Symbolic Bisimulation for Timed Processes*

Michele Boreale
Università di Roma "La Sapienza"

Abstract

Basing on symbolic transition systems, we propose a novel approach to the semantics of timed processes. A process algebra in which actions may occur within specified time intervals is introduced, together with a notion of bisimulation equivalence, based on standard transition systems.

The language is also equipped with a new, symbolic operational semantics. The latter, contrary to standard operational semantics, gives rise to transition systems which are finitely branching and, for a large class of processes, finite. On top of the symbolic operational semantics, we introduce a notion of symbolic bisimulation, for which a tractable proof technique exists. We then prove that symbolic and standard bisimulations coincide for our processes. A proof system to reason about bisimilarity is also presented. The soundness and completeness proofs for the system take great advantage of the symbolic characterization of bisimilarity.

1 Introduction

The treatment of timing aspects is often a delicate issue in concurrent systems, particularly if they must react in real-time to stimuli from the external environment. When describing a system that controls a dangerous industrial process, we should be able to specify that, after the detection of some error, a certain recovery action must take place within a given time interval. Several attempts have been made to incorporating timing aspects in classical process algebras, such as CCS and CSP: among the many, [8, 3, 2]. Here, we focus on a specific approach, taken e.g. in [3, 2]: it is based on attaching *time-stamps* to standard actions, to record their time of occurrence. For example, in the notation of [3], the expression

$$\int_{t\in[0,5]} a(t).P \qquad (1)$$

represents a non-deterministic process that may engage action a at any real time between 0 and 5, and that then behaves like P.

A central goal of a theory of timed systems should be that of providing feasible reasoning techniques, perhaps amenable to mechanization. In this respect, a first,

*Work done while the author was at Istituto per Elaborazione dell'Informazione - CNR, Pisa. The work has been partially supported by EEC, HCM Project Express and by CNR within the project "Specifica ad Alto Livello e Verifica di Sistemi Digitali". Author's e-mail address: michele@dsi.uniroma1.it.

obvious difficulty is encountered if trying (as in [3]) to model time by means of real numbers, which cannot have a finite representation. A reasonable solution to this is restricting to rational numbers, which, for all applicative purposes, are the best possible approximation of reals. Even so, formulating a tractable theory remains problematic. The standard way of modelling the operational behaviour of (1) is to use a transition system having a transition labelled by $a(r)$ for each value r allowed in the interval $[0, 5]$ (see e.g. [3]). In other words, the temporal parameter t is instantiated in all possible manners. This leads typically to transition systems which are infinite-branching (*infinitary*), even in very simple cases like (1). Usual behavioural relations, like bisimilarity [7], can be defined in a familiar way over such structures, but reasoning with them is generally quite difficult. In particular, the fact that the considered transition systems are infinitary prevents us from applying usual reasoning techniques of process algebras: the existence of complete proof systems, for example, is problematic in this setting, or relies on very ad-hoc techniques [5]. For similar reasons, there seems to be no obvious way of extending known automatic verification methods to this setting.

In the present paper, we try to address some of the above issues, focusing on a specific behavioural equivalence, bisimilarity, written $\sim$. We introduce a language of rational-timed processes, equipped with a notion of bisimilarity based on a standard infinitary transition system. The language we consider is quite expressive (more than, e.g., Wang's timed CCS [8]). For example, our language naturally permits describing time-outs or specifying that one action must take place within a certain time interval after another (relative delays). In this timed setting, we then develop a theory of *symbolic bisimulation*, in the style of Hennessy's and Lin's theory for value-passing processes [6]. Our theory yields an alternative, finitary representation of processes behaviours and a more tractable characterization of bisimilarity. For the latter, in particular, "exhibiting symbolic bisimulation relations" turns out to be a feasible reasoning technique. We also propose a sound and complete proof system for reasoning about $\sim$. A more detailed account of our work is as follows.

Our approach can be explained in two steps. First, besides the standard infinitary one, a new *symbolic* operational semantics is introduced. There, the infinitely many standard transitions raised by expressions like (1) are reduced to a single, symbolic transition. In the latter, no instantiation of the temporal parameter t is performed; rather, the temporal constraints on t are recorded via a logical formula. Thanks to this fact, the resulting symbolic transition system is finite-branching, and, for a large class of timed processes (including all those without recursion), is even finite. To clarify these points, let us write expression (1) in the more conventional form $a(t, 0 \leq t \leq 5).P$. According to our approach, the latter gives rise to the single symbolic transition:

$$a(t, 0 \leq t \leq 5).P \overset{a(t),(0\leq t\leq 5)}{\longmapsto} P\,.$$

In general, our logical formulae are built, via the usual boolean connectives, from temporal variables, rational values, operator $+$ and predicates $<$ and $=$. As a second step, on top of the symbolic transition system, a new, symbolic equivalence is defined. Intuitively, two timed processes P and Q are symbolically equivalent if for every symbolic move $P \overset{a(t)\,\phi}{\longmapsto} P'$ of P, we can find a decomposition in sub-cases of the formula ϕ, such that each subcase "entails", in a logical sense, a matching transition for Q (and vice-versa for Q, P). As a simple example, consider the processes:

$$\begin{array}{rcl} P & \overset{def}{=} & a(t, 0 \leq t \leq 10).P' \qquad \text{and} \\ Q & \overset{def}{=} & a(t, 0 \leq t \leq 5).P' + a(t, 5 < t < 10).P' \end{array}$$

where $+$ is the operator of non-deterministic choice. Proving the equivalence of P and Q by relying on the definition of $\sim$ would imply exhibiting an infinite binary relation of processes, due to the necessity of instantiating the parameter t. On the contrary, the symbolic equivalence of P and Q is readily established by noting that the formula $(0 \leq t \leq 10)$, attached to the only symbolic transition of P, can be decomposed into the set of subcases $\{(0 \leq t \leq 5), (5 < t < 10)\}$, each of which implies a formula in a symbolic transition of Q (the vice-versa for Q, P is obvious). It is worth noting that the decompositions can always be taken *finite*.

Our main theorem shows that the symbolic equivalence, $\simeq$, can be used to establish the standard one, $\sim$. More precisely, we prove that standard and symbolic bisimilarities coincide on processes not containing free temporal parameters. This makes "exhibiting symbolic bisimulation relations" a feasible proof technique for $\sim$ and lays the basis for the development of automatic verification tools. For the latter, in particular, it should be possible to adapt to our setting Hennessy's and Lin's algorithms for checking value-passing symbolic equivalence [6]: but this point remains to be fully worked out.

