LFCS

Laboratory for Foundations of Computer Science
Department of Computer Science - University of Edinburgh

Timing Concurrent Processes

by

Chris Tofts

Timing Concurrent Processes

LFCS Report Series

ECS-LFCS-89-103
(also published as CSR-322-89)

December 1989

LFCS
Department of Computer Science
University of Edinburgh
The King's Buildings
Edinburgh EH9 3JZ

Timing Concurrent Processes.

Chris Tofts

December 18, 1989

Abstract

In this paper two timed calculi for concurrent processes are presented. They differ in their concepts of time cost. We will show that they are strongly related and present a number of natural equivalences and orders over timed processes. We present a number of examples including a temporal analysis of an alternating bit protocol.

1 Introduction.

Algebraic approaches to the study of concurrent systems [Mil80, Mil86, Hoa85, Ber85] and others, have proved effective. Unfortunately they either give no account of the passage of time, and its passage is explained in the form of synchrony[Mil83]. The temporal properties of concurrent processes give an insight into some interesting aspects of concurrent programming. There have been some attempts to provide a formalism within which these concepts can be expressed [Ros86, Koy83, Koy87, Jef]. Some of these assume synchrony which results in some of the more interesting temporal properties of process being inexpressible. In a previous paper [Tof88] we provided an extension to CCS which admitted a notion of timing. That approach was unsatisfactory in that the account it gave of time was somewhat eccentric. Processes could only evolve simultaneously by communication, time and action were interleaved otherwise. In order to give a fuller account of time for asynchronous processes we need to just let time pass, and observe what the processes produce and when they produce it.

In order to give this fuller account the state transitions have been split into two orthogonal parts; one part is our normal notion of action (which can be regarded as computation) and the other part is the passage of time. There are sound reasons for making this separation. Computation involves energy change and there is a result of quantum mechanics which states that energy changes and time cannot be measured simultaneously [Dir58, Sch82]. Thus it seems reasonable when producing models of time and computation not to permit the simultaneous observance of the two activities.

We therefore assume that actions have no duration; although if we wish to construct actions with duration we can in the manner of [Cas85].

2 Language Definition.

We define a timed extension of the language CCS [Mil80] as follows.

Let Λ be a set of (atomic action) symbols not containing τ or ϵ, and let $Act = \Lambda \cup \{\tau\}$. We also have times t taken from one of the following: positive integers, positive rationals or positive reals, representing the divisions of time. We assume a complementation bijection $\text{-}. : Act \rightarrow Act$ which is its own inverse. The letter λ ranges over Λ, the letter μ over Act, and S over *relabelling functions*, i.e. those $S : Act \rightarrow Act$ such that $S(\overline{\mu}) = \overline{S(\mu)}$ and $S(\mu) \neq \tau$, unless $\mu = \tau$. The languages wTCCS (weakly timed CCS) and sTCCS (strongly timed CCS) consist of an infinite set Var of variables ranged over by X and Y, a constant symbol Nil, unary function symbols $\mu.$, $\backslash\lambda$, $[S]$, and FIX_X ($X \in Var$), and the binary function symbols $+$ and $|$. With unary functions $\mu.$ taking a process and prefixing the process by the given action. In wTCCS the function symbol [], denotes a function which takes a process and a time and yields a process prefixed by that amount of time. The function () takes a process and a time (not zero), and returns the process prefixed by that time, along with the unary function δ which returns a delayed process in the calculus sTCCS.

The set P.of wTCCS-expressions ranged over by P is the set given by the following definition:

$$P \ ::= \ Nil \mid \mu.P \mid [t]P \mid P \mid P \mid P + P \mid FIX_X P \mid X \mid P\backslash L \mid P[S].$$

The set P of sTCCS-expressions ranged over by P is the set given by the following definition:

$$P \ ::= \ Nil \mid \mu.P \mid (t)P \mid \delta P \mid P \mid P \mid P + P \mid FIX_X P \mid X \mid P\backslash L \mid P[S].$$

(It will be clear from the context which version of the timing system is being used.) The intention of the time action prefix is as follows. In wTCCS, $[t]P$ means that after a period of time t' where $t' \geq t$ the process P is reached; this is very similar to the *wait* introduced into CSP by Roscoe and Reed [Ros86]. On the other hand, the system sTCCS represents separation of delay from the initial timing so $(t)P$ represents a process that will becomes P in precisely a period of time t. The δ operator providing a way of introducing delays to allow for synchronisation.

2.1 Derivation Laws.

The action-evolution of a process can be derived from the operational rules presented in Figure 8-1. The temporal evolutions of wTCCS are derived using the operational rules presented in Figure 8-2. The alternative set of rules presented in Figure 8-3, give the temporal evolutions for sTCCS. The transition relations between processes is the least set of transitions satisfying the set of action laws plus the appropriate set of temporal laws.

Definition 2.1 *We define the operator $\dot{-}$ on times as follows;*

ACT: $\dfrac{}{\mu.P \xrightarrow{\mu} P}$

SUM0: $\dfrac{P \xrightarrow{\mu} P'}{P+Q \xrightarrow{\mu} P'}$ SUM1: $\dfrac{Q \xrightarrow{\mu} Q'}{P+Q \xrightarrow{\mu} Q'}$

COM0: $\dfrac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q}$ COM1: $\dfrac{Q \xrightarrow{\mu} Q'}{P \mid Q \xrightarrow{\mu} P \mid Q'}$

COM2: $\dfrac{P \xrightarrow{\lambda} P' \quad Q \xrightarrow{\overline{\lambda}} Q'}{P \mid Q \xrightarrow{\tau} P' \mid Q'}$

REL: $\dfrac{P \xrightarrow{\mu} P'}{P[S] \xrightarrow{S(\mu)} P'[S]}$

RES: $\dfrac{P \xrightarrow{\mu} P'}{P\backslash L \xrightarrow{\mu} P'\backslash L} \quad \mu, \overline{\mu} \notin L$

REC: $\dfrac{P_j\{\tilde{P}/\tilde{X}\} \xrightarrow{\mu} P_j'}{FIX_j\{X_i = P_i;\ i \in I\} \xrightarrow{\mu} P_j'}$

Figure 1: Operational rules for wTCCS and sTCCS

$$t \dot{-} t' = \begin{cases} t - t' \ \textit{when } t' \leq t \\ 0 \ \textit{otherwise,} \end{cases}$$

where $-$ *is the usual subtraction operator.*

2.2 An Example.

In wTCCS the following process deadlocks,

$$(a.b.nil \mid \overline{a}.[5]\overline{b}.nil)\backslash a, b.$$

The process has the following derivation,

$$(a.b.nil \mid \overline{a}.[5]\overline{b}.nil)\backslash a, b \xrightarrow{\tau} (b.nil \mid [5]\overline{b}.nil)\backslash a, b.$$

In the parallel composition above, the right hand process now requires a period of time of length at least 5 to pass before it will undergo any further activity; the left hand process requires a matching $\overline{b}$ action to proceed but will not delay. No further action is therefore possible.

So actions not separated by delays must be performed sequentially and immediately in time; failure to do so leads to a deadlock.

TIME: $\dfrac{}{[t]P \overset{t'}{\rightsquigarrow} [t \dot{-} t']P}$ TRANS: $\dfrac{P \overset{t}{\rightsquigarrow} P' \quad P' \overset{s}{\rightsquigarrow} P''}{P \overset{s+t}{\rightsquigarrow} P''}$

COM-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P' \quad Q \overset{t}{\rightsquigarrow} Q'}{P \mid Q \overset{t}{\rightsquigarrow} P' \mid Q'}$

SUM-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P' \quad Q \overset{t}{\rightsquigarrow} Q'}{P + Q \overset{t}{\rightsquigarrow} P' + Q'}$

ACT-TIME: $\dfrac{P \xrightarrow{\mu} P'}{[0]P \xrightarrow{\mu} P'}$

REL-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P'}{P[S] \overset{t}{\rightsquigarrow} P'[S]}$

RES-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P'}{P \backslash L \overset{t}{\rightsquigarrow} P' \backslash L}$

REC-TIME: $\dfrac{P_j\{\tilde{P}/\tilde{X}\} \overset{t}{\rightsquigarrow} P'_j}{FIX_j\{X_i = P_i;\ i \in I\} \overset{t}{\rightsquigarrow} P'_j}$

Figure 2: Temporal rules for wTCCS

TIME: $\dfrac{}{(t)P \overset{t'}{\rightsquigarrow} (t-t')P}$ for $t > t'$.

