Chapter 1

GLOBAL TIMED BISIMULATION: AN INTRODUCTION

David de Frutos-Escrig, Natalia López, and Manuel Núñez
Dept. de Sistemas Informáticos y Programación
Universidad Complutense de Madrid, Spain
{defrutos,nlopezb,manuelnu}@eucmax.sim.ucm.es

Abstract Bisimulations are a broadly used formalism to define the semantics of process algebras. In particular, by means of weak bisimulation most of the internal activity of processes may be abstracted. Unfortunately, this is not fully accomplished: for instance, the internal choice operator becomes non-associative since bisimulation can see the branching structure of processes. In this paper we propose *global timed bisimulation* as a weakening of weak timed bisimulation. Global timed bisimulation is defined exactly as weak timed bisimulation once ordinary transitions are replaced by the adequate notions of generalized transitions. In order to asses the definition of our global timed bisimulation we present a collection of small examples that illustrate each of the clauses of that definition. Finally, a more elaborated example is presented to summarize the main properties of that notion.

Keywords: Timed Systems; Formal Semantics; Bisimulations; Non-determinism.

1. INTRODUCTION

Operational semantics is probably the simplest and more intuitive way to give semantics to any process language. It is usually defined by means of a *labeled transition system* which contains the transitions, $P \xrightarrow{l} P'$ that each process can execute. But these semantics are not abstract enough since they contain the full information about the syntax of processes. *Bisimulations* [9] are the usual way to solve this problem, by defining a slightly weaker equivalence which is a rather reasonable, simple and nice way to define the semantics of non-deterministic processes.

Strong bisimulation is a rather satisfactory way to define the semantics of processes whose transitions are labeled by visible actions.

Nevertheless, processes having all their transitions visible are not sufficiently flexible: we need some kind of internal transitions to reflect internal behavior of processes that will be neither observable nor controllable from outside. In particular, internal transitions are introduced to reflect internal choices at the operational level. Therefore, when we define the operational semantics of a process language we usually use labeled transitions, whose labels are visible actions, and unlabeled ones which correspond to internal moves. But these internal actions must be treated in a special way in order to capture the fact that they are not observable. This is why a weaker notion of bisimulation is introduced: *weak bisimulation*.

Weak bisimulation partially abstract internal moves, since in its definition whenever we have the possibility to execute an internal transition we can freely choose between executing one, several or even none of them. As a consequence, using CCS-syntax, we have, for instance $\tau\,;P \sim P \sim \tau\,;\tau\,;P$. Nevertheless, weak bisimulation must not ignore internal transitions when they occur in the scope of a choice. Thus we have $P+Q \not\sim P+\tau\,;Q$, which is indeed rather reasonable. But, unfortunately, this capability to *see* internal actions remains when nested choices appear. So, we have $\tau;(\tau;P+\tau;Q)+\tau;R \not\sim \tau;P+\tau;(\tau;Q+\tau;R)$, which in a CSP-like language would lead us to $(P\oplus Q)\oplus R \not\sim P\oplus(Q\oplus R)$, where $\oplus$ represents the internal choice operator. The fact that internal choice is not associative under weak bisimulation is, in our opinion, a serious drawback. Erdogmus et al. [5] have found the same problem, although they have tried to solve it in a different way to the one we present in this paper. They consider it is not the notion of bisimulation that is *wrong*, but instead what is wrong is the operational semantics from which the bisimulation equivalence is defined. Therefore, they suggest to use a more involved notion of transition system to define the operational semantics of non-deterministic processes.

Instead we prefer to use original transitions systems. We maintain the original operational semantics, given by ordinary transition systems as the basis to define our new semantics although we do not directly use these transitions in the definition of bisimulations. So in [3] we define *global bisimulations* by combining two different kind of moves: *dynamic*, which correspond to the execution of visible actions; and *static*, which correspond to (partial) resolutions of non-determinism. These last moves justify our name for the new notion of bisimulation, since the involved transitions do not need to appear at the beginning of the computations, but anywhere else.