For processes without recursion, we also prove decidability of timed bisimilarity and put forward a sound and complete proof system. The corresponding proofs take great advantage of the symbolic characterization of $\sim$ and are themselves examples of application of the proposed techniques.

A few words about the "local-time vs. global-time" issue are in order. Like in [2], we adopt for our language the local-time point of view. However, we will argue that this choice is not critical for the development of our theory.

The rest of the paper is organized as follows. Section 2 describes the language of timed processes and standard bisimulation over it. Symbolic transitional semantics and symbolic bisimulation are introduced and discussed in Section 3. In Section 4, after having established some technical properties, the main theorems, stating the correspondence between symbolic and standard bisimulation, are discussed. The proof system is described in Section 5. Finally, Section 6 contains comparison with related work and a few conclusive remarks.

2 The Language

2.1 Syntax

To manipulate temporal constraints in processes, we use a simple language of boolean formulae, BF. We introduce BF below, and then describe the syntax and the standard semantics of timed processes.

2.1.1 Expressions and boolean formulae

We assume a countable set Var of *variables* ranged over by $t, t', \ldots$. The letters $r, r' \ldots$ range over the non-negative rationals, $\mathbf{Q}^{\geq 0}$. Letters $p, p', \ldots$ range over $Var \cup \mathbf{Q}^{\geq 0}$.

Expressions e are given by $e := p \;\mid\; p + r$. The set of rational values and the set of variables occurring in e are denoted, respectively, by $val(e)$ and $var(e)$. The language of *boolean formulae* BF, ranged over by $\phi, \psi, \ldots$, is given by the following grammar:

$$\phi := true \;\mid\; p < e \;\mid\; p = e \;\mid\; \phi \wedge \phi \;\mid\; \neg\phi .$$

In the sequel, standard abbreviations such as $false$ for $\neg true$, $\phi \vee \psi$ for $\neg(\neg\phi \wedge \neg\psi)$, $p \leq e$ for $(p < e) \vee (p = e)$ and so on, will be freely used. We denote by $var(\phi)$ (resp. $val(\phi)$) the set variables (resp. rational values) occurring in ϕ.

Environments, ranged over by $\sigma, \rho, \ldots$, are finite partial maps from Var to $\mathbf{Q}^{\geq 0}$; $[r_1/t_1, \ldots, r_n/t_n]$, $n \geq 0$, denotes the environment mapping t_i to r_i, for $1 \leq i \leq n$, and undefined elsewhere. For a given σ, $\sigma[{}^r\!/t]$ denotes the environment which maps t to r and behaves like σ elsewhere. The domain of σ is denoted by $dom(\sigma)$. For any expression or formula h, $h\sigma$ denotes the result of substituting each $x \in var(h) \cap dom(\sigma)$ by $\sigma(x)$.

An expression or a formula is *ground* if it does not contain variables. Given a ground expression e, its *evaluation* $[\![e]\!]$ is the non-negative rational value obtained by evaluating e in the expected way (once the operator symbol $+$ is interpreted as the standard summation over $\mathbf{Q}^{\geq 0}$). Given a ground formula ϕ, the evaluation of ϕ into the set $\{true, false\}$, $[\![\phi]\!]$, is inductively defined in the expected way, once we set that: $[\![true]\!] = true$, $[\![r < e]\!] = true$ iff $r < [\![e]\!]$ and $[\![r = e]\!] = true$ iff $r = [\![e]\!]$. Some basic (somewhat standard) notions on environments and formulae are summarized below.

- $\sigma \models \phi$ (σ *satisfies* ϕ), holds if $dom(\sigma) \supseteq var(\phi)$ and $[\![\phi\sigma]\!] = true$. A formula ϕ is *satisfiable* if there exists σ s.t. $\sigma \models \phi$.

- $\phi \models \psi$ (ϕ *logically implies* ψ) holds if for every σ s.t. $dom(\sigma) \supseteq var(\phi) \cup var(\psi)$, $\sigma \models \phi$ implies $\sigma \models \psi$. We say that ϕ and ψ are *equivalent* if $\phi \models \psi$ and $\psi \models \phi$.

- $\bigvee D$, where $D = \{\phi_1, \ldots, \phi_n\} \subseteq_{fin} BF$, $n > 0$, is the boolean formula $\phi_1 \vee \ldots \vee \phi_n$. A similar notation will be used for $\bigwedge D$. Furthermore, we let $\bigvee \emptyset$ denote $false$ and $\bigwedge \emptyset$ denote $true$.

2.1.2 Timed processes

As in CCS, we assume a countable set A of *actions* and a bijection $\bar{\_} : A \longrightarrow \overline{A}$, giving the complementary action for each action a, with the property that $\bar{\bar{a}} = a$; *Act* is $A \cup \overline{A}$ and is ranged over by $a, b, \ldots$. Each action in A is assigned positive *duration*, given by a function $\Delta : A \longrightarrow \mathbf{Q}^{+}$; the latter is extended to *Act* by letting $\Delta(\bar{a}) = \Delta(a)$. A family of labels $T = \{\tau_\delta | \delta \in \mathbf{Q}^{+} \}$ disjoint from *Act*, to be used for internal actions of processes, is also assumed. Letters $c, c', \ldots$ range over $Act \cup T$. A countable set of *agent identifiers*, ranged over by $A, B, \ldots$ and each having a non-negative arity, is presupposed. The language of *agent terms*, $\mathcal{P}$, ranged over by $P, Q, \ldots$, is built from the operators of *inaction, action prefix, summation, boolean guard, parallel composition, restriction* and *agent identifier*:

$$P := \mathbf{0} \;\mid\; \alpha.P \;\mid\; P + P \;\mid\; \phi P \;\mid\; P \,|\, P \;\mid\; P \backslash a \;\mid\; A(e_1, \ldots e_k)$$

where k is the arity of the identifier A and $\alpha := wait_\delta \;\mid\; a(t, \phi)$, with $\delta \in \mathbf{Q}^{+}$ and $\phi \models t \leq e$, for some expression e not containing t. The prefix $a(t, \phi)$. is intented to be a *binder* for t in $a(t, \phi).P$, thus the notions of *free variables* $fvar(\,.\,)$, *bound variables* $bvar(\,.\,)$ and *α-equivalence* over agent terms are the expected ones. We shall consider agent terms up to α-equivalence: thus terms differing only in the choice of the bound names will be identified. We will sometimes use $a(e).P$ as a shorthand for $a(t, t = e).P$, whenever $t \notin \mathit{fvar}(P, e)$. We sometimes abbreviate $a(t, \phi).\mathbf{0}$ simply as $a(t, \phi)$. We assume an arbitrarily fixed *finite* set $\mathcal{D}$ of guarded identifiers definitions, each having the form $A(t_1, \ldots, t_n) \Leftarrow P$, with the t_i's distinct and $\mathit{fvar}(P) \subseteq \{t_1, \ldots, t_n\}$. A *process* is a closed agent term, i.e. a P such that $\mathit{fvar}(P) = \emptyset$. The set of processes is denoted by $\mathcal{P}_c$.