TIME1: $\dfrac{}{(t)P \overset{t}{\rightsquigarrow} P}$

TRANS: $\dfrac{P \overset{t}{\rightsquigarrow} P' \quad P' \overset{s}{\rightsquigarrow} P''}{P \overset{s+t}{\rightsquigarrow} P''}$

DELAY: $\dfrac{}{\delta P \overset{t}{\rightsquigarrow} \delta P}$

UN-DELAY: $\dfrac{P \xrightarrow{\mu} P'}{\delta P \xrightarrow{\mu} P'}$

COM-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P' \quad Q \overset{t}{\rightsquigarrow} Q'}{P \mid Q \overset{t}{\rightsquigarrow} P' \mid Q'}$

SUM-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P' \quad Q \overset{t}{\rightsquigarrow} Q'}{P + Q \overset{t}{\rightsquigarrow} P' + Q'}$

REL-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P'}{P[S] \overset{t}{\rightsquigarrow} P'[S]}$

RES-TIME: $\dfrac{P \overset{t}{\rightsquigarrow} P'}{P \backslash L \overset{t}{\rightsquigarrow} P' \backslash L}$

REC-TIME: $\dfrac{P_j\{\tilde{P}/\tilde{X}\} \overset{t}{\rightsquigarrow} P'_j}{FIX_j\{X_i = P_i;\ i \in I\} \overset{t}{\rightsquigarrow} P'_j}$

Figure 3: Temporal rules for sTCCS

EXT-DELAY-1: $\dfrac{P \overset{t}{\rightsquigarrow} P'}{P|Q \overset{t}{\rightsquigarrow} P'|Q}$

EXT-DELAY-2: $\dfrac{Q \overset{t}{\rightsquigarrow} Q'}{P|Q \overset{t}{\rightsquigarrow} P|Q'}$

Figure 4: Delay Rules (DR).

This would seem to imply that it would be useful to add *temporal* evolution laws as in figure 8-4.

Proposition 2.2 *Let P be a process in wTCCS in which all actions are guarded by time actions, and all Nil processes are guarded by* [0]. *Then P has exactly the same evolutions in wTCCS + DR.*

> **Proof:** Since all actions are guarded by time prefixes, we never reach a state where we must infer (by ACT alone) that $P \xrightarrow{\lambda} P'$. In other words for all processes P we can always infer $P \overset{t'}{\rightsquigarrow} P'$ for t' larger than some arbitrary t, thus we can never reach a state where in a parallel composition $P \mid Q$, either P or Q can only evolve through an action, therefore we can infer any evolution that we could infer with the addition of the rules DR.

Proposition 2.3 *Every process P in wTCCS is directly equivalent to a process $\tilde{P}$ in sTCCS, given by replacing every weak temporal evolution of P with a strong temporal evolution plus a delay in $\tilde{P}$. In other words we write $(t)\delta\tilde{Q}$ for every occurrence of $[t]Q$.*

> **Proof:** It is clear from the definition of wTCCS and sTCCS that any action evolution of P is directly matched by $\tilde{P}$. Thus we need only consider the temporal evolutions. Given $P \overset{t}{\rightsquigarrow} P'$ then we can also infer $\tilde{P} \overset{t}{\rightsquigarrow} \tilde{P}'$. There are two cases,
>
> 1. t is less than the time for any time action in the current derivation to reach zero. In this case the time rule is identical in both cases.
> 2. t is greater then the time for at least one time action in the current derivation to reach zero. Then we have to use both the TIME and the EXT-DELAY rule to infer that $\tilde{P} \overset{t_1}{\rightsquigarrow} \tilde{P}''$ and $\tilde{P}'' \overset{t_2}{\rightsquigarrow} \tilde{P}'$ with t_1 the least time to the exhaustion of any temporal action and t_2 an arbitrary time introduced by the delay such that $t = t_1 + t_2$ combining these two evolutions we get $\tilde{P} \overset{t}{\rightsquigarrow} \tilde{P}'$.

Notation: we will use the abbreviation P^t to represent the process P' when $P \overset{t}{\rightsquigarrow} P'$, whenever it is inconvenient to introduce a new process name for all the timed intermediates of a process.

2.3 Temporal Deadlock.

The *Nil* or *deadlocked* process is one which is capable of no action. There is an equivalent temporal process which can always permit the passage of time, but never produces an action. The following are the simplest definitions for both wTCCS and sTCCS respectively:

$$[0]Nil,$$
$$\delta Nil.$$

There are, however, an infinite variety of processes which are equivalent to this temporal nil. For any P in wTCCS and Q in sTCCS, the following processes are equivalent to the temporal nil:

$$[0][1]P,$$
$$\delta(1)Q,$$

since in neither case can we infer a direct action, and so we cannot remove the leading [0] or the δ operators.

2.4 Deadlock.

The Nil processes acts like a *deadlock* [*Ber*85] with respect to the continuing temporal evolution of the system. If we examine the derivation laws $COM-TIME$ and $SUM-TIME$, we observe that the following are true for any non-zero t.

$$Nil + (t)P \equiv Nil$$
$$Nil \mid (t)P \equiv Nil$$

Thus once we have an unguarded Nil process in any leading binary term *all* further temporal evolution is blocked. Any composite process is stopped immediately, the process cannot evolve further in time. We can use this property to compare the initial time behaviour of any processes.

Definition 2.4 *Given any equivalence [order] we can construct the time pre-fix equivalence [order] simply by composing in parallel both processes we wish to show equivalent[related] with the process* $(t)Nil$.

This is motivated from the proceeding observation. After a period of time t has passed the processes $(t)Nil \mid P$ and $(t)Nil \mid Q$ become equivalent to Nil; until t has passed they exhibit all the possible behaviour of the respective processes within that period.

2.5 Time Actions.

There is an interesting distinction between weak and strong time actions.

Proposition 2.5 *Consider two successive time actions in both wTCCS and sTCCS then;*

1. *if* $t_2 > 0$ *then* $[t_1][t_2]P$ *has different derivations to* $[t_1 + t_2]P$,

2. $(t_1)(t_2)P$ *has identical derivations as* $(t_1 + t_2)P$.

Proof:

1. for $t_2 > 0$ this process is the temporal nil, and if P has at least one immediate action then $[t_1 + t_2]P$ is not a temporal nil.
2. directly from the inference rules for any $t < t_1 + t_2$ we can only infer that $(t_1)(t_2)P \stackrel{t}{\rightsquigarrow} P'$ with $P' \equiv (t_1 + t_2 - t)P$ and $(t_1 + t_2)P \stackrel{t}{\rightsquigarrow} P'$, and furthermore for $t = t_1 + t_2$ we can infer $(t_1)(t_2)P \stackrel{t}{\rightsquigarrow} P$ and $(t_1 + t_2)P \stackrel{t}{\rightsquigarrow} P$.

3 Strong Time Sensitive Pre-order.

It is possible, much as for CCS, to produce two different basic notions of equivalence (strong and weak), which are based on orders with respect to time. The first of which requires that the time taken after any action be matched directly by the time taken after a similar action in the other process, not that the total time taken to go from one state to another via an action is greater.

Definition 3.1 *(Strong pre-order) We will say that* P *is* faster *than* Q *iff there exists a relation* R *with* $(P,Q) \in R$ *iff for all* $\mu \in Act$ *and for all times* t*;*

1. *if* $P \stackrel{\mu}{\rightarrow} P'$ *then there exists* Q' *such that* $Q \stackrel{\mu}{\rightarrow} Q'$ *and* $(P',Q') \in R$,
2. *if* $P \stackrel{t}{\rightsquigarrow} P'$ *then there exists* Q', t' *such that* $Q \stackrel{t'}{\rightsquigarrow} Q'$ *and* $(P',Q') \in R$ *and* $t' \geq t$,
3. *if* $Q \stackrel{\mu}{\rightarrow} Q'$ *then there exists* P' *such that* $P \stackrel{\mu}{\rightarrow} P'$ *and* $(P',Q') \in R$,
4. *if* $Q \stackrel{t}{\rightsquigarrow} Q'$ *then there exists* P', t' *such that* $P \stackrel{t'}{\rightsquigarrow} P'$ *and* $(P',Q') \in R$ *and* $t' \leq t$.

The relation R *is called s* strong pre-order.

Proposition 3.2 *If* R, R' *and* R_i *for* $i \in I$ *are all* strong pre-orders *then so are;*

1. Id_P,
2. RR',
3. $\bigcup_{i \in I} R_i$.

Proof:

Parts 1 and 3 are obvious so we will prove part 2.

Given $P\ RR'\ Q$ then there exists S such that $P\ R\ S$ and $S\ R'\ Q$. Since R and R' are both *strong pre-orders* from the definition.

If $P \xrightarrow{\mu} P'$ then there exists S' such that $S \xrightarrow{\mu} S'$ and $P'\ R\ S'$. Similarly from the second equivalence there is a Q' such that $Q \xrightarrow{\mu} Q'$ and $S'\ R'\ Q'$.

Thus for $P \xrightarrow{\mu} P'$ then there exists Q' such that $Q \xrightarrow{\mu} Q'$ and $P'\ RR'Q'$.