In this paper we extend our previous study to the framework of timed processes. We consider that this is interesting for several reasons. First, by means of this extension we have seen that our ideas are rather flexible and can be satisfactorily applied to different kinds of process algebras. Second, given that one of our goals was to study and relate the different kinds of non-determinism, timed processes are a good benchmark for it, since timed transitions are a new source of non-determinism behaviour. Finally, because the alternative ways to define the semantics by means of weaker equivalence relations than weak bisimulation, do not work as expected in the timed case. For instance, testing semantics is much more stronger than in the untimed case, since the urgency of internal actions strongly increases the discriminatory power of tests [8]. So the definition of alternative semantics is specially appealing in the case of timed processes. Actually, the new semantic framework described in this paper can be applied to the definition of the semantics of the core of E-LOTOS [11] where time information has been included in order to remove one of the limitations of former LOTOS.

The rest of the paper is structured as follows. In Section 2 we present our formalism to describe timed transition systems. We also include several examples showing the expressive power of our model. In Section 3 we define our notion of global timed bisimulation and we show its advantages with respect to weak bisimulation. In Section 4 we summarize the main ideas presented in the paper about the treatment of non-determinism by studying possible implementations of a faulty channel. Finally, in Section 5 we present our conclusions and some lines for future work.

2. BASIC CONCEPTS

In this section we present the abstract syntax which we will use to model timed systems. As we said in the introduction, processes will be defined in terms of (timed) *labeled transition systems*. By using such a *syntax* we can abstract the features of a chosen language. In other words, instead of defining an operational semantics over a fix language, and then to study the induced labeled transition systems, we directly consider a (general) kind of labeled transition systems. In particular, our labeled transitions systems are very close to those induced by the semantics of the new version of E-LOTOS [11].

Definition 1 (**Preliminaries**) We consider a *set of visible actions* Act, and an action τ not belonging to Act. To model time we use a *discrete set of times* $\mathcal{T}$. We call $\mathcal{T}$ the *time domain*. We may consider $\mathcal{T} = \{\delta, 2\delta, 3\delta, \ldots\}$, where δ indicates the *minimum amount* of time. □

Once we have introduced both the set of actions and the set of times which will allow us to represent systems, we define processes as *timed labeled transition systems*. Let us note that in the definition of our processes we are taking into account some of the properties that are commonly found in most temporal models like *time determinacy* and *maximal progress* for τ actions.

Definition 2 A process P is defined as a triple $(S, \longrightarrow, s_0)$, where S is a *set of states*, $s_0 \in S$ is the *initial state*, and $\longrightarrow \subseteq S \times (Act \cup \{\tau\} \cup \mathcal{T}) \times S$ is the *transition relation*. We will usually write $s \xrightarrow{\alpha} s'$ instead of $(s, \alpha, s') \in \longrightarrow$. We impose the following restrictions over the relation:

- **Maximal Progress**: Let $s \in S$. Whenever we have $s \xrightarrow{\tau} s'$ then there do not exist $s'' \in S$, $t \in \mathcal{T}$ such that $s \xrightarrow{t} s''$.
- **Time Determinacy**: Let $s \in S, t \in \mathcal{T}$. Then there exist at most a state $s' \in S$ such that $s \xrightarrow{t} s'$.

□

We will usually identify the nodes of these labeled transition systems with the process defined by the corresponding node as initial state. Formally, let $P = (S, \longrightarrow, s_0)$ be a process. If $s \in S$ then we consider P_s as the process $(S, \longrightarrow, s)$.

In order to make easier the presentation and study of the main features of our new semantics, we will only consider in this paper *tree-like* transition systems that are those such that the *underlying* graph is a rooted tree. It is important to note that this does not represent a lose of generality since by unfolding any transition system we can found an equivalent tree-like transition system. This is so, because our definition of global bisimulation does not use in any way the global structure of the underlying. Besides, it is applicable to any arbitrary timed system either finite or infinite. Thus even if we start from a finite state system with an arbitrary net structure which would become an infinite tree when unfolded, this does not represent a problem at all. It is true however that if we were interested on deciding global bisimilarity in an effective way we should maintain the finiteness of systems when studying them. This can be done indeed, but since it is technically a bit involved we will present in a forthcoming paper.

Once we have defined the notation for describing timed systems, we will explain by means of some simple examples the intended meaning of each construction and the expressive power of the model.

