# A Real-Time Process Algebra with Open Intervals and Maximal Progress

Harald Fecher

Universität Mannheim
Fakultät für Mathematik und Informatik,
D7, 27, 68131 Mannheim, Germany
hfecher@pi2.informatik.uni-mannheim.de
published at http://www.ii.uib.no/~nwpt00

## Extended Abstract

*Real-time* models are used to model time-dependent physical problems. The common approach is to restrict action occurrences to closed, more precisely to left closed intervals. Contrary to this, *timed automata* [AD94] and some process algebras (for example [Che92, Dan91]) allow actions to occur in open intervals in order to increase their expressive power.

Another important property of timed process algebras is the *maximal progress assumption*, i.e. an action must occur at the instant at which all participants are ready. This assumption thus guarantees that time cannot proceed if an internal action is enabled. Wang's timed CCS [Wan91] and Schneider's timed CSP [Sch00] are examples of real-time process algebras which satisfy the maximal progress assumption.

The combination of the maximal progress assumption and action occurrence in open intervals is useful to model real-time problems as illustrated by the following example.

*Example 1.* Let us consider a plane which regularly sends an *ok*-signal to the tower (perform an $\overline{ok}$-action), confirming that everything is all right. If the *ok*-signal which the safety officer received last is more than 5 minutes ago, he is allowed to send off a rescue team. If an *ok*-signal arrives before the rescue team is sent off, the safety officer is no longer allowed to send them off until another 5 minutes have passed without an *ok*-signal having been received. The safety officer can be modeled by the CCS like term

$$E_{s-officer} = \left(\mu X.\left(ok_{\geq 0}^{<\infty}.X + send_{>5}^{<\infty}.F\right) | E_p\right) \setminus \{ok\},$$

where the plane is represented by $E_p$, the activity of the rescue team by the process $F$ and the sending of the rescue team by the *send*-action.

The maximal progress assumption is needed here, because otherwise the *send*-action could stay enabled even if an $\overline{ok}$-action was enabled by $E_p$.

Considering a process algebra with those two properties, the internal action in the interval $(0,1)$ (denoted by $\tau_{>0}^{<1}$) is forced to perform at time point 0, since otherwise it would contradict the maximal progress assumption. On the other hand, we want to distinguish, for example, between $\tau^{<1}_{>0}$ and $\tau^{<1}_{\geq 0}$, since in the summation $\tau^{<1}_{>0}.E_1 + \tau^{<1}_{\geq 0}.E_2$ the latter will always disable the former. To solve this problem, we assume that the internal action in the process $\tau^{<1}_{>0}$ will perform a non-observable time step after 0. Therefore, the time domain of the transition system is chosen to be the positive real numbers extended by a non-observable time step (denoted by $\iota$).

Let us consider the process $a^{<1}_{>0}$. On the one hand, it must be able to perform an $a$-action after an $\iota$-step, since it could participate in an internal action (for example in $a^{<1}_{>0}|\bar{a}^{<1}_{>0}$). But on the other hand, it must not perform an $a$-action after an $\iota$-step, since it would then perform the action at the time point 0, which contradicts the specification by the interval $(0, 1)$. To overcome this problem, we extend the action set by a marked copy of it (the elements of the marked set are denoted for example by $\hat{a}$), where the marked internal action is the internal action itself. So for example $a^{<1}_{>0} \xrightarrow{\iota} \hat{a}^{<1}_{\geq 0}$. The marked actions are only used to produce internal actions, i.e. branches containing marked actions will not be considered in the observation.

We only mark the action after an $\iota$-step if the action's interval is left open at zero, i.e. is of the kind $(0, d)$ or $(0, d]$, because otherwise it would not be possible for the process $\left(\tau^{\leq 1}_{>0}.b^{\leq 0}_{\geq 0}\right) \Big| a^{\leq 1}_{\geq 0}$ to perform the $a$-action after the $b$-action without passing real time, which is in our sense counterintuitive.

When a real-time step takes place, we remove the mark from the action, since if it has already been marked before this step it is now in 'the middle of its interval' and therefore behaves normally.

If an action results in an internal action, for example by communication between the parallel operator in CCS or by the hiding operator in CSP, no difference is made between marked and unmarked actions. In CCS style, for example, the action $\bar{a}$ (and also $\hat{\bar{a}}$) can communicate with $\hat{a}$ and, of course, with $a$ as well ($\hat{a}^{\leq 1}_{\geq 0}|\bar{a}^{\leq 3}_{\geq 0} \xrightarrow{\tau}$ ).