If $P \overset{t}{\rightsquigarrow} P'$ then there exists S' and t' such that $S \overset{t'}{\rightsquigarrow} S'$ with $P'\ R\ S'$ and $t' \geq t$. Similarly from the second equivalence there is a Q' and t'' such that $Q \overset{t''}{\rightsquigarrow} Q'$ with $S'\ R'\ Q'$ and $t'' \geq t'$.

Thus for $P \overset{t}{\rightsquigarrow} P'$ then there exists Q' and t'' such that $Q \xrightarrow{t''} Q'$ with $P'\ RR'\ Q'$ and $t'' \geq t$.

The symmetric cases can be proved similarly. □

Definition 3.3 *A functional $\mathcal{F}$ on binary relations $R \subseteq P \times P$ such that $(P,Q) \in \mathcal{F}(R)$ iff for all $\mu \in Act$ and for all time t:*

1. *if $P \xrightarrow{\mu} P'$ then there exists Q' such that $Q \xrightarrow{\mu} Q'$ and $(P',Q') \in R$,*
2. *if $P \overset{t}{\rightsquigarrow} P'$ then there exists Q', t' such that $Q \overset{t'}{\rightsquigarrow} Q'$ and $(P',Q') \in R$ and $t' \geq t$,*
3. *if $Q \xrightarrow{\mu} Q'$ then there exists P' such that $P \xrightarrow{\mu} P'$ and $(P',Q') \in R$,*
4. *if $Q \overset{t}{\rightsquigarrow} Q'$ then there exists P', t' such that $P \overset{t'}{\rightsquigarrow} P'$ and $(P',Q') \in R$ and $t' \leq t$.*

Proposition 3.4

- *$\mathcal{F}$ is monotonic,*
- *R is a strong pre-order iff $R \subseteq \mathcal{F}(R)$.*

Proof: both parts directly from the definition of $\mathcal{F}$.

Definition 3.5 *Call* $\geq \equiv \ \{ \bigcup R \mid R$ *is a* strong pre-order $\}$.

Proposition 3.6 $\geq$ *is the largest strong pre-order.*

Proposition 3.7 $\geq$ *is the fixed point of* $\mathcal{F}$.

Proof: $\geq$ is a strong pre-order hence $\geq \subseteq \mathcal{F}(\geq)$,
$\mathcal{F}$ is monotonic thus $\mathcal{F}(\geq) \subseteq \mathcal{F}(\mathcal{F}(\geq))$,
(In other words $\mathcal{F}(\geq)$ is a pre-fixed point of $\mathcal{F}$.)
but $\geq$ contains all pre-fixed points of $\mathcal{F}$ hence $\mathcal{F}(\geq) \subseteq \geq$ □

Proposition 3.8 *If $P \geq Q$ for P, Q in wTCCS; then for their respective CCS equivalents $\hat{P} \sim \hat{Q}$. Two process can only be temporally related if their computational behaviour is identical.*

Proposition 3.9 Strong Pre-Order *is substitutive with respect to the finite operators of wTCCS. Thus for* $P \geq Q$ *the following hold;*

1. $[t]P \geq [t]Q$,
2. $a.P \geq a.Q$,
3. $P + E \geq Q + E$,
4. $P \mid E \geq Q \mid E$,
5. $P\backslash L \geq Q\backslash L$,
6. $P[S] \geq Q[S]$.

Proof:(following [Mil86], Proposition 5.17)

Most of the cases are self-evident so we shall provide a proof of one as an example. We shall show that given $P \geq Q$ then $P \mid E \geq Q \mid E$. We will show that the relation $T = \{(P \mid E\ ,\ Q \mid E) \mid P \geq Q\}$ is a strong pre-order. Consider firstly the action evolutions of $P \mid E$ so if $P \mid E \xrightarrow{\mu} R$ then there are three cases;

1. $R \equiv P' \mid E$ then $P \xrightarrow{\mu} P'$ and from the definition $Q \xrightarrow{\mu} Q'$ with $P' \geq Q'$ so $Q \mid E \xrightarrow{\mu} Q' \mid E$ with $(P' \mid E, Q' \mid E) \in T$.
2. $R \equiv P \mid E'$ then $Q \mid E \xrightarrow{\mu} Q \mid E'$ with $(P \mid E', Q \mid E') \in T$.
3. $R \equiv P' \mid E'$ and $\mu = \tau$; then there is a μ such that $P \xrightarrow{\mu} P'$ and $E \xrightarrow{\overline{\mu}} E'$ so $Q \xrightarrow{\mu} Q'$ with $P' \geq Q'$ therefore we can infer $Q \mid E \xrightarrow{\tau} Q' \mid E'$ and $(P' \mid E', Q' \mid E') \in T$.

The time evolutions are much simpler since we know that if $P \mid E \overset{t}{\rightsquigarrow} R$ then $R \equiv P' \mid E'$ where $P \overset{t}{\rightsquigarrow} P'$ and $E \overset{t}{\rightsquigarrow} E'$ thus there exits Q' and $t' \geq t$ such that $Q \mid E \overset{t'}{\rightsquigarrow} Q' \mid E''$ with $P' \mid E' \geq Q' \mid E''$, since from $E \overset{t}{\rightsquigarrow} E'$ we can infer $E \overset{t'}{\rightsquigarrow} E''$ with $E' \equiv E''$ from the time evolution rule.

Unfortunately this pre-order is *not* substitutive in sTCCS, for consider the following processes;

$$P = (5)a \quad Q = (7)a \quad E = (6)b,$$
$$R = P \mid E \text{ and } S = Q \mid E.$$

Clearly $P \geq Q$ but there does not exists a $t \geq 5$ such that $S \overset{t'}{\rightsquigarrow} S'$ with $a \mid (1)b \geq S'$, since we can *never* reach an S' which can perform an a action without performing a b action first. A similar lack of congruence can be observed with respect to the non-determinism operator.

Strong pre-order is not substitutive with respect to sTCCS, owing to the introduction of causality via timing information. In the system sTCCS the following two processes are equivalent,

$$(5)a \mid (7)b \text{ and } (5)a.(2)b.$$

The process on the left has the causality introduced implicitly by timing, while the other process has the causality explicitly introduced by action-prefix. Any interleaving 'faster than' relation will not preserve timing causality, but will preserve structural causality. Thus to obtain a notion of 'faster than' that will be an order for sTCCS, we shall need to distinguish between the *explicit* introduction of causality through structure and the *implicit* causality introduced by timing.

We will not proceed with further study of the strong pre-order. Since we regard the system sTCCS as the more fundamental we will instead attempt to find an equivalence relation which is substitutive for that system.

4 An Equality for sTCCS.

Definition 4.1 *Processes P and Q are* time-equivalent *iff there exists a relationship R between P and Q such that for all $\mu \in Act$ and for all times t;*

1. *if $P \xrightarrow{\mu} P'$ then there exists Q' such that $Q \xrightarrow{\mu} Q'$ and $(P', Q') \in R$,*
2. *if $P \overset{t}{\rightsquigarrow} P'$ then there exists Q' such that $Q \overset{t}{\rightsquigarrow} Q'$ and $(P', Q') \in R$,*
3. *if $Q \xrightarrow{\mu} Q'$ then there exists P' such that $P \xrightarrow{\mu} P'$ and $(P', Q') \in R$,*
4. *if $Q \overset{t}{\rightsquigarrow} Q'$ then there exists P' such that $P \overset{t}{\rightsquigarrow} P'$ and $(P', Q') \in R$,*

the relation R is called a time-equivalence.

Proposition 4.2 *If S, S' and S_i for all $i \in I$, are time-equivalences; then the following are also time-equivalences;*

1. *Id_P,*
2. *SS',*

3. S^{-1},

4. $\bigcup_{i \in I} S_i$.

Proof: A trivial extension to the proof of proposition 8.3.2

Definition 4.3 $\sim_T \equiv \bigcup\{S \mid S \textit{ is a time-equivalence}\}$

Proposition 4.4

1. $\sim_T$ *is a time-equivalence,*

2. $\sim_T$ *is the largest time-equivalence.*

Proof:

1. directly from the above proposition.
2. immediately from the definition of $\sim_T$.

Definition 4.5 T *is a time-equivalence up to* $\sim_T$ *whenever* $\sim_T T \sim_T$ *is a time-equivalence.*

Proposition 4.6 *If* T *is a* time-equivalence to $\sim_T$ *then* $T \subseteq \sim_T$.

Proof: for $(P,Q) \in T$ we have $P \sim_T PTQ \sim_T Q$ and thus $P \sim_T T \sim_T Q$.

The usual functional extension in the style of [Par81] can be used to demonstrate time equivalence well founded; as the details are precisely those of the earlier definition and proof we omit them.