Some intuitive explanation on the operators of our language is in order. Inaction, summation, parallel composition and restriction have essentially the same meaning as in CCS. The meaning of the other constructs is as follows. In $a(t, \phi).P$, action a can be engaged starting at a certain time t and then last for a time $\Delta(a)$; the time t must satisfy a certain temporal constraint, expressed by the formula ϕ. E.g., the action a in $a(t, 5 \le t \le 8).P$ can start at any time between 5 and 8. Note that we require that ϕ determine some upper-bound to the value of t: for example, $a(t, \mathit{true})$. is a forbidden action prefix. In other words, actions cannot delay arbitrarily long. This assumption can be fulfilled in many practical situations and greatly simplifies the theory of symbolic bisimulation, as we shall see in later sections. The action wait_δ represents an internal activity that takes time δ. Process ϕP behaves like P if ϕ is true and like $\mathbf{0}$ otherwise. Finally, agent identifiers allow us to specify recursive behaviours, using a set of temporal parameters. The following example shows that the language $\mathcal{P}_c$ is indeed quite expressive.

Example 2.1 Let us describe a simple communication protocol supporting time-out. Starting at time t_0, a communication $\overline{\mathit{com}}$ occurs, after which an acknowledgment ack is awaited for a time T; if this time elapses without receiving any ack, a recovery procedure $\overline{\mathit{recov}}$ is invokated (we assume for simplicity that the duration of actions $\overline{\mathit{com}}$, ack and $\overline{\mathit{recov}}$ is 1):

$$A(t_0) \Leftarrow \overline{\mathit{com}}(t_0).\big(\mathit{ack}(t, t < t_0 + T).A(t+1) + \overline{\mathit{recov}}(t', t' = t_0 + T).A(t'+1)\big).$$

2.1.3 Configurations

In order to describe standard operational semantics, we follow [2] and use *configurations* to keep track of the time of occurrence of actions. The idea is that of associating with each process P a "clock", $\bullet\, e$, recording the current time, thus obtaining a configuration $P \bullet e$. A process P is then viewed as a configuration starting at time 0, $P \bullet 0$. The syntax of configurations $\mathcal{C}$, ranged over by $S, R, \ldots$, is as follows:

$$S := \mathbf{0} \quad | \quad P \bullet e \quad | \quad \phi S \quad | \quad S + S \quad | \quad S \,|\, S \quad | \quad S \backslash a \ .$$

Following [2], we will consider configurations in canonical form with respect to the clock $\bullet\, e$, which is assumed to be distributive over all operators different from α. and identifiers. Formally, let $\equiv$ be defined as the least congruence generated by the axioms below:

$$\begin{array}{rclrcl} (P + Q) \bullet e & = & (P \bullet e) + (Q \bullet e) & (\phi\, P) \bullet e & = & \phi(P \bullet e) \\ (P \,|\, Q) \bullet e & = & (P \bullet e) \,|\, (Q \bullet e) & (P \backslash a) \bullet e & = & (P \bullet e) \backslash a \end{array}$$
$$\mathbf{0} \bullet e = \mathbf{0}\,.$$

Applying these axioms as left-to-right rewriting rules, it is easy to see that each configuration S is reducible to a $\equiv$-congruent canonical configuration generated by the grammar:

$$\texttt{Act}\,\frac{-}{(a(t,\phi).P)\bullet e \xrightarrow{a(r)} (P[^r/_t])\bullet (r+\Delta(a))}\,,\ r\geq [\![e]\!] \text{ and } [\![\phi[^r/_t]]\!] = true$$

$$\texttt{Wait}\,\frac{-}{(wait_\delta.P)\bullet e \xrightarrow{\tau_\delta([\![e]\!])} P\bullet([\![e]\!]+\delta)}$$

$$\texttt{Sum}\,\frac{S_1 \xrightarrow{\mu} S_1'}{S_1+S_2 \xrightarrow{\mu} S_1'} \qquad \texttt{Guard}\,\frac{S \xrightarrow{\mu} S'}{\phi S \xrightarrow{\mu} S'}\,,\ [\![\phi]\!]=true \qquad \texttt{Res}\,\frac{S \xrightarrow{\mu} S'}{S\backslash a \xrightarrow{\mu} S'\backslash a}\,,\ a,\bar{a} \text{ not in } \mu$$

$$\texttt{Par}\,\frac{S_1 \xrightarrow{\mu} S_1'}{S_1\,|\,S_2 \xrightarrow{\mu} S_1'\,|\,S_2} \qquad \texttt{Com}\,\frac{S_1 \xrightarrow{a(r)} S_1',\ S_2 \xrightarrow{\bar{a}(r)} S_2'}{S_1\,|\,S_2 \xrightarrow{\tau_{\Delta(a)}(r)} S_1'\,|\,S_2'}$$

$$\texttt{Ide}\,\frac{(P\{e_1/t_1,\ldots,e_n/t_n\})\bullet e \xrightarrow{\mu} S'}{A(e_1,\ldots,e_n)\bullet e \xrightarrow{\mu} S'} \text{ if } A(t_1,\ldots,t_n) \Leftarrow P \text{ is in } \mathcal{D}$$

Table 1: Concrete operational semantics over closed configurations $\mathcal{C}_c$.

$$S := \mathbf{0} \ \mid\ (\alpha.P)\bullet e \ \mid\ A(e_1,\ldots,e_k)\bullet e \ \mid\ \phi S \ \mid\ S+S \ \mid\ S\backslash a \ \mid\ S\,|\,S\,.$$

In the rest of the paper, we will always consider configurations in canonical form w.r.t. $\equiv$. The notions of free and bound variables extend to configurations in the expected way. When giving the standard, "concrete" operational semantics, we will be interested in closed configurations, i.e. the set of those S s.t. $fvar(S)=\emptyset$, which is referred to by $\mathcal{C}_c$.