First, we have that *time-outs* can be modeled as follows: if we want to specify a system offering actions a and b exactly at time 0, and such

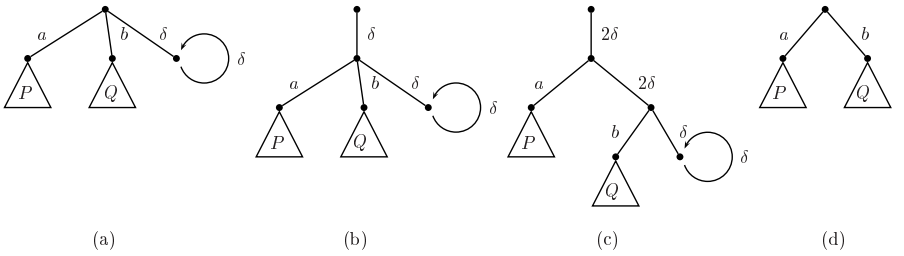

Figure 1.1 Examples of Timed Processes (I).

that when none of these actions is executed it gets deadlocked, we will use the process represented in Figure 1.1 (a).

Delays are specified by means of timed transitions. For instance, if we consider a variation of the previous system where the actions are offered at time δ, it would be represented by the process depicted in Figure 1.1 (b). We can specify processes having different delays. For instance, a choice between the action a delayed 2δ units of time and the action b delayed 4δ units of time would be represented as described in Figure 1.1 (c). Processes *blocking* the time can also be included. For instance, in Figure 1.1 (d) we show a process offering a and b at time 0 that does not allow the passing of time.

Let us remark that in our representation we only make explicit the time transitions corresponding to the time between two offerings of actions represented by states. Obviously, we need to allow a delayed process to *consume* partially its delay (i.e. to *age*). For example, the process $\mathtt{delay}(t);P$ will evolve by a timed transition to the process $\mathtt{delay}(t-t');P$ after passing t' units of time, for any $t' < t$. This will be formalized later.

We can also specify systems offering actions during an interval of time. For example, if the system offers a in the interval $[0..2\delta]$ and b in the interval $[\delta..3\delta]$ we would use the process given in Figure 1.2 (a). Note that τ actions are urgent, so it makes no sense to offer a τ over an interval of time. Moreover, we can specify a process offering an action over an infinite amount of time. For instance, the process depicted in Figure 1.2 (b) offers the action a in the interval $[0..\infty)$, the action b in $[\delta..\infty)$, and the action c in $[\delta..2\delta]$.

Next, in order to manage our processes we introduce some auxiliary notation that will be intensively used all along the rest of the paper.

Definition 3 Let $P = (S, \longrightarrow, s_0)$ be a process. Let $\{s_{a_i} \in S \mid s_0 \xrightarrow{a} s_{a_i}\}$ be the set of states that are reached by executing $\xrightarrow{a}$ from s_0, and let $\{P_{a_i}\}$ be the set of the corresponding induced processes. Similarly,

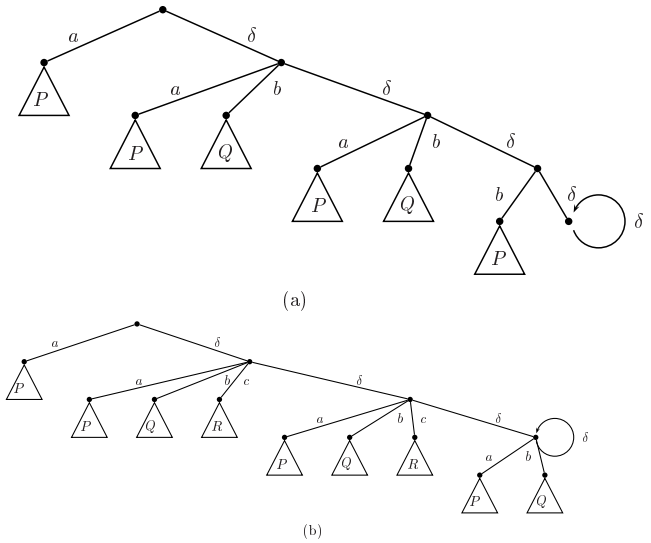

Figure 1.2 Examples of Timed Processes (II).

let $\{s_{\tau_i} \in S | s_0 \xrightarrow{\tau} s_{\tau_i}\}$ (resp. $\{P_{\tau_i}\}$) be the set of states (resp. processes) that are reached after the execution of $\xrightarrow{\tau}$ from s_0. Then to refer to P, we will use the following notation:

- $P = \sum_{a \in Act} a\,;P_{a_i} + \sum \tau\,;P_{\tau_i}$ if $\{P_{\tau_i}\} \neq \emptyset$
- $P = \sum_{a \in Act} a\,;P_{a_i} + \texttt{delay}(t)\,;P'$ if $\{P_{\tau_i}\} = \emptyset \wedge \exists P' :\ P \xrightarrow{t} P'$
- $P = \sum_{a \in Act} a\,;P_{a_i}$ if $\{P_{\tau_i}\} = \emptyset \wedge\ \nexists P' :\ P \xrightarrow{t} P'$
- $P = \texttt{stop}$ if $S = \{s_0\}\ \wedge\ \forall a \in Act :\ \{P_{a_i}\} = \emptyset$ $\wedge\ \{P_{\tau_i}\} = \emptyset\ \wedge\ s_0 \xrightarrow{\delta} s_0$
- $P = \texttt{block}$ if $S = \{s_0\}\ \wedge\ \forall a \in Act :\ \{P_{a_i}\} = \emptyset$ $\wedge\ \{P_{\tau_i}\} = \emptyset\ \wedge\ \nexists t \in \mathcal{T} :\ s_0 \xrightarrow{t} s_0$

□

Note that some of the sums $\sum_{a \in Act} a\,;P_{a_i}$ may be empty. Additionally, we will also write $P = \sum_{\alpha} \alpha\,;P_{\alpha}$ when we do not need to distinguish between visible, internal and timed descendants, and we will use $P_1 + P_2 + \cdots + P_n$ as a shorthand of $\sum P_i$.

In the previous definition we have introduced two new concepts: the processes `stop` and `block`. Both processes are very similar because they cannot execute any action. The difference is that while `stop` allows the time to pass, `block` does not. Next, we define the usual extensions of the transition relation to allow several steps.

Definition 4 Let $P = (S, \longrightarrow, s_0)$ be a process. We define the following relations on states:

- We denote by $\stackrel{\tau}{\Longrightarrow}$ the transitive and reflexive closure of $\stackrel{\tau}{\longrightarrow}$.
- Let $s, s' \in S$. We write $s \stackrel{t}{\Longrightarrow} s'$ if there exist $t_1, t_2, \ldots, t_n \in \mathcal{T}$ such that $\sum t_i = t$ and there exist $s_1, s'_1, s_2, s'_2, \ldots, s_n, s'_n$ such that $s \stackrel{\tau}{\Longrightarrow} s_1 \stackrel{t_1}{\longrightarrow} s'_1 \stackrel{\tau}{\Longrightarrow} s_2 \stackrel{t_2}{\longrightarrow} s'_2 \cdots \stackrel{\tau}{\Longrightarrow} s_n \stackrel{t_n}{\longrightarrow} s'_n \stackrel{\tau}{\Longrightarrow} s'$
- Let $s, s' \in S$. We write $s \stackrel{a}{\Longrightarrow} s'$ if there exist s_1, s_2 such that $s \stackrel{\tau}{\Longrightarrow} s_1 \stackrel{a}{\longrightarrow} s_2 \stackrel{\tau}{\Longrightarrow} s'$.

□

As commented above, we need to generate all the different time transitions that correspond to each of the ones explicitly represented in the given transition system. So we extend the transition relations as follows: Given a process $P = \sum_{a \in Act} a\,;P_{a_i} + \texttt{delay}(t)\,;P'$ in addition to the transitions that are allowed by its transition relation, it has the following timed transitions: $P \stackrel{t'}{\longrightarrow} \texttt{delay}(t - t')\,;P' \quad [\text{if } t' < t]$

We conclude this section by defining the adequate notion of weak timed bisimulation.

Definition 5 A binary relation $\mathcal{R}$ on processes is called a *weak timed bisimulation* if for all P, Q such as $P\ \mathcal{R}\ Q$, and any $\alpha \in Act \cup \{\tau\} \cup \mathcal{T}$:

- Whenever $P \stackrel{\alpha}{\longrightarrow} P'$ we have some Q' such that $Q \stackrel{\alpha}{\Longrightarrow} Q'$ and $P'\ \mathcal{R}\ Q'$.
- Whenever $Q \stackrel{\alpha}{\longrightarrow} Q'$ we have some P' such that $P \stackrel{\alpha}{\Longrightarrow} P'$ and $P'\ \mathcal{R}\ Q'$.