Proposition 4.7 *Time-equivalence is substitutive for the finite operators of sTCCS. In other words, given* $P \sim_T Q$ *then;*

1. $a.P \sim_T a.Q$,

2. $(t)P \sim_T (t)P$,

3. $\delta P \sim_T \delta Q$,

4. $P + E \sim_T Q + E$,

5. $P \mid E \sim_T Q \mid E$,

6. $P \backslash L \sim_T Q \backslash L$,

7. $P[S] \sim_T Q[S]$.

In order to prove the above we need to demonstrate that appropriate bisimulations can be found. The detail has been presented earlier and is therefore omitted.

Definition 4.8 *Let E and F be two expressions, with free variables $\tilde{X}$. Then we will say that $E \sim_T F$ iff for all vectors of processes $\tilde{P}$*

$$E[\tilde{P}/\tilde{X}] \sim_T F[\tilde{P}/\tilde{X}]$$

Proposition 4.9 *given $E \sim_T F$ then $Fix_{\tilde{X}} E \sim_T Fix_{\tilde{X}} F$.*

Proof: consider only a single pair of equations:

$$A \equiv E[A/X] \text{ and } B \equiv F[B/X]$$

with $E \sim_T F$. We will show that the relation
$\{(G[A/X], G[B/X]) \mid G$ contains at most X free $\}$ is a time-equivalence to $\sim_T$. We proceed by induction over the depth of inference by which either $G[A/X] \xrightarrow{\mu} P'$ or $G[A/X] \overset{t}{\rightsquigarrow} P''$ is inferred. Then we continue by case analysis over the structure of G. The resulting proof mirrors that of [Mil80,Mil86]; but we have to perform each case both for action and temporal evolutions. For example consider the case for the parallel composition operator. Suppose $G \equiv G_1 \mid G_2$ then

- if $G_1[A/X] \xrightarrow{\mu} G_1'[A/X]$ then $G[A/X] \xrightarrow{\mu} G_1'[A/X] \mid G_2[A/X]$ but from the inductive assumption $G_1[A/X] \sim_T G_1[B/X]$ as there actions are derived by a shorter inference; as we can then immediately obtain the desired result.
- Similarly for $G_2[A/X] \xrightarrow{\mu} G_2'[A/X]$.
- if $G_1[A/X] \mid G_2[A/X] \xrightarrow{\tau} G_1'[A/X] \mid G_2'[A/X]$ then there is a λ such that

  $$G_1[A/X] \xrightarrow{\lambda} P_1' \text{ and } G_2[A/X] \xrightarrow{\overline{\lambda}} P_2'.$$

  So from the inductive assumption we can find:

  $$G_1[B/X] \xrightarrow{\lambda} Q_1' \text{ and } G_2[B/X] \xrightarrow{\overline{\lambda}} Q_2';$$
  $$\text{with } P_1' \sim_T T \sim_T Q_1' \text{ and } P_2' \sim_T T \sim_T Q_2'.$$

  Take $P' = P_1' \mid P_2'$ and $Q' = Q_1' \mid Q_2'$ then $P_i = H_i[A/X]$ for some H and similarly $Q_i = H_i[B/X]$. But $(P_i, Q_i) \in \sim_T T \sim_T$ so letting $H = H_1 \mid H_2$ we have $(P', Q') \equiv (H[A/X], H[B/X]) \in T$ and hence

  $$G[A/X] \xrightarrow{\tau} P' \text{ and } G[B/X] \xrightarrow{\tau} Q'$$
  $$\text{with } (P', Q') \in T.$$

- $G_1[A/X] \mid G_2[B/X] \overset{t}{\rightsquigarrow} P' \mid Q'$ this is the same as the above case but with time actions replacing the normal actions.

The other structural cases follow the same pattern.

4.1 Stratified Bisimulations.

In his original discription of the language CCS [Mil80], Milner used a stratified notion of bisimulation, in that the bisimulation was given in terms of the number of actions for which two processes were initally equivalent. In the limit this gives the full bisimulation.

In the timed system we have two approaches available to giving a stratified account of bisimulation; one is that for an arbitary number of initial actions of both time and computation our processes match, the other is that for an initial period of time our processes are identical. We would hope that the limiting cases of both these forms of bisimulation give us our original notion of bisimulation. Unfortunately when using time we do not have ordinals and thus can only work with finitely branching processes and do not recover the full bisimulation. In the following we shall be working with the ordinal numbers, which we will denote $\mathcal{O}$.

Definition 4.10 *(Action Stratified Bisimilation)*

1. $P \sim_0 Q$ *for all processes* P *and* Q,
2. $P \sim_{n+1} Q$ *iff*
   - *For all* t *such that* $P \stackrel{t}{\rightsquigarrow} P'$ *then there exists* Q' *such that* $Q \stackrel{t}{\rightsquigarrow} Q'$ *and* $P \sim_n Q$,
   - *For all* a *such that* $P \stackrel{a}{\longrightarrow} P'$ *then there exists* Q' *such that* $Q \stackrel{a}{\longrightarrow} Q'$ *and* $P' \sim_n Q'$,
   - *For all* t *such that* $Q \stackrel{t}{\rightsquigarrow} Q'$ *then there exists* P' *such that* $P \stackrel{t}{\rightsquigarrow} P'$ *and* $P \sim_n Q$,
   - *For all* a *such that* $Q \stackrel{a}{\longrightarrow} Q'$ *then there exists* P' *such that* $P \stackrel{a}{\longrightarrow} P'$ *and* $P' \sim_n Q'$,
3. *for each limit ordinal* λ $P \sim_\lambda Q$ *iff for all* $\kappa < \lambda$ $P \sim_\kappa Q$.

Proposition 4.11 $\sim = \bigcap_{\kappa \in \mathcal{O}} \sim_k$

Definition 4.12 *(Time Stratified Bisimulation(1))* $P \stackrel{t}{\sim} Q$ *iff*

- *for all* t' *such that* $P \stackrel{t'}{\rightsquigarrow} P'$ *then there exists* Q' *such that* $Q \stackrel{t'}{\rightsquigarrow} Q'$ *and* $P' \stackrel{t-t'}{\sim} Q'$,
- *for all* a *such that* $P \stackrel{a}{\longrightarrow} P'$ *then there exists* Q' *such that* $Q \stackrel{a}{\longrightarrow} Q'$ *and* $P' \stackrel{t}{\sim} Q'$,
- *for all* t' *such that* $Q \stackrel{t'}{\rightsquigarrow} Q'$ *then there exists* P' *such that* $P \stackrel{t'}{\rightsquigarrow} P'$ *and* $P' \stackrel{t-t'}{\sim} Q'$,
- *for all* a *such that* $Q \stackrel{a}{\longrightarrow} Q'$ *then there exists* P' *such that* $P \stackrel{a}{\longrightarrow} P'$ *and* $P' \stackrel{t}{\sim} Q'$,

Definition 4.13 *(Time Stratified Bisimulation (2))* $P \stackrel{t}{\sim_i} Q$ *iff* $(t)nil \mid P \sim (t)nil \mid Q$.

Proposition 4.14 $P \stackrel{t}{\sim} Q$ *iff* $P \stackrel{t}{\sim_i} Q$.

Proposition 4.15 *For finitely branching* P *and* Q. $P \sim Q$ *iff for all* t $P \stackrel{t}{\sim} Q$.

Corollary 4.16 *Let* P *and* Q *be finitely branching processes and* $t_1, \ldots, t_i, \ldots$ *an infinite strictly increasing sequence of times. Then if for all* t_i $P \stackrel{t_i}{\sim} Q$ *then* $P \sim Q$.

5 Temporal Structure independence

We wish to equate processes where performing the same action costs the same amount of time. Whilst the strong equivalence appears to do this, it is only relevant if we assume that all of the time cost of an event is located either immediately before it or immediately after it, not if it has a cost distributed either side of it. Consider the following two processes;

$$P \equiv (1)a.(5)R,$$
$$Q \equiv (3)a.(3)R,$$

where R is an arbitary process. If we take the time actions above as the cost of performing the a action in both cases, then these processes should be considered equivalent in that sense, but they are certainly not equivalent in the strong sense.

Essentially we wish to identify processes where we can find time slices between which they are alway capable of the same computation, but not necessarily at precisely the same time. We would like to base this notion of equivalence on an order. We start by abstracting actions to remove the precise location with respect to time and replace it with a notion of associated cost in time.

Proposition 5.1 *Let P be a process in wTCCS, or sTCCS over a dense time (ie, the reals or the rationals). If $P \overset{t}{\rightsquigarrow} P'$ then there exists P'' such that $P \overset{t_1}{\rightsquigarrow} P''$ and $P \overset{t_2}{\rightsquigarrow} P'$, with $t_1 + t_2 = t$.*

> **Proof:** Immediate from the nature of the TIME rule in both the case of sTCCS and wTCCS.