2.2 Standard operational semantics

We are now set to introduce the standard "concrete" operational semantics over closed configurations, $\mathcal{C}_c$. In the sequel, for H an agent term or a configuration, we let $H\sigma$ denote the result of substituting each $x\in fvar(H)\cap dom(\sigma)$ by $\sigma(x)$. Furthermore, the notation $P\{e_1/t_1,\ldots,e_n/t_n\}$ is used to indicate the simultaneous substitution of the variables t_i's with the expressions e_i's (this may involve renaming of bound names in P); in order to keep the result of this operation within our syntax, we assume that formulae are suitably normalized. (E.g., $\big(a(t,t<t_1+5).\mathbf{0}\big)\{t'+4/t_1\} = a(t,t<t'+9).\mathbf{0}$). Operational semantics of $\mathcal{C}_c$ is presented in Table 1. There, the symmetrical rules of `Sum` and `Par` have been omitted, e is confined to ground expressions and μ ranges over transition labels.

Some comments on the operational semantics are in order. Our rules are related to the operational semantics of [2]. In the latter, all actions are assumed to be "eager", in the sense that they occur as soon as they are allowed. This assumption permits modelling processes behaviour by means of finitary transition systems. Our rule `Act` is the obvious adaption of the corresponding rule in [2] to a more general setting, where actions may also delay. The meaning of `Guard` should be obvious. The other rules of Table 1 are formally identical to rules in [2]. In particular, rule `Par` says that, in a configuration $P \bullet e \,|\, Q \bullet e'$, e and e' are the local clocks at P and Q. These clocks never interact, except when a sinchronization between P and Q takes place (rule `Com`); this also causes clocks to be synchronized (this view is adopted, for example, in certain communication protocols, in which agents use local clocks to time-stamp messages, but periodically synchronize clocks).

On top of closed configurations operational semantics, we define bisimilarity in the standard way.

Definition 2.2 (Bisimilarity) *A symmetric binary relation $\mathcal{R}$ over $\mathcal{C}_c$ is a* bisimulation *iff whenever $S\,\mathcal{R}\,R$ and $S \xrightarrow{\mu} S'$, there exists R' such that $R \xrightarrow{\mu} R'$ and $S'\,\mathcal{R}\,R'$. We say that S is* bisimilar *to R, and write $S \sim R$ iff there exists a bisimulation $\mathcal{R}$ such that $S\,\mathcal{R}\,R$. Given two processes P and Q, we say that P and Q are* bisimilar*, and write $P \sim Q$, iff $P \bullet 0 \sim Q \bullet 0$.*

It is easy to show that the relation $\sim$, over $\mathcal{P}_c$, is preserved by all operators different from $a(t, \phi)$. The binding nature of latter operator raises a little problem: if we want to deduce that $a(t, \phi).P \sim a(t, \phi).Q$ from the equivalence of P and Q, which are in general open terms (as may contain free occurrences of t), we have to extend the relation $\sim$ over such terms. This can be done by "closing" $\sim$ under all possible environments. The congruence properties of $\sim$ will be further discussed in Section 5.

From now on, we will freeely use such abbreviations as $fvar(P, S, \phi, t)$ to mean $fvar(P) \cup fvar(S) \cup var(\phi) \cup \{t\}$. Furthermore, $v(\cdot)$ stands for $val(\cdot) \cup fvar(\cdot)$.

3 Symbolic Semantics

First symbolic operational semantics and then symbolic bisimulation will be introduced over the set of configurations (either closed or open), $\mathcal{C}$.

The rules for operational symbolic semantics over configurations, $\overset{c(t),\phi}{\longmapsto}$, are reported in Table 2. Rules symmetrical to `S − Sum` and `S − Par` have been omitted to save space. In any symbolic transition $S \overset{c(t),\phi}{\longmapsto} S'$, the formula ϕ collects all conditions, on the values and the free variables of S and on the time t itself, necessary for the action c to take place at time t. Note that each symbolic rule is the counter-part of a concrete one. In `S − Act` and `S − Guard`, the side-conditions of the corresponding concrete rules have been, so to speak, moved on the transitions. Here, the side-conditions on the variable t are of syntactic nature, they just ensure that t is taken "fresh", i.e. different from any other free variable in the configuration.

Note that, since we are working up to α-equivalence, α-equivalent configurations are deemed to have the same transitions. Thus, the symbolic transition system is actually finite-branching up to α-equivalence, while the concrete one has an infinite branching factor.

We are now ready to introduce symbolic bisimulation. In a value-passing setting, this notion was first introduced by Hennessy and Lin [6]. It is convenient to define a *family* of symbolic equivalences $\simeq^{\phi}$, each depending on a formula ϕ (equivalence under ϕ), instead of a single relation. The intuitive idea is as follows: S and R are symbolically equivalent under ϕ, $S \simeq^{\phi} R$, if for every symbolic transition of S, say $S \overset{c(t),\psi}{\longmapsto} S'$, one can find a decomposition of the formula $\phi \wedge \psi$ into a *finite* set of

$$\mathtt{S-Act}\ \frac{-}{(a(t,\phi).P)\bullet e \overset{a(t),\phi\wedge(t\geq e)}{\longmapsto} P\bullet(t+\Delta(a))},\ t\notin var(e)$$

$$\mathtt{S-Wait}\ \frac{-}{(wait_\delta.P)\bullet e \overset{\tau_\delta(t),t=e}{\longmapsto} P\bullet(t+\delta)},\ t\notin fvar(P)\cup var(e)$$

$$\mathtt{S-Sum}\ \frac{S_1 \overset{c(t),\phi}{\longmapsto} S_1'}{S_1+S_2 \overset{c(t),\phi}{\longmapsto} S_1'}\qquad \mathtt{S-Guard}\ \frac{S \overset{c(t),\psi}{\longmapsto} S'}{\phi S \overset{c(t),\psi\wedge\phi}{\longmapsto} S'},\ t\notin var(\phi)\qquad \mathtt{S-Res}\ \frac{S \overset{c(t),\phi}{\longmapsto} S'}{S\backslash a \overset{c(t),\phi}{\longmapsto} S'\backslash a},\ c\neq a,\overline{a}$$

$$\mathtt{S-Par}\ \frac{S_1 \overset{c(t),\phi}{\longmapsto} S_1'}{S_1\mid S_2 \overset{c(t),\phi}{\longmapsto} S_1'\mid S_2},\ t\notin fvar(S_2)\qquad \mathtt{S-Com}\ \frac{S_1 \overset{a(t),\phi}{\longmapsto} S_1',\ S_2 \overset{\overline{a}(t),\psi}{\longmapsto} S_2'}{S_1\mid S_2 \overset{\tau_{\Delta(a)}(t),\phi\wedge\psi}{\longmapsto} S_1'\mid S_2'}$$

$$\mathtt{S-Ide}\ \frac{(P\{e_1/t_1,\ldots,e_n/t_n\})\bullet e \overset{c(t),\phi}{\longmapsto} S'}{A(e_1,\ldots,e_n) \overset{c(t),\phi}{\longmapsto} S'},\ \text{if } A(t_1,\ldots,t_n)\Leftarrow P \text{ is in } \mathcal{D}$$