We say that two processes P and Q are *weakly timed bisimilar* if there exists a weak timed bisimulation containing the pair $\langle P, Q \rangle$. □

3. A GLOBAL TIMED BISIMULATION

In this section we will define a timed extension of *global bisimulations* [3]. We will denote by $\rightsquigarrow$ and $\stackrel{\gamma}{\rightsquigarrow}$, with $\gamma \in Act \cup \mathcal{T}$, the *static* and *dynamic* generalized transitions to be defined below.

Definition 6 Let P be a process. We say that P may evolve by a *static* transition to P', denoted by $P \rightsquigarrow P'$, and we say that P may evolve by a *dynamic* transition to P' after $\alpha \in Act \cup \mathcal{T}$, denoted by $P \stackrel{\alpha}{\rightsquigarrow} P'$, if the corresponding transition can be derived from the following rules:

1. *Preserving the transition relation of P*:

- If $P \stackrel{\tau}{\longrightarrow} P'$ then $P \rightsquigarrow P'$.
- If $P \stackrel{\alpha}{\longrightarrow} P'$ for some $\alpha \in Act \cup \mathcal{T}$ then $P \stackrel{\alpha}{\rightsquigarrow} P'$.

2. *Simultaneous execution of several internal transitions*:

$$P = \sum_{\alpha} \alpha \,;\, P_{\alpha_i} + \sum_{i \in I} \tau \,;\, P_i \rightsquigarrow \sum_{i \in I} \tau \,;\, P_i$$

3. *Removal of non-determinism* (generated by simultaneous offerings of the same visible action):

$$P = \sum_{\alpha} \alpha \,;\, P_{\alpha_i} + a \,;\, P_a + \sum_{i \in I} a \,;\, P_i \rightsquigarrow \sum_{\alpha} \alpha \,;\, P_{\alpha_i} + a \,;\, P_a$$

4. *Forward execution of an internal move*:

$$P_\beta \rightsquigarrow P'_\beta \implies P = \sum_{\alpha} \alpha \,;\, P_{\alpha_i} + \beta \,;\, P_\beta \rightsquigarrow \sum_{\alpha} \alpha \,;\, P_{\alpha_i} + \beta \,;\, P'_\beta$$

where either $\beta \in Act \cup \{\tau\}$ or $\beta = \mathtt{delay}(t)$ for some $t \in \mathcal{T}$.

5. *Simultaneous execution of several instances of an offered action*:

$$\forall a \in Act, \forall I \subseteq I_a \text{ with } I_a = \{\!|\, i \mid P \stackrel{a}{\Longrightarrow} P_i \,|\!\} \implies P \stackrel{a}{\rightsquigarrow} \sum_{i \in I} \tau \,;\, P_i$$

6. *Synchronous passing of time under τ's*:

$$\forall i \in I : \; P_i \stackrel{t}{\approx\!>} P'_i \implies P = \sum_{a \in Act} a \,;\, P_{a_i} + \sum_{i \in I} \tau \,;\, P_i \stackrel{t}{\rightsquigarrow} \sum_{i \in I} \tau \,;\, P'_i$$

where $\approx\!>$ represents the reflexive and transitive closure of $\rightsquigarrow$, and we have $P \stackrel{t}{\approx\!>} P'$ where $P \approx\!> \cdot \stackrel{t_1}{\rightsquigarrow} \cdot \cdots \cdot \stackrel{t_n}{\rightsquigarrow} \cdots \approx\!> P'$ with $t = \sum t_i$. Finally, for any $a \in Act$, we take $\stackrel{a}{\approx\!>}$ equal to $\approx\!> \cdot \stackrel{a}{\rightsquigarrow} \cdot \approx\!>$.