Definition 5.2 *For any process P, $P \xrightarrow[t]{\mu} Q$ iff one of the following holds*

1. *there exists P' and P'' such that :*
   - *(a)* $P \overset{t_1}{\rightsquigarrow} P'$,
   - *(b)* $P' \overset{\mu}{\rightarrow} P''$,
   - *(c)* $P'' \overset{t_2}{\rightsquigarrow} Q$,
   - *(d)* $t = t_1 + t_2$.
2. *There exists P' such that:*
   - *(a)* $P \overset{t}{\rightsquigarrow} P'$,
   - *(b)* $P' \overset{\mu}{\rightarrow} Q$.
3. *There exists P' such that:*

(a) $P \xrightarrow{\mu} P'$,

(b) $P' \overset{t}{\rightsquigarrow} Q$.

4. *Failing the above.*

(a) $P \xrightarrow{\mu} Q$,

(b) $t = 0$.

Now we can define a temporal simulation. The evolution defined above is an abstraction from the underlying timed system; a similar method is used in [Smo90].

Definition 5.3 *We will say P is faster than Q iff there exists a relation T, called a* Temporal Simulation, *such that for all $\mu \in Act$ and for all times t;*

1. *if $P \xrightarrow[t]{\mu} P'$ then there exists Q', t' such that $Q \xrightarrow[t']{\mu} Q'$ and $(P', Q') \in T$ with $t' \geq t$,*

2. *if $Q \xrightarrow[t]{\mu} Q'$ then there exists P', t' such that $P \xrightarrow[t']{\mu} P'$ and $(P', Q') \in T$ with $t' \leq t$.*

Unfortunately, owing to the different nature of TCCS and both sTCCS and wTCCS this is not the same as the form of temporal simulation as defined in [Tof88]. However it does have the property of equating the processes A and B defined earlier.

Proposition 5.4 *If T, T' and T_i for all $i \in I$, are temporal simulations then the following are all temporal simulations;*

1. Id_P,

2. TT',

3. $\bigcup_{i \in I} T_i$.

Definition 5.5 $\geq_T = \bigcup\{T \mid T \text{ is a temporal simulation}\}$.

Proposition 5.6

- $\geq_T$ *is a temporal simulation,*
- $\geq_T$ *is the largest temporal simulation.*

Once again the above can be demonstrated well founded by a functional definition in the style of [Par81].

Proposition 5.7 *If $P \geq_T Q$ then for finite wTCCS processes P and Q then;*

- $[t]P \geq_T [t']Q$ *with* $t' \geq t$,
- $P + E \geq_T Q + E$,
- $P \mid E \geq_T Q \mid E$,
- $P[S] \geq_T Q[S]$,
- $P\backslash L \geq_T Q\backslash L$.

Unfortunately the problems with implicit causality prevent this order being substitutive for sTCCS. We do not include a proof of the above proposition since it is in essence identical to that given earlier.

6 Equational Characterisation.

Consider the following equations.

Given $P \sim_T Q$ and $t_1 \leq t_2$ then the following equations are true of sTCCS processes:

1. Action Prefix;
   (a) $(t_1)(t_2)P = (t_1 + t_2)P$,
2. Non-determinism;
   (a) $P + P = P$,
   (b) $P + \delta Nil = P$,
   (c) $(t)P + Nil = Nil$,
   (d) $P + R = R + P$,
   (e) $(t_1)P + (t_2)Q = (t_1)(P + (t_2 - t_1)Q)$,
   (f) $P + (R + S) = (P + R) + S$,
   (g) $(t_1)\delta P + (t_2)\delta P = (t_1)\delta P$,
   (h) $\delta a.P + a.P = a.P$,
   (i) $\delta P + \delta R = \delta(P + R)$,
   (j) $(t_1)a.P + (t_2)R = (t_1)a.P$,
   (k) $a.P(t)S = a.P$,

3. Composition;

   (a) $P \mid \delta Nil = P$,

   (b) $(t)P \mid Nil = Nil$,

   (c) $P \mid R = R \mid P$,

   (d) $P \mid (R \mid S) = (P \mid R) \mid S$,

   (e) $(t_1)P \mid (t_2)Q = (t_1)(P \mid (t_2 - t_1)Q)$,

4. Restriction;

   (a) $a.P\backslash L = Nil$ if a, $overlinea \in L$,

   (b) $a.(P\backslash L) = (a.P)\backslash L$ if a, $\overline{a} \notin L$,

   (c) $(t)(P\backslash L) = ((t)P)\backslash L$,

   (d) $\delta(P\backslash L) = (\delta P)\backslash L$,

   (e) $P = P\backslash L$ if for all $a \in \mathcal{L}(P)$ $a \notin L$,

   (f) $P\backslash L_1\backslash L_2 = P\backslash L_1 \cup L_2$,

   (g) $P\backslash L + Q\backslash L = (P + Q)\backslash L$,

   (h) $(P\backslash L) \mid Q = (P \mid Q)\backslash L$, if for all $a \in \mathcal{L}(Q)$ a, $\overline{a} \notin L$,

We have not given rules for wTCCS since we cannot obtain the same structural identities. But there is no manipulation possible of the temporal operators owing to the property that processes of the form

$$[2][4]P$$

deadlock. In other words, in

$$R \equiv [2]a.P \mid [3]Q$$

if we tried to replace this process by the obvious,

$$S \equiv [2](a.P \mid [1]Q);$$

then this would temporally deadlock in the context $S\backslash a$ whereas $R\backslash a$ does not. Since further passage of time will permit the process Q to evolve. In the strong system however both the processes;

$$R' = (2)a.P \mid (3)Q \text{ and } S' = (2)(a.P \mid (1)Q),$$

deadlock in the context $\backslash a$. Thus it is possible to manipulate the time action prefixes in the strong system. A similar property ensures that we cannot distribute weak time over a non-deterministic pair of processes.

Most of the above equations come directly from CCS and from the natural properties of time. The non-determinism equations come from the unwillingness of a sTCCS process to delay when it can perform a normal action. As in, for instance, equation 2.k where we are stating that either it immediately evolve either through the action or by undelaying the alternative.

There is no general equation analogous to the expansion theorem, so this equational system is not complete. This results from the problems of introducing unintentional causality. In [Mol89] an extended form of the calculus sTCCS is shown to admit an equational theory which is both sound and complete with respect to strong bisimulation.

7 Observational Evolutions.

We can define the usual observational notions of evolution.

Definition 7.1 *For* $s = \lambda_1 \ldots \lambda_n \in Act^*$ *we say that,* $P \xrightarrow[t]{s} P'$ *iff* $P \xrightarrow[t_1]{\lambda_1} \ldots \xrightarrow[t_n]{\lambda_n} P'$ *with* $t = t_1 + \ldots + t_n$.

Definition 7.2 $P \stackrel{s}{\Longrightarrow}_t P'$ *iff* $P \xrightarrow[t]{s_0} P'$ *for some* $\hat{s_0} = s$

Time pressure means that we have not looked at the equivalence induced on processes by this evolution, but we suspect that we will obtain results on wTCCS which mirror those for CCS. However since we cannot yet provide a substitutive order for sTCCS, we do not suspect that an order induced by observational evolutions will be substitutive.

8 Process Logic For Timed CCS.

We introduce a simple extension of the process logic $\mathcal{PL}$ [Mil86], with a timed modal operator. Since our space of times may be dense, it is not sufficient to add a next operator which is interpreted as at the next instant the proposition holds. A similar logic is presented in [Koy87]. The formulae of our logic ($\sqcup\mathcal{PL}$) are defined as follows:

$$F ::= \bigwedge_{i \in I} F_i \mid \neg F \mid < a > F \mid \{t\}F.$$

Definition 8.1 *The satisfaction relation between processes and formulae is defined as follows;* $P \models_T F$ *iff:*

- $P \models_T \bigwedge_{i \in I} F_i$ *iff for all* $i \in I$, $P \models_T F_i$,

- $P \models_T \neg F$ *iff* $P \models_T F$ *is* false,
- $P \models_T < a > F$ *iff there exits* P' *such that* $P \xrightarrow{a} P'$ *and* $P' \models_T F$,
- $P \models_T \{t\}F$ *iff either*
  - *there exists* a, P' *such that* $P \xrightarrow{a} P'$ *and* $P \models_T F$,
  - *or there exists* P', t' *such that* $P \overset{t'}{\rightsquigarrow} P'$ *with* $t' \leq t$ *and* $P' \models_T F$.

Proposition 8.2 *If* $P \models_T \{t\}F$ *then for all* $t' \geq t$, $P \models_T \{t'\}F$.

Proof: Immediate from the definition of the operator $\{t\}$.

Note: since zero times are not permitted in sTCCS the first clause of the temporal modality only applies to delays and these are matched over the equivalence.