Table 2: Symbolic operational semantics over configurations $\mathcal{C}$.

sub-conditions, each of which "entails" (in a logical sense) a matching transition for R. We will now give the formal definitions and shortly after that explain the necessity of this decomposition.

Definition 3.1 (ϕ-decomposition) *Given ϕ and a finite set of formulae $D = \{\phi_1,\ldots,\phi_n\}$, we say that D is a ϕ-*decomposition *if ϕ is equivalent to $\bigvee D$.*

Definition 3.2 (Symbolic bisimulation)

- *A family $\mathcal{F} = \{\mathcal{R}^\phi \mid \phi\in BF\}$ of symmetric binary relations over configurations, indexed over the set BF of boolean formulae, is a* family of symbolic bisimulations (FSB) *iff for each ϕ and $(S,R)\in\mathcal{R}^\phi$, whenever $S \overset{c(t),\psi}{\longmapsto} S'$ with $t\notin fvar(S,R,\phi)$ then:*

  there exists a $\phi\wedge\psi$-decomposition D, such that for all $\zeta\in D$, there is a transition $R \overset{c(t),\chi}{\longmapsto} R'$ with $\zeta\models\chi$ and $(S',R')\in\mathcal{R}^\zeta$.

- *$S\simeq^\phi R$ iff there exists a FSB $\{\mathcal{R}^\psi \mid \psi\in BF\}$ such that $(S,R)\in\mathcal{R}^\phi$.*

To check symbolic equivalence, it is not necessary to consider all the cases corresponding to the (infinitely many) instantiations of the parameter t in prefixes $a(t,\phi)$. Rather, it is sufficient to consider the single symbolic transition which $a(t,\phi)$. gives rise to. The infinitary "case-analysis" on the value of the time-stamp, implicitly present in the standard definition of bisimilarity, is here embodied in the decomposition D, which can be always taken finite. Furthermore, by choosing D appropriately, the number of cases to deal with can often be kept small. A simple example will help to clarify these points.

Example 3.3 Consider the processes

$$\begin{array}{lcll} P & \overset{def}{=} & a(t, t \leq 10).b(t + \Delta(a)) & \text{and} \\ Q & \overset{def}{=} & a(t, t \leq 10).P' & \text{where:} \\ P' & \overset{def}{=} & (t < 5)b(t + \Delta(a)) + (t \geq 5)b(t + \Delta(a)) + (t = 3)b(3 + \Delta(a)) \,. \end{array}$$

It should be immediate to establish that P and Q are equivalent, in that both of them can engage action a at any time $t \in [0, 10]$, after which action b is immediately executed. We show that $P \bullet 0 \simeq^{true} Q \bullet 0$, by exhibiting an appropriate family of relations. Consider the family of relations $\{\mathcal{R}^{\phi} \cup (\mathcal{R}^{\phi})^{-1} \,|\phi \in BF\}$, where the relations $\mathcal{R}^{\phi}$ are defined as follows:

$$\begin{array}{rcl} \mathcal{R}^{true} & = & \{(P \bullet 0, Q \bullet 0)\} \\ \mathcal{R}^{(t<5)} & = & \{\big(P' \bullet (t + \Delta(a)),\ b(t + \Delta(a)) \bullet (t + \Delta(a))\big)\} \\ \mathcal{R}^{(t\geq 5)} & = & \mathcal{R}^{(t<5)} \ = \ \mathcal{R}^{(t=3)} \ = \ \mathcal{R}^{(t<5)} \\ \mathcal{R}^{\phi} & = & \{(\mathbf{0}, \mathbf{0})\} \text{ for any other different } \phi. \end{array}$$

This corresponds to decomposing the condition $(t \leq 10)$, present in the symbolic transition of P, into the set $\{(t < 5), (t \geq 5), (t = 3)\}$. Indeed, each of these conditions is sufficient to establish that $P' \bullet (t + \Delta(a))$ is equivalent to $b(t + \Delta(a)) \bullet (t + \Delta(a))$.

Note that if want to establish the equivalence of P and Q using the concrete equivalence $\sim$, then we have to exhibit the infinite relation $\mathcal{S} \cup \mathcal{S}^{-1}$, where $\mathcal{S}$ is: $\{(P \bullet 0, Q \bullet 0)\} \cup \{\big(P[^r/_t]' \bullet (r + \Delta(a)),\ b(r + \Delta(a)) \bullet (r + \Delta(a))\big) \,|r \in [0, 10]\} \cup \{(\mathbf{0}, \mathbf{0})\}$.

In [6], Hennessy and Lin present an algorithm that, taken two generic finite symbolic transition systems T and T', produces a formula ψ, which is the weakest formula under which T and T' are symbolically equivalent. Thus, checking the symbolic equivalence under ϕ of T and T' reduces to the problem of checking whether $\phi \models \psi$. Now, modulo the different language, our definition of symbolic bisimulation is formally the same as the one given by Hennessy and Lin [6]. Thus, it should not be too difficult to adapt their algorithm for checking symbolic equivalence to our setting. We want also point out that, as we shall see, the relation $\models$ is decidable for our language. This could lay the basis for automatic verification of timed systems. For processes not containing recursion (agent identifiers), a direct proof of the decidability of symbolic bisimulation can be given, by relying on the existence of a "canonical" decomposition.

4 Consistency and Adequacy of Symbolic Bisimulation

In order to establish the correspondence between symbolic and standard semantics, we have to develop a few technical properties.

4.1 Fundamental properties

In what follows, the symbol V will range over finite subsets of $Var \cup \mathbf{Q}^{\geq 0}$, unless otherwise stated. Furthermore, we say that ρ is an environment *on* V if $dom(\rho) \supseteq$

$V \cap Var$.