□

In Figure 1.3 we graphically present the rules given in the previous definition. Let us briefly comment the intuitive meaning of each of these rules, Rule 1 is included to keep the ordinary transitions of processes. Rule 2 allows a process to evolve by selecting a subset of its τ descendants (also note that in this rule some of the α's may be equal to τ). Rule 3 allows to reduce nondeterminism by cutting some of the branches labeled by the same offered action (note here that some of the α's may also be equal to a). Rule 4 indicates that a process may evolve by executing a static transition not only at its root but at any point. One can see that by iterating the application of this rule a static transition can

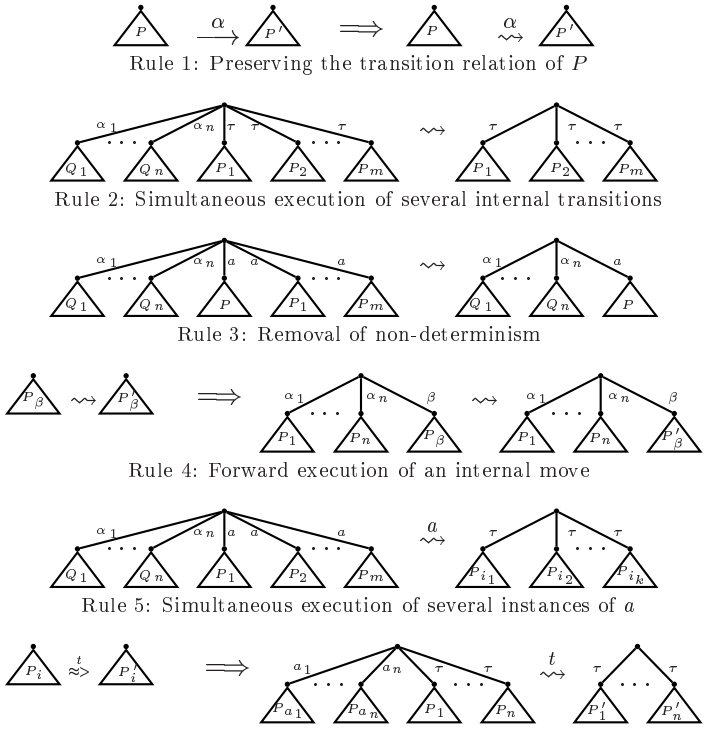

Figure 1.3 Static and dynamic transitions

be done not only after a single step but also after any sequence of them. We also need two additional rules for generating dynamic transitions. Rule 5 allows us to execute not just a single occurrence of an action, but a subset of them, by non-deterministically choosing between them, what is done by composing the reached states by prefixing any of them by a τ action. Finally, Rule 6 allows a process to age under τ actions, thus delaying the corresponding internal choice. Note that in this case all the other branches are lost.

Finally, we introduce our notion of *global timed bisimulation* which follows the same pattern that weak bisimulation, but replacing usual transitions by static and dynamic transitions.

Definition 7 A binary relation $\mathcal{R}$ between pairs of processes is a *global timed bisimulation* iff for all P, Q such that $P\ \mathcal{R}\ Q$ and any $\alpha \in Act \cup \{\epsilon\} \cup \mathcal{T}$ we have:

- Whenever $P \overset{\alpha}{\rightsquigarrow} P'$ we have some Q' with $Q \overset{\alpha}{\approx\!>} Q'$ and $P'\ \mathcal{R}\ Q'$.

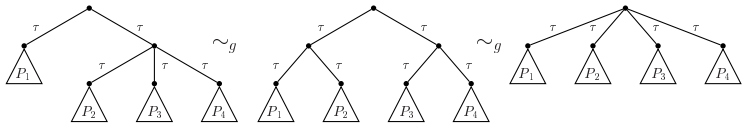

Figure 1.4 Associativity of pure τ-choices.

- Whenever $Q \overset{\alpha}{\rightsquigarrow} Q'$ we have some P' with $P \overset{\alpha}{\approx\!>} P'$ and $P' \mathcal{R} Q'$.

We say that two processes P and Q are *globally timed bisimilar*, and we write $P \sim_g Q$, if there exists a global timed bisimulation containing the pair $\langle P, Q \rangle$. □

It can be checked that weak timed bisimulation identifies less processes that global timed bisimulation. The proof is an adaptation of that given in [3] for the untimed case.

Theorem 8 Let P and P' be two processes. We have that $P \sim P'$ implies $P \sim_g P'$.