8.1 Stratifying $t\mathcal{PL}$

We wish to show that $t\mathcal{PL}$ characterses strong temporal bisimilation. We start by defining the depth of formulae in our logic.

Definition 8.3 *We define the depth of a formula recursively over its structure;*

- $depth(\{t\}F) = 1 + depth(F)$,
- $depth(< a > F) = 1 + depth(F)$,
- $depth(\neg F) = depth(F)$,
- $depth(\bigwedge_{i \in I} F_i) = max_{i \in I}(depth(F_i))$.

Definition 8.4 *The stratified family of formulae* $t\mathcal{PL}_k \overset{def}{\equiv} \{F \mid depth(f) \leq k\}$

Proposition 8.5 *For each* $k \in \mathcal{O}$, $P \sim_k Q$ *iff for every* $F \in t\mathcal{PL}_k$

$$P \models F \text{ iff } Q \models F.$$

Proof: We start by proving that bisimularity up to k implies that the processes satisfy exactly the same formulae of $t\mathcal{PL}_k$ we proceed by induction on k. Assume $P \sim_k Q$ and $P \models F$ where $depthF \leq k$ we require to prove that $Q \models F$. There are two critical cases;

1. $F \equiv < a > F'$,
2. $F \equiv \{t\}F'$.

In the first case, we have that $P \xrightarrow{a} P'$ for some P', and $P' \models F'$ with $depth(F') = \lambda < k$. Since $P \sim_k Q$ there exists Q' with $Q \xrightarrow{a} Q'$ and $P' \sim_{k-1} Q'$, thus by inductive assumption $P' \models F'$ iff $Q' \models F'$ since $depth(F') \leq k-1$ and hence $Q \models F$.

In the second case, we have that $P \overset{t'}{\rightsquigarrow} P'$ for some P' and $t' \leq t$ with $P' \models F'$ with $depth(F') = \lambda < k$ and an identical argument to above holds. Or $P \xrightarrow{a} P''$ and $P \models F'$, but if $P \sim_k Q$ then $P \sim k-1Q$, and there must exist Q'' such that $Q \xrightarrow{a} Q''$ and thus $Q \models F'$ as required.

We now show that if two processes satisfy exactly the same formulae in $t\mathcal{PL}_k$ then they are bisimilar to k. Assume $P \not\sim_k Q$, we will look for $F \in t\mathcal{PL}_k$ such that $P \models F$ and $Q \not\models F$. Consider $k = \lambda + 1$, then wlog assume, either

1. $P \xrightarrow{a} P'$ and for all Q' such that $Q \xrightarrow{a} Q'$ then $P' \not\sim_\lambda Q'$, or
2. $P \overset{t}{\rightsquigarrow} P'$ and for all Q' such that $Q \overset{t}{\rightsquigarrow} Q'$ then $P' \not\sim_\lambda Q'$.

From the inductive assumption if $P' \not\sim_\lambda Q'_i$ then there exists a set of formulae $F_i \in t\mathcal{PL}_\lambda$ such that $P' \models F_i$ and $Q'_i \not\models F'_i$. Where we have a set of Q derivatives $\{Q_i \mid i \in I\}$. Then to distinguish our process we can use the formulae;

1. use $F = < a > \bigwedge_{i \in I} F_i$, and
2. $F = t \bigwedge_{i \in I} F_i$ respectively,

by construction the depth of the formulae above is lees than or equal to k. The proof for limit ordinals follows immediately. □

Corollary 8.6 *$P \sim Q$ iff for all $F \in t\mathcal{PL}$*

$$P \models F \text{ iff } Q \models F.$$

To handle observational congruences we define an extended operator, for $s \in \Lambda^*$ and $t \in Times$, $[[s,t]]^*_T$, in the following way;

Definition 8.7 *$P \models [[s,t]]^*_T F$ iff there exists $\{t_0\}[s_1]\{t_1\}...[s_n]\{t_n\}$ such that $P \models \{t_0\}[s_1]\{t_1\}...[s_n]\{t_n\}F$ and $t_0 + t_1 + ... + t_n \leq t$.*

We can define the observational version of the modal operators, by using the hat operator, i.e $P \models_T [[\hat{s},t]]_T F$ iff $P \models_T [[s,t]]^*_T F$.

Note: we can add quantifiers and implications in the same manner as for $\mathcal{PL}$.

9 Examples.

9.1 A Simple Timer.

Consider the example of the timed and the timer controlled actions presented in [Tof88].

$$E_a \equiv a.[t]E_a$$
$$Timer \equiv \overline{alarm}.[t']Timer$$
$$E'_a \equiv alarm.[t_1]a.[0].E'_a$$
$$R_a \equiv (Timer \mid E'_a)\backslash alarm$$

We consider the evolutions of the two systems with $t_1 < t'$ and $t_2 = t' - t_1$:

$$E_a \xrightarrow{a}\!\!\rightsquigarrow^{t} E_a,$$
$$\text{thus } E_a \overset{a}{\underset{t}{\Longrightarrow}} E_a,$$
$$R_a \xrightarrow{\tau}\!\!\rightsquigarrow^{t_1} \xrightarrow{a}\!\!\rightsquigarrow^{t_2} R_a,$$
$$\text{thus } R_a \overset{a}{\underset{t'}{\Longrightarrow}} R_a.$$

Note, these are minimum time paths.

It seems that the time of the process R_a is independent of the value of t_1 provided it remains less than t'. It seems that we can achieve an analogous result to the weakness replacement [Tof88]. The following example shows that in these systems weakness replacement will hold only when processes behaviours are much more constrained than the requirements of the original result.

Consider the following processes;

$$P = \overline{b}.[5]P_1,$$
$$Q = \overline{b}.[10]Q_1,$$
$$E = b.[20]Nil,$$
$$R_1 = E \parallel P,$$
$$R_2 = E \parallel Q.$$

Even if P_1 and Q_1 are identical then R_1 will not be the same as R_2. The former can evolve to a state equivalent to P_1, twice as fast as the latter. Thus the condition for the processes to be made identical by communicating with a slower process must include repeated communication.

9.2 Action Available For a Period.

In the following process:

$$(5)\delta a.P + (7)a.P$$

the action a can be inferred at any time between 5 and 7, but it must have been used by at latest 7 or the process will deadlock. This can be used to represent a process that requires a certain time to start and is then only available for a limited period.

9.3 Actions With Duration.

So far we have assumed that our actions take no time, when modelling circuits however we would like our actions to extend over a period of time. In [Cas] the method of splitting an action into a pair of new actions is suggested, one signifying the start of the signal, the other the conclusion. Following that method to get an a action of duration 5 we take the new actions $a_{sta\ rt}$ and a_{finish} and construct the following:

$$a_{start}.(5)a_{finish}.P$$

which represents a process pre-fixed by an a action of duration 5. To synchronize with the above we need the action pair, $\overline{a}_{start}$ and $\overline{a}_{finish}$ (the natural duration version of $\overline{a}$). So the following process could synchronize on the a action:

$$\overline{a}_{start}.(5)\overline{a}_{finish}.Q.$$

We can construct processes that will respond to signals of arbitary duration; for example the following requires an a of duration at least 2 to synchronise with

$$\overline{a}_{start}.(2)\delta\overline{a}_{finish}.Q.$$

9.4 Representing SCCS in sTCCS.

We will assume that our action set *Act* is an abelian group with τ as its identity, and $\overline{a}$ as the inverse of a. We can then represent an SCCS process as an sTCCS process using the following translation.

Definition 9.1 *Let P be an SCCS process and $P^\dagger$ its translation into sTCCS, we define $P^\dagger$ as follows:*

SCCS	*sTCCS*
$\mathbf{0}$	Nil
$a.P$	$(1)a.P^\dagger$
$P+Q$	$P^\dagger + Q^\dagger$
$P \times Q$	$P^\dagger \mid Q^\dagger$

Definition 9.2 *Let $s \in Act^*$ be a sequence of actions. Then prod(s) is the action $s_1 \times s_2 \times \ldots \times s_n$, where $s = s_1 s_2 \ldots s_n$*

Note, for any permutation s' of a sequence of actions s; $prod(s) = prod(s')$.

Proposition 9.3 *Whenever can infer* $P \xrightarrow{a} Q$ *in SCCS iff we can infer* $P^{\dagger} \overset{1}{\rightsquigarrow} \xrightarrow{s} Q$ *in sTCCS, with* $prod(s) = a$.