A central role in our theory is played by the concept of *complete formula*. In the standard sense, a formula ϕ is complete if for *every* formula ψ either $\phi \models \psi$ or $\phi \models \neg\psi$. This notion is however too strong for our purposes. Intuitively, we will always deal with some fixed, finite set of variables and non-negative rational values (those occurring in the two configurations being compared). Therefore we are only interested in completeness relatively to those formulae whose variables and values are in this set. For technical reasons, also the "closure" of the given set of values, under the arithmetical operations $+$ and $-$, must be considered. These considerations lead to the following definitions.

Definition 4.1 (Closure under $+$ and $-$) *For any $W \subseteq_{fin} \mathbf{Q}^{\geq 0}$ the* closure *of W is the set $\Sigma(W) \subseteq \mathbf{Q}^{\geq 0}$ defined as*
$\{r \geq 0 \,|\, r = \sum_{r' \in W \cup \{1\}} (q_{r'} * r')$ *for some integers $q_{r'}$, with $r' \in W \cup \{1\}$* $\}$.

Note that, given any $r \in \mathbf{Q}^{\geq 0}$, the set $\Sigma(W) \cap [0, r]$ is finite[1]. We denote by $\Sigma(V)$ the set $\Sigma(V \cap \mathbf{Q}^{\geq 0}) \cup V$.

Definition 4.2 (Completeness for formulae) *A formula ϕ is* complete over *V if for each ψ with $v(\psi) \subseteq \Sigma(V)$, either $\phi \models \psi$ or $\phi \models \neg\psi$.*

The next result ensures that, under certain conditions, a formula can be decomposed into a finite set of formulae, each of which is complete, in the sense of Definition 4.2. In the sequel, given a set $Z \subseteq_{fin} Var$, we say that a formula ϕ is *bounding for Z* if there exists a rational r such that $\phi \models \bigwedge_{t \in Z} t \leq r$. By slight abuse of notation, we sometimes say that ϕ is bounding for V, meaning that ϕ is bounding for $V \cap Var$.

Lemma 4.3 (Existence of finite decomposition) *Suppose that ϕ is bounding for V. Then there exists a finite set D of formulae s.t.*

1. *D is a ϕ-decomposition;*

2. *each $\zeta \in D$ is satisfiable, complete over V and bounding for V.*

The next lemma is important as far as decidability of symbolic bisimulation is concerned.

Lemma 4.4 *The relation $\models \subseteq BF \times BF$ is decidable.*

We now come to some important properties of operational semantics. The following proposition is crucial, since it relates symbolic and standard operational semantics. It can be easily shown by transition induction.

Proposition 4.5 (Operational correspondence) *Let ρ be an environment on $fvar(S)$.*

1. *Given any $t \notin fvar(S)$, $S\rho \xrightarrow{c(r)} S_1$ implies $S \overset{c(t),\phi}{\longmapsto} S'$ for some ϕ and S' s.t. $\rho[^r/_t] \models \phi$ and $S_1 = S'\rho[^r/_t]$.*

[1] It can be shown that this property would *not* hold in general if W contained non-rational real numbers.

2. $S \overset{c(t),\phi}{\longmapsto} S'$ *and* $\rho[{}^r/_t] \models \phi$ *imply* $S\rho \xrightarrow{c(r)} S_1$, *with* $S_1 = S'\rho[{}^r/_t]$.

The following definition introduce the concept of "closing" bisimilarity under a family of environments. This notion is useful for stating the next theorem and the theorem of correspondence between symbolic and standard bisimulation.

Definition 4.6 (Closure of $\sim$ under ϕ) *Given any boolean formula ϕ, the relation $\sim^{\phi}$ is defined as: $S_1 \sim^{\phi} S_2$ iff for each environment ρ on $fvar(S_1, S_2, \phi)$ s.t. $\rho \models \phi$, it holds that $S_1\rho \sim S_2\rho$.*

Note, over closed configurations (hence over processes), $\sim^{true}$ is the same as $\sim$. In general, when checking $S \sim^{\phi} R$, one has to check bisimilarity of S and R under all the (possibly infinitely many) environments satisfying ϕ. The next theorem states that considering just one such environment is sufficient when ϕ is complete over $fvar(S,R)$. This is crucial for proving the correspondence between symbolic and standard bisimulation.

Theorem 4.7 *Consider S, R and ζ s.t. ζ is complete over $v(S,R)$. Suppose that for some ρ on $fvar(S,R,\zeta)$ it holds that $\rho \models \zeta$ and $S\rho \sim R\rho$. Then $S \sim^{\zeta} R$.*

4.2 Main results

In order to prove that standard, "concrete" bisimulation $\sim$ coincides with $\simeq^{true}$ over closed configurations, and hence over processes, it is convenient to show a more general correspondence on open configurations. The proof of the latter can be naturally split into two parts, consistency and adequacy.

Theorem 4.8 (Consistency of symbolic bisimulation) *Let S and R be configurations. Then $S \simeq^{\phi} R$ implies $S \sim^{\phi} R$.*

Proof: (Sketch). The relation
$\mathcal{R} = \{(S\sigma, R\sigma) \,|\, \sigma$ is on $fvar(S,R)$ and there exists ϕ with $\sigma \models \phi$ and $S \simeq^{\phi} R\,\}$
is a bisimulation. The thesis is a consequence of this fact, which can be shown by exploiting Proposition 4.5. □

Theorem 4.9 (Adequacy of symbolic bisimulation) *Let ϕ be bounding for $fvar(S,R)$ and suppose that $S \sim^{\phi} R$. Then $S \simeq^{\phi} R$.*

Proof: We prove that the family of relations given by
$$\mathcal{R}^{\phi} = \{(S,R) \,|\, S \sim^{\phi} R \text{ and } \phi \text{ is bounding for } fvar(S,R)\,\}$$
for each $\phi \in BF$, is a family of symbolic bisimulations. This fact implies the thesis.

Suppose that $S\,\mathcal{R}^{\phi}\,R$ and assume that $S \overset{c(t),\psi}{\longmapsto} S'$, for some fresh t. We have to show that there exists a $\phi \wedge \psi$-decomposition D s.t. for each $\zeta \in D$ there is a transition $R \overset{c(t),\chi}{\longmapsto} R'$ such that $\zeta \models \chi$ and $S'\mathcal{R}^{\zeta}R'$. Let $V = v(S,R)$.