The difference between marked and unmarked actions is only made in the observation. Therefore, a bisimulation equivalence is used that abstracts away from $\iota$-steps and neglects branches which contain marked actions. Unfortunately, this equivalence fails to be a congruence. For example, $\tau^{\leq 1}_{>0} + a^{\leq 1}_{>0}.b^{\leq 0}_{\geq 0}$ and $\tau^{\leq 1}_{>0}$ are equivalent with respect to the considered equivalence. But if we use them in parallel with $\bar{a}^{\leq 1}_{>0}$, they will behave differently, since the former may perform a $b$-action after an internal action.

Introducing the process algebra *Open Timed CCS* we illustrate how a process algebra that meets both properties, the maximal progress assumption and action occurrences in open intervals, can be constructed. Open Timed CCS is based on the real-time process algebra of [Che92], and therefore based on CCS. The transition rules of Open Timed CCS which deal with time passing in action prefix terms are presented in Table 1. The *time additivity* [NS92] (for real-time steps) and the *time determinism* [NS92] properties are satisfied by Open Timed CCS.

$$\frac{0 < e'}{\alpha^{\prec' e'}_{> \ 0}.E \xrightarrow{\iota} \hat{\alpha}^{\prec' e'}_{\geq \ 0}.E} \qquad \frac{0 < e' \quad (\alpha = \tau \Rightarrow 0 < e) \quad (e = 0 \Rightarrow \succ = \geq)}{\alpha^{\prec' e'}_{\succ \ e}.E \xrightarrow{\iota} \alpha^{\prec' e'}_{\succ \ e}.E}$$

$$\frac{d \prec' e' \quad d \leq e}{\alpha^{\prec' e'}_{\succ \ e}.E \xrightarrow{d} \check{\alpha}^{\prec' e'-d}_{\succ \ e-d}.E} \qquad \frac{d \prec' e' \quad \alpha \neq \tau \quad d > e}{\alpha^{\prec' e'}_{\succ \ e}.E \xrightarrow{d} \check{\alpha}^{\prec' e'-d}_{\geq \ 0}.E}$$

where $\check{\alpha}$ denotes the unmarking of $\alpha$

**Table 1.** Operational Semantics (incomplete)

Open Timed CCS allows to model dynamic priority of actions [CLN99] at the same point in time. This is contrary to [BCL99], where it is necessary to pass real-time to model dynamic priority. We model dynamic priority in Open Timed CCS by using internal actions which get enabled in open intervals. Example 2 illustrates this idea. Since in Open Timed CCS internal actions stipulate a choice in the summation operator, we have to copy each process of higher priority behind each introduced internal action.

*Example 2.* Consider the process $a^{\leq 7}_{\geq 2}.E + b^{\leq 6}_{\geq 2}.F$, where $a$ has priority 0 and $b$ has priority $-2$. Then the process

$$a^{\leq 7}_{\geq 2}.E + \tau^{\leq 3}_{> 2}.\left(a^{\leq 5}_{\geq 0}.E + \tau^{\leq 1}_{> 0}.\left(a^{\leq 5}_{\geq 0}.E + b^{\leq 4}_{\geq 0}.F\right)\right)$$

has the desired behavior, if we abstract away of $\iota$-steps and internal actions.

## References


[AD94] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[BCL99] Girish Bhat, Rance Cleaveland, and Gerald Lüttgen. A practical approach to implementing real-time semantics. *Annals of Software Engineering*, 7:127–155, 1999.

[Che92] L. Chen. An interleaving model for real–time systems. *LNCS*, 620:81–92, 1992.

[CLN99] R. Cleaveland, G. Lüttgen, and V. Natarajan. Priority in process algebras. Technical Report TR-99-3, Institute for Computer Applications in Science and Engineering, 1999.

[Dan91] M. Daniels. Modelling real–time behaviour with an interval time calculus. *LNCS*, 571:53–72, 1991.

[NS92] X. Nicollin and J. Sifakis. An overview and synthesis on timed process algebras. *LNCS*, 600:526–548, 1992.

[Sch00] S. Schneider. *Concurrent and Real-time Systems: The CSP Approach.* Wiley, 2000.

[Wan91] Y. Wang. CCS+time = an interleaving model for real time systems. *LNCS*, 510:217–228, 1991.