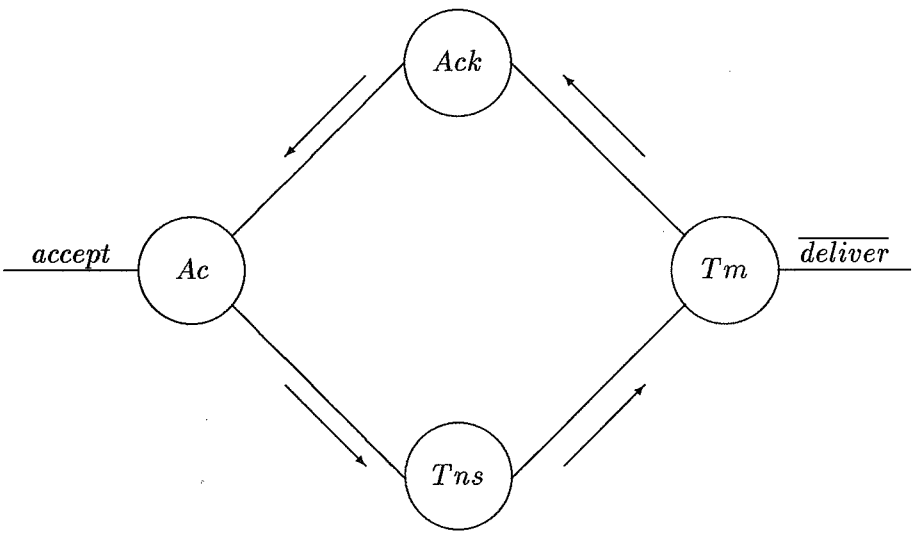
Proof: By a induction over the the structure of the processes.

10 Value Passing.

If we use the operators of CCS with values and either set of temporal operators then we can derive a value passing calculus from our basic calculus in an identical manner to that used for CCS. The translation was presented in chapter 1.

11 The Alternating Bit Protocol.

In his book [Mil86] Milner presents an implementation of the alternating bit protocol in CCS, and demonstrates that the protocol is correct. Perforce this implementation ignores the exact temporal properties of the system and its components. We extend the implementation in CCS to one in sTCCS, where we can take advantage of the temporal properties of the system components. In this version some of the system complexity can be reduced by exploiting the temporal information. An alternating bit protocol realisation can be viewed as follows. (We let $\tilde{b}$ stand for the negation of the boolean value b.)



The process Ac will work in the following manner. After accepting a message, it sends it with bit b along the channel Tns and waits, subsequently there are three possibilities:

- it times out, and re-Transmits the message;
- it gets an acknowledgement b from the Ack line (correct transmission), so it can now accept another message;
- it gets an acknowledgement $\tilde{b}$ (superfluous extra acknowledgement of earlier message) which is ignored.

The replier Tm works in a dual manner. After a message is delivered it sends an acknowledgement with bit b along the Ack line. There are then three possibilities:

- it times out, and re-transmits the acknowledgement;
- it gets a new message with bit $\tilde{b}$ from the Tns line, which it delivers and acknowledges with bit $\tilde{b}$;
- it gets a repetition of the old message with bit b which is ignored.

The channels in the implementation are identical and have the ability to duplicate or lose an arbitrary message, arbitrarily many times. For convenience we ignore actual messages and concentrate on the value of the control bits.

We now give the definition of a timed version of the alternating bit protocol in the following fashion. There are 4 fundamental times involved, the transmission times on both channels and the re-try times in both senders.

$$Ac(b) = \delta ack(b).Ac(b) + \delta ack(\tilde{b}).Ac(b) + \delta accept.Sd(\tilde{b})$$
$$Sd(b) = \overline{send}(b).Sd_1(b) + \delta ack(b).Sd(b) + \delta ack(\tilde{b}).Ac(\tilde{b})$$
$$Sd_1(b) = (t_{rt})\overline{send}(b).Sd_1(b) + \delta ack(b).(t_a)Sd_1(b) + \delta ack(\tilde{b}).Ac(\tilde{b})$$

$$Tm(b) = \delta transmit(b).Tm(b) + \delta transmit(\tilde{b}).Tm(b) + \overline{deliver}.Rp(b)$$
$$Rp(b) = \overline{reply}(b).Rp_1(b) + \delta transmit(\tilde{b}).Rp(b) + \delta transmit(b).Tm(\tilde{b})$$
$$Rp_1(b) = (t_{rt}).\overline{reply}(b).Rp_1(b) + \delta transmit(\tilde{b}).Rp_1(b) + \delta transmit(b).Tm(\tilde{b})$$

Let $s = s_1 b_2 s_2$

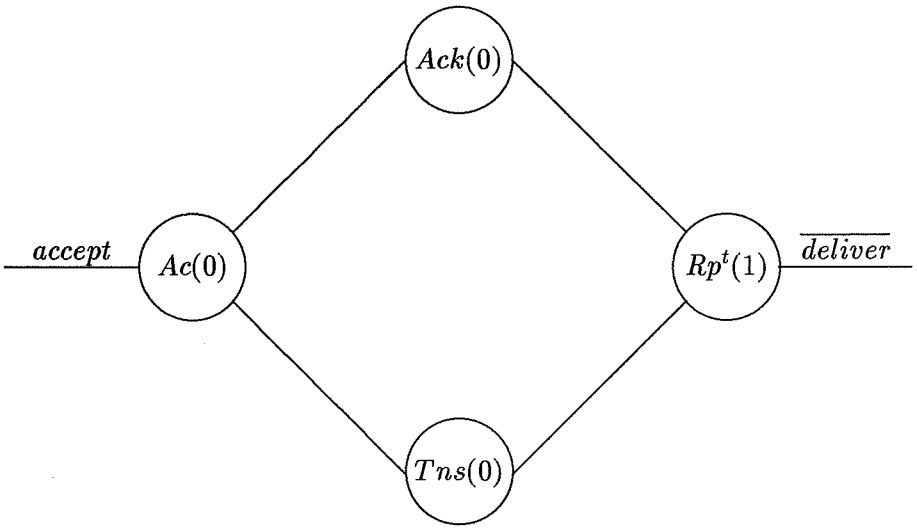
$$Tns(b_1 s_1 b_2 s_2) = \delta send(b).Tns(bb_1 s) + (t_s)\overline{Transmit}(b_1).Tns(s) + \delta\tau.Tns(b_1 s_1 s_2) + \delta\tau.Tns(s)$$

$$Ack(b_1 s_1 b_2 s_2) = \delta reply(b).Ack(bb_1 s) + (t_s)\overline{ack}(b).Ack(s) + \delta\tau.Ack(b_1 s_1 s_2) + \delta\tau.Ack(s)$$

With initial state,

$$Ab \equiv Ac(0) \parallel Ack(0) \parallel Tns(0) \parallel Rp^t(1)$$

A static flow diagram for the above is



We firstly impose the restriction that t_a and t_t are less than t_s; this is sufficient to ensure that the *ack* and *transmit* ports are always available to the transmission channels. We impose the condition on the re-try rate that $t_{rt} > 2t_s$. This seems reasonable as we cannot be sure that a transmission has failed until this period of time has elapsed. The external environment is forced to take delivery as soon as possible. This enables us to calculate the period of time between the reception of an *accept* action, and the return of the process to a position where the next transmission may be attempted.

From the constraint on re-tries we can observe that there is no path with a time short enough to require that the transmission channel should contain more than one data item at once. We redefine our channels as follows.

$$Ack = \delta reply(b).Ack(b)$$
$$Ack(b) = (t_s)\overline{ack}.Ack + \delta\tau.Ack$$

$$Tns = \delta send(b).Tns(b)$$
$$Tns(b) = (t_s)\overline{transmit}.Tns + \delta\tau.Tns$$

With initial state,
$$Ab = Ac(0) \parallel Ack \parallel Tns \parallel Rp_1^{t_s}(1).$$

It should be noted that we have arranged that at no time is an immediate *ack* or *transmit* action of either type impossible, apart for a period immediately after the reception of such an action, and then the evolution is possible before any more actions of that type can be produced

by the channels. Thus we can use actions with no delay guarding in the channels to force the evolution to proceed at certain intervals.

