.R. Hoare, *A model for communicating sequential processes*, On the construction of programs CUP (1980), 229-248.

[H2] C.A.R. Hoare, *Communicating sequential processes*, Prentice-Hall International, 1985.

[J] G. Jones, *A timed model for communicating processes*, Oxford University D.Phil thesis, 1982.

[KSdRGA-K] R. Koymans, R.K. Shyamasundar, W.P. de Roever, R. Gerth and S. Arun-Kumar, *Compositional semantics for real-time distributed computing* Faculteit der Wiskunde en Natuurwetenschappen, Katholieke Universiteit, Nijmegen, Technical report 68, 1985.

[N] M. Nivat, *Infinite words, infinite trees, infinite computations*, Foundations of Computer Science III (Math. Centre Tracts 109, 1979), 3-52.

[OH] E.R. Olderog and C.A.R. Hoare, *Specification-oriented semantics for communicating processes*, Springer LNCS 154 (1983), 561-572.

[P] I. Phillips, *Refusal testing*, Proceedings of ICALP'86, Springer LNCS 226 (1986), 304-313.

[Ros] A.W. Roscoe, *A mathematical theory of communicating processes*, Oxford University D.Phil thesis 1982.

[Rou] W.C. Rounds, *Applications of topology to the semantics of communicating processes*, Proceedings of the Pittsburgh Seminar on Concurrency, Springer LNCS 197 (1985).

[RR] G.M. Reed and A.W. Roscoe, *A timed model for communicating sequential processes*, Proceedings of ICALP'86, Springer LNCS 226 (1986), 314-323.