On the other hand global bisimulation is still a rather strong relation. We can prove in fact that it is stronger than the testing equivalence(see [3]), and as a consequence it is strong enough to detect deadlocks. Finally it is interesting to note that the relations $\overset{\alpha}{\rightsquigarrow}$ give us a derived timed labeled transition systems which defines an alternative operational semantics of our processes. In such a case global bisimulation becomes plain weak bisimulation, and as a consequence all the results for weak bisimulation could be extended to global bisimulation, although relative to that (a bit strange and complicated) derived operational semantics. In particular we can prove that if the original transition system is finite, the derived system can be presented also as a finite state system. Thus global timed bisimulation becomes decidable for finite state system, and the algorithms to decide ordinary bisimulation can be applied to decide global bisimilarity.

3.1 ASSOCIATIVITY OF PURE τ-CHOICES

As we already explained, one of the motivations of our work was to develop a notion of observational semantics based on bisimulation where the internal choice operator were associative. So, considering our definition of global timed bisimulation we have the situation described in Figure 1.4.

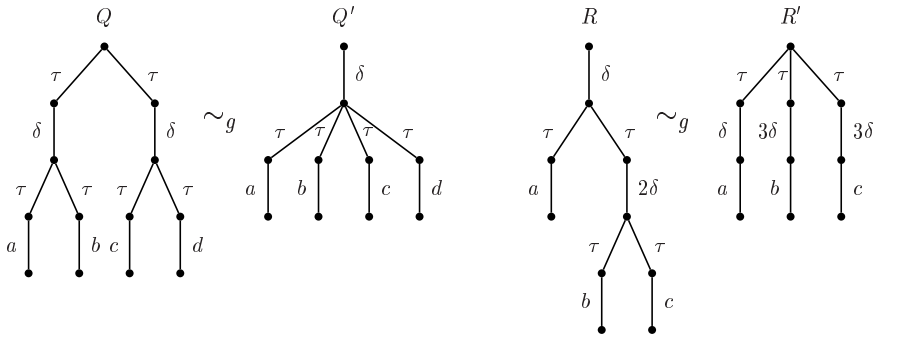

Figure 1.5 Associativity in the presence of time.

To see how the notions of generalized transitions have been extended in order to get associativity, we will consider the following processes:
$P = \tau\,;(\tau\,;P_1 + \tau\,;P_2) + \tau\,;P_3 \qquad P' = \tau\,;P_1 + \tau\,;(\tau\,;P_2 + \tau\,;P_3)$
$$P'' = \tau\,;P_1 + \tau\,;P_2 + \tau\,;P_3$$
Let us study which of the *capabilities* that we have added to the usual transitions relation allow us to get $P \sim_g P'$ and $P' \sim_g P''$.

In order to show that P and P' are globally timed bisimilar, if we want to simulate the transition $P \rightsquigarrow \tau\,;P_1 + \tau\,;P_2$, it is necessary to apply Rule 4 to P' to execute a τ action under another one, so we have $\tau\,;P_2 + \tau\,;P_3 \xrightarrow{\tau} P_2$, and thus we obtain $P' \rightsquigarrow \tau\,;P_1 + \tau\,;P_2$.

On the other hand, in order that P'' will be able to simulate the same transition of P we apply Rule 2 to P'': by simultaneously executing the two internal transitions $P'' \xrightarrow{\tau} P_1$ and $P'' \xrightarrow{\tau} P_2$, without resolving the choice between them, P'' evolves into $\tau\,;P_1 + \tau\,;P_2$.

Let us now study associativity in the case where *timed decisions* are taken at different places. Let us consider the processes Q and Q' given in Figure 1.5. Let us show that under global bisimulation they are equivalent. We have that when Q executes the transition $Q \rightsquigarrow \delta\,;(a+b)$, Q' can simulate it by executing at the same time two internal actions without aging, that is by applying Rules 2 and 4, obtaining $Q' \rightsquigarrow \delta\,;(\tau\,;a + \tau\,;b)$. On the other hand, in order to let Q to simulate the timed transition $Q' \xrightarrow{\delta} \tau\,;a + \tau\,;b + \tau\,;c + \tau\,;d$, it must be aged without resolving the choice, to obtain $Q \overset{\delta}{\rightsquigarrow} \tau\,;(\tau\,;a + \tau\,;b) + \tau\,;(\tau\,;c + \tau\,;d)$. Note that this transition is allowed by Rule 6.