Let us examine the evolution of the process Ab. (Note that we will not divide time actions below the minimum to permit another physical evolution. We will include a path only if two immediate evolutions lead eventually to the same state. For clarity we will subscript the τ actions with the initial letter of the original action from which they where inferred. We will mark points at which genuine divergence can take place with a number and the evolution will be reconsidered from there later.)

$$
\begin{array}{ll}
& Ac(0) \parallel Ack \parallel Tns \parallel Rp^{t_s}(1) \\
\stackrel{accept}{\longrightarrow} & Sd(1) \parallel Ack \parallel Tns \parallel Rp_1^{t_s}(1) \\
\stackrel{\tau_s}{\longrightarrow} & Sd_1(1) \parallel Ack \parallel Tns(1) \parallel Rp_1^{t_s}(1) \\
\stackrel{t_s}{\rightsquigarrow} & Sd_1^{t_s}(1) \parallel Ack \parallel transmit1.Tns \parallel Rp_1^{2t_s}(1) \qquad (1) \\
\stackrel{\tau_t}{\longrightarrow} & Sd_1^{t_s}(1) \parallel Ack \parallel Tns \parallel Tm(0) \\
\stackrel{deliver}{\longrightarrow} & Sd_1^{t_s}(1) \parallel Ack \parallel Tns \parallel Rp(0) \\
\stackrel{\tau_a}{\longrightarrow} & Sd_1^{t_s}(1) \parallel Ack(0) \parallel Tns \parallel Rp_1(0) \\
\stackrel{t_s}{\rightsquigarrow} & Sd_1^{2t_s}(1) \parallel ack0.Ack \parallel Tns \parallel Rp_1^{t_s}(0) \qquad (2) \\
\stackrel{\tau_a}{\longrightarrow} & Ac(1) \parallel Ack \parallel Tns \parallel Rp_1^{t_s}(0)
\end{array}
$$

This is the inversion of the initial state so their has been a correct transmission in time $2t_s$, if there are no errors, as we would expect. We now consider the evolution if there is an error on the first transmission (which we labelled (1) above) some period of time t, after the transmission was started. Note that t must be less than or equal to our transmission time t_s.

$$
\begin{array}{ll}
& Sd_1(1) \parallel Ack \parallel Tns(1) \parallel Rp_1^{t_s}(1) \\
\stackrel{t}{\rightsquigarrow} & Sd_1^{t}(1) \parallel Ack \parallel Tns \parallel Rp_1^{t_s+t}(1) \\
\stackrel{t_{rt}-t_s-t}{\rightsquigarrow} & Sd_1^{t_{rt}-t_s}(1) \parallel Ack \parallel Tns \parallel Rp(1) \qquad (*) \\
\stackrel{\tau_r}{\longrightarrow} & Sd^{t_{rt}-t_s}(1) \parallel Ack(1) \parallel Tns \parallel Rp_1(1) \\
\stackrel{t_s}{\rightsquigarrow} & Sd_1^{t_{rt}} \parallel ack1.Ack \parallel Tns \parallel Rp_1^{t_s}(1)
\end{array}
$$

There are now two possible evolution paths, one leading to the re-try as desired the other requiring a further complete cycle before the re-try can even be attempted. If the following alternative is chosen,

$$
\begin{array}{ll}
& \stackrel{\tau_a}{\longrightarrow} Sd_1(1) \parallel Ack \parallel Tns \parallel Rp_1^{t_s}(1) \\
& \stackrel{t_{rt}-t_s}{\rightsquigarrow} Sd_1^{t_{rt}-t_s} \parallel Ack \parallel Tns \parallel Rp(1)
\end{array}
$$

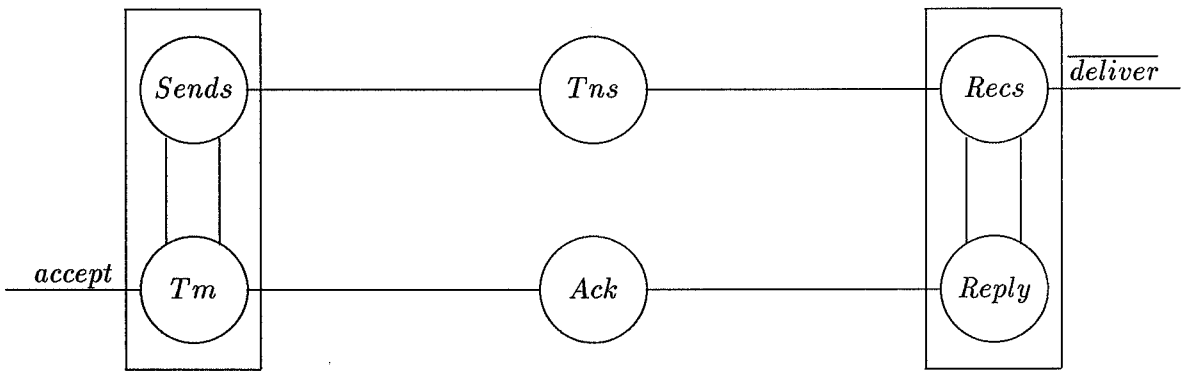
This is the same as state (*) and thus has the same behaviour. Thus we have an implementation of the protocol which has a non-linear response to error: after one error we have a potential computation which will *never* correctly transmit. This can be corrected by separating the activities of re-trying, and responding to enquiries. The next version of the alternating-bit protocol is produced along these lines, and has a linear response to errors.

$$Tmc(b) = \delta ack(b).Tmc(b) + \delta ack(\tilde{b}).Tmc(b) + accept.\overline{start_s}(b).Tms(\tilde{b})$$
$$Tms(b) = \delta ack(b).Tms(b) + \delta ack(\tilde{b}).\overline{stop_s}.\ Tmc(\tilde{b})$$
$$Sends = \delta start_s(b).\overline{send}(b).Send(b) + \delta stop_s.Sends$$
$$Send(b) = (t_{rt})\overline{send}(b).Send(b) + \delta stop_s.Sends$$
$$Recc(b) = \delta transmit(b).Recc(b) + \delta transmit(\tilde{b})Recc(b) + \overline{deliver}.\overline{start_r}(b).Recs(b)$$
$$Recs(b) = \delta transmit(\tilde{b}).Recs(b) + \delta transmit(b).\ \overline{stop_r}.Recc(\tilde{b})$$
$$Reply = \delta start_r(b).\overline{reply}(b).Replys(b) + \delta stop_r.Reply$$
$$Replys = (t_{rt})\overline{reply}(b) + \delta stop_r.Reply$$

With the restriction that $t_{rt} > 2t_s$ we can re-use our earlier channels and the complete protocol's initial state is;

$$Tmc(0) \parallel Sends \parallel Tns \parallel Ack \parallel Recs(1) \parallel Reply^{t_s}(1).$$

A static flow diagram for the above is.



It is relatively easy to check that the time performance of this process is linearly affected by the number of errors in transmission.

12 Conclusions.

We have presented a temporal model which is an extension of CCS. This system has most of the properties we desire, excepting that we have so far not demonstrated an order over processes. We believe that such an order may be obtained using the techniques of [Cas87]. The temporal behaviour of the systems sTCCS and wTCCS has little effect on the operational behaviour derived from CCS, and we believe that the methodology of separating action and temporal evolutions can successfully extend any underlying operational reasoning system for computation actions.

13 Bibliography.

[Ber85] J. A. Bergstra, J. W. Klop, Algebra of Communicating Processes with Abstraction, Theoretical Computer Science, Volume 37, No 1, 1985.

[Car85] L. Cardelli, Real Time Agents, Proc. 9th ICALP LNCS 140.

[Cas87] I. Castellani and M. Hennessey, Distributed Bisimulation, Report Sussex University 5/87, July 1987.

[Dir58] P.A.M. Dirac, Principles of Quantum Mechanics, 4 th. Edition Oxford 1958.

[Hoa85] A. Hoare, Communicating Sequential Processes, Prentice Hall 1985.

[Jef] A. Jeffrey, Synchronous CSP, Oxford University, to appear.

[Koy83] R. Koymans, J. Vytopil, W.P. de Roever, Real-Time and Asynchronous Message Passing, Technical Report, RUU-CS-83-9, University of Eindhoven,1983.

[Koy87] R. Koymans, Specifying Message Passing and Real-Time Systems with Real-Time Temporal Logic, Technical report, University of Eindhoven, 1987.

[Mil80] R. Milner, A Calculus of Communicating Systems, Springer LNCS vol 92.

[Mil83] R. Milner, Calculi for Synchrony and Asynchrony, Theoretical Computer Science 25(3), pp 267-310, 1983.

[Mil89] R. Milner, Communication and Concurrency, Prentice Hall, 1989.

[Mol89] F. Moller and C.Tofts, A Temporal Calculus for Communicating Systems, LFCS-89-104, university of Edinburgh.

[Nie84] H. Nielson, Hoare Logic's for Run Time Analysis of Programs, Thesis Edinburgh University 1984.

[Par81] D. Park, Concurrency and Automata on infinite sequences, Springer LNCS 104.

[Plo81] G. D. Plotkin, A structured approach to operational semantics. Technical report Daimi Fn-19, Computer Science Department, Aarhus University. 1981

[Rei85] W. Reisig, Pertri Nets, an Introduction, Springer-Verlag 1985.

[Ros86] G. M. Reed and A. W. Roscoe, A Timed Model for CSP, LNCS 226: ICALP 1986.

[Shi82] L. I. Schiff, Quantum Mechanics, 3 rd. edition, Mcgraw Hill 1982.

[Smo90] S. Smolska, B. Steffen and C. Tofts, A calculus of Relative Frequency, to appear.

[Tof88] C. Tofts, Temporal Ordering for Concurrency, LFCS-88-49, University of Edinburgh.