Since every action prefix imposes a bound to its variable, it holds that $\psi \models t \leq e$ for some e with $var(e) \subseteq V$. Since ϕ is bounding for V, it easily follows that $\phi \wedge \psi$ is bounding for $V \cup \{t\}$. Therefore, by Lemma 4.3, there exists a $\phi \wedge \psi$-decomposition D such that each $\zeta \in D$ is satisfiable and is complete and bounding for $V \cup \{t\}$. Fix now a generic $\zeta \in D$ and let ρ be any environment on $var(\phi) \cup V \cup \{t\}$ such that $\rho \models \zeta$. Since $\rho \models \bigvee D$, it must be $\rho \models \phi \wedge \psi$, hence $\rho \models \psi$. From this, applying Proposition 4.5.2 to the transition $S \overset{c(t),\psi}{\longmapsto} S'$ we get $S\rho \xrightarrow{c(r)} S'\rho$, where $r = \rho(t)$. Since it also holds that $\rho \models \phi$ and $S \sim^{\phi} R$, it follows that $S\rho \sim R\rho$, hence for some R_1 we have

$$R\rho \xrightarrow{c(r)} R_1 \sim S'\rho \, . \tag{2}$$

Applying Proposition 4.5.1 to the above transition, we get a symbolic transition

$$R \overset{c(t),\chi}{\longmapsto} R', \text{ where } R_1 = R'\rho \text{ and } \rho \models \chi \, . \tag{3}$$

We now show that (a) $\zeta \models \chi$, and that (b) $S'\mathcal{R}^{\zeta}R'$, which will conclude the proof. As to (a), first note that $v(\chi) \subseteq \Sigma(V) \cup \{t\}$; since ζ is complete for $V \cup \{t\}$ and ρ satisfies both ζ and χ (from (3)), it follows that $\zeta \models \chi$. As to (b), from (2) and (3), we deduce that $S'\rho \sim R'\rho$; hence, noting that ζ is complete over $v(S', R')$ (indeed it is $v(S', R') \subseteq_{fin} \Sigma(V) \cup \{t\}$), and that $\rho \models \zeta$, we can apply Theorem 4.7 and deduce that $S' \sim^{\zeta} R'$. Finally, by hypothesis ζ is bounding for $fvar(S', R') \subseteq fvar(S, R) \cup \{t\}$. These facts imply that $S' \mathcal{R}^{\zeta} R'$. $\square$

Putting together the two preceding theorems, we obtain the result we are most interested in, the symbolic characterization of $\sim$:

Corollary 4.10 (Symbolic characterization of $\sim$) *Given two closed configuration S and R, $S \sim R$ if and only if $S \simeq^{true} R$.*

As a corollary of consistency and adequacy and of Theorem 4.7, it is not hard to prove decidability of $\sim$ over finite configurations.

Corollary 4.11 *Let S and R be closed configurations without agent identifiers. Then it is decidable whether $S \sim R$.*

We want to argue that the obtained symbolic characterization of $\sim^{\phi}$ is independent of any specific syntax and local/global-clock assumption for timed processes. The proofs of Theorems 4.8 and 4.9 depend only on the logic BF and the properties of the operational semantics stated in Proposition 4.5 and Theorem 4.7. The proof of the latter theorem depends, in turn, only on BF and Proposition 4.5. Therefore, Theorems 4.8 and 4.9 would still hold for any pair of (bounded-delay) rational-timed and symbolic transition systems, $\mathcal{T}$ and $\mathcal{ST}$, for which the statement of Proposition 4.5 is true (this of course implies that $\mathcal{T}$ and $\mathcal{ST}$ are equipped with reasonable notions of free variables and environments).

5 A Proof System

We present a proof system for reasoning about $\simeq^{\phi}$ on finite (not containing agent identifiers) configurations. The system is based on a syntax of *extended* configurations, $\mathcal{EC}$, which may involve an new kind of action prefix, $c(t, \phi)$:, with $c \in Act \cup \{\tau_\delta \,|\delta \in \mathbf{Q}^+\}$ and $\phi \models t \leq e$ for some e not containing t. The latter prefix is used to express

Axioms

Summation Laws

$(S1)$	$S + \mathbf{0}$	$=$	S
$(S2)$	$S + S$	$=$	S
$(S3)$	$S + R$	$=$	$R + S$
$(S4)$	$S + (R + T)$	$=$	$(S + R) + T$

Restriction Laws

$(R1)$	$(S + R)\backslash a$	$=$	$S\backslash a + R\backslash a$
$(R2)$	$(c(t,\phi) : P)\backslash a$	$=$	$c(t,\phi) : (P\backslash a)$, if $a \neq c$
$(R3)$	$(a(t,\phi) : P)\backslash a$	$=$	$\mathbf{0}$

Clock Laws

$(C1)$	$(a(t,\phi).P) \bullet e$	$=$	$a(t, \phi \wedge t \geq e) : (P \bullet (t + \Delta(a)))$, if $t \notin var(e)$
$(C2)$	$(wait_\delta.P) \bullet e$	$=$	$\tau_\delta(t, t = e) : (P \bullet (t + \delta))$, for t fresh

Boolean Guard Law

(G)	$\phi\, c(t,\psi) : S$	$=$	$c(t, \psi \wedge \phi) : S$, if $t \notin var(\phi)$

Expansion Law

Let $S \equiv \sum_{i \in I} \mu_i : S_i$, $R \equiv \sum_{j \in J} \nu.R_j$ and, for each i, j, $bvar(\mu_i) = bvar(\nu_j) = t$ fresh:

$$(E)\ S \mid R = \sum_{i \in I} \mu_i : (S_i \mid R) + \sum_{j \in J} \nu_j : (S \mid R_j) + \sum_{\mu_i = a(t,\phi_i),\ \nu_j = \bar{a}(t,\psi_j)} \tau_{\Delta(a)}(t, \phi_i \wedge \psi_j) : (S_i \mid R_j)$$

Table 3: Axioms of the proof system on finite extended configurations.

a convenient form of expansion theorem. The operational, concrete and symbolic, rules for the new prefix are, respectively:

$$\texttt{Pre}\frac{}{c(t,\phi) : S \xrightarrow{c(r)} S[r/t]},\ [\![\phi[r/t]]\!] = true \qquad \text{and} \qquad \texttt{S}-\texttt{Pre}\frac{}{c(t,\phi) : S \overset{c(t),\phi}{\longmapsto} S}.$$

All notions and definitions that hold for $\mathcal{C}$, including standard and symbolic bisimulation, conservatively extend to $\mathcal{EC}$ in the expected manner. The consistency and adequacy theorems of symbolic bisimulation of Section 4 are still valid in the new more general setting. In a different timed setting, configurations similar to ours were considered in [2].

In the sequel, $R, S, \ldots$ will range over finite extended configurations. The proof system consists of a set of axioms and inference rules, presented in Table 3 and 4 (standard rules for reflexivity, symmetry and transitivity of $=$ omitted to save space). The statements derivable within the proof system are of the form $\phi \vdash S = R$, to be read as: *under* ϕ, S *equals* R. The notation $c : S$ is used to denote the configuration $c(t, t = 0) : S$, whenever $t \notin fvar(S)$ (the specific value 0 is not actually relevant). We will also use $\equiv$ to denote syntactical equality, as opposed to proof-theoretic equality $=$.

Correctness and completeness of the system are proven by exploiting the consistency and adequacy theorems (Th. 4.8 and 4.9). The proofs are not too different from the corresponding proofs for the value-passing case in [6]. However, additional complications are introduced here by the fact that the instantiation of the parameter t in a prefix $a(t,\phi)$ is constrained by ϕ, whereas in a value-passing setting the instantiation is not constrained (prefixes are of the form $a(t)$). We omit our proofs due to lack of space.

Theorem 5.1 (Correctness and completeness of the proof system) *Let ϕ be a bounding formula for $fvar(S,R)$. Then $\phi \vdash S = R$ if and only if $S \simeq^\phi R$.*

Inference Rules

$(Congr)\quad \dfrac{\phi \vdash S = R}{\phi \vdash S' = R'}$ where $S' = R'$ stands for either of: $c : S = c : R$, $\psi S = \psi R$, $S + T = R + T$, $S|T = R|T$, $S\backslash a = R\backslash a$.

$(Pre)\quad \dfrac{\phi \wedge (t \leq r_0) \vdash \sum_{i\in I} \phi_i c : S_i = \sum_{j\in J} \psi_j c : R_j}{\phi \vdash \sum_{i\in I} c(t, \phi_i) : S_i = \sum_{j\in J} c(t, \psi_j) : R_j}$ if $t \notin var(\phi)$ and for each i and j: $\phi \wedge \phi_i \models t \leq r_0$ and $\phi \wedge \psi_j \models t \leq r_0$

$(Guard)\quad \dfrac{\phi \wedge \psi \vdash S = R,\ \phi \wedge \neg\psi \vdash R = \mathbf{0}}{\phi \vdash \psi S = R}$ $\qquad (False)\quad \dfrac{-}{false \vdash S = R}$

$(Cut)\quad \dfrac{\phi_1 \vdash S = R,\ \phi_2 \vdash S = R}{\phi \vdash S = R}$ if $\phi \models \phi_1 \vee \phi_2$ $\qquad (Axiom)\quad \dfrac{-}{true \vdash S = R}$ for each axiom $S = R$

Table 4: Inference rules of the proof system on finite extended configurations.

6 Conclusions an Related Work

We have proposed a theory of symbolic bisimulation for an algebra of timed processes, that yields a more tractable characterization of bisimilarity. For future work, we regard as very promising the development of verification algorithms for timed symbolic bisimulation, in the style of the algorithms for value-passing of [6].

Our work is mainly related to [6] and [2]. In [6], a theory of symbolic bisimulation has been developed in a setting where values on transitions represent input/output data, rather than time-stamps. Technically, the basic difference between our work and [6], is as follows. Hennessy and Lin work within a general framework, with no specific language for formulae and values. However, in order to find a suitable decomposition when proving their adequacy theorem, they assume the existence of a very powerful logic. Here, we had to deal with a *specific* and non-trivial logic (BF), which lacks the property required in [6]. As a consequence, a substantially different development has to be undertaken to prove, e.g., that a suitable decomposition always exists. Additional care (such as restricting to rationales and to bounded-delay actions) is needed to ensure that the decomposition can always be chosen finite.

In [2], Aceto and Murphy put forward a timed process algebra with "eager" actions, which must occur as soon as possible. As a consequence, finitary transition systems are obtained; this makes it possible to re-use familiar techniques and devise, for instance, complete axiomatizations of equivalences.

Our work is also related to a series of papers (mainly [3, 5, 8, 4]), where the notion of occurrence of an action within a time interval is considered. In [3], Baeten and Bergstra introduces a timed version of ACP that incorporates real-time actions, a global-clock assumption and also relative delays. Bisimilarity over this language has been given an effective axiomatization by Fokkink and Klusener [5]. The problem of devising a finitary representation of process behaviours is not tackled though, and the proof relies therefore on rather ad-hoc techniques. One of the first timed versions of

CCS has been proposed in [8]. Conditional axiomatizations of bisimilarity for (variants of) Wang's language have been obtained by Chen [4]. The languages by considered by Chen are less expressive than ours: for example, relative delays are not expressible.

Alur and Dill have proposed *timed automata* [1], whose transitions are equipped with time constraints on "clock variables". Their setting is quite different from ours, in that automata are compared on the basis of the (timed) traces they accept, and fairness requirements are imposed on the accepted traces.

Acknowledgments

I would like to thank Flavio Corradini for discussions on the topics of the paper and Giuseppe Melfi and Luca Trevisan for technical suggestions. Rocco De Nicola read a preliminary draft, suggesting several improvements. Two anonymous referees provided valuable comments.

References

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

[2] L. Aceto and D. Murphy. On the ill-timed – but well-caused. In E. Best, editor, *Proceedings of CONCUR '93, LNCS 715*. Springer-Verlag, Berlin, 1993. Full version to appear in *Acta Informatica*.

[3] J.C.M. Baeten and A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3:142–188, 1991.

[4] L. Chen. Axiomatizing real-timed processes. In S. Brooks, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Proceedings of MFPS'93, LNCS 802*, pages 215–229. Springer-Verlag, Berlin, 1993.

[5] W.J. Fokkink and S. Klusener. An effective axiomatization for real time ACP. Technical Report CS-R9542, CWI, Computer Science, 1995. To appear in *Information and Computation*.

[6] M. Hennessy and H. Lin. Symbolic bisimulations. *Theoretical Computer Science*, 138:353–389, 1995.

[7] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[8] Y. Wang. Real time behaviour of asynchronous agents. In J.C.M. Baeten and J.W. Klop, editors, *Proc. 1st Conference on Concurrency Theory (CONCUR'90), LNCS 458*. Springer-Verlag, Berlin, 1990.