Consider now the processes R and R' in Figure 1.5. We have that these two processes are globally timed bisimilar. This is a natural consequence of our theory, because we abstract away from *when* the decisions are

taken. We have that when R performs the transition $R \stackrel{\delta}{\rightsquigarrow} R_1 = (\tau\,;\, a) + (\tau\,;\, \mathtt{delay}(2\delta)\,;\, (\tau\,;\, b + \tau\,;\, c))$. Indeed, R' should be aged, by the same amount of time in every branch. By applying Rule 6 we obtain $R' \stackrel{\delta}{\rightsquigarrow} R_1' = \tau\,;a+\tau\,;\mathtt{delay}(2\delta)\,;b+\tau\,;\mathtt{delay}(2\delta)\,;c$. At this point, R_1 has two possibilities: on the one hand, if it decides to perform the transition $R_1 \stackrel{\tau}{\longrightarrow} a$, R_1' can execute the same transition to simulate it; on the other hand, if R_1 performs the transition $R_1 \stackrel{\tau}{\longrightarrow} \mathtt{delay}(2\delta)\,;\,(\tau\,;\,b+\tau\,;\,c)$, we have the same situation that in the previous example.

3.2 DIFFERENT KINDS OF SYMMETRIC NON-DETERMINISM

Once we have shown that we have gained the ability to resolve several internal choices at the same time, thus getting associativity, we fix our attention on the alternative ways to generate non-determinism. In particular, an implicit internal choice is generated when we have an external choice between two processes offering some common visible actions. Under weak bisimulation this kind of non-determinism is considered to be different to that generated by internal choices. More exactly, if we consider two processes P_1, P_2 that are not weak bisimilar, we have that the process $a;P_1+a;P_2$ is not equivalent to any process that does not contain an external choice between two processes offering the action a. It is easy to check that this result does not hold when we consider global timed bisimulation. Next we consider the following three simple processes:

$P_1 = a\,;\, b + a\,;\, c \qquad P_2 = \tau\,;\, a\,;\, b + \tau\,;\, a\,;\, c \qquad P_3 = a\,;\,(\tau\,;\, b + \tau\,;\, c)$

Let us see that under global timed bisimulation we have $P_1 \sim_g P_2$. We have indeed that $P_2 \stackrel{\tau}{\longrightarrow} a\,;\,b$ can be simulated by removing the branch $a\,;\,c$ of P_1 (applying Rule 3) to get $P_1 \rightsquigarrow a\,;\,b$. We also have $P_1 \sim_g P_3$: in order to simulate the transition $P_3 \stackrel{a}{\rightsquigarrow} \tau\,;\,b+\tau\,;\,c$, P_1 may simultaneously execute its two a actions, by postponing the choice between their continuations, then getting $P_1 \stackrel{a}{\rightsquigarrow} \tau\,;\,b+\tau\,;\,c$ by applying Rule 5. Besides, if we consider the transition $P_2 \rightsquigarrow a;b$ that it is obtained by Rule 1, we have that P_1 can simulate it by means of the transition $P_1 \rightsquigarrow a\,;\,b$, which is generated by applying Rule 3.

3.3 DIFFERENT KINDS OF TIMED SYMMETRIC NON-DETERMINISM

In this section we discuss the problem of symmetric non-determinism when it appears mixed with time. Weak timed bisimulation can correctly deal with some simple interactions between internal situations produced by τ actions and time transitions. For example, we have $\mathtt{delay}(t);P \sim \tau;$

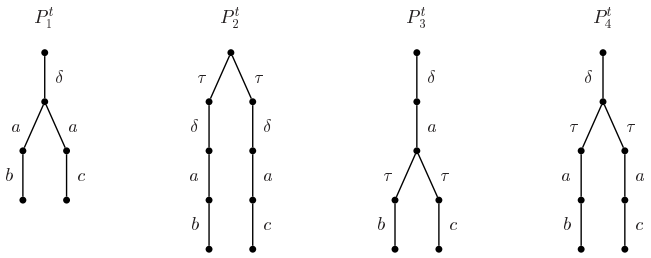

Figure 1.6 Different kinds of non-determinism.

$\texttt{delay}(t);P \sim \texttt{delay}(t);\tau;P$. On the contrary, mixing non-deterministic choices and time in a more complex way produces undesirable results in the context of weak timed bisimulation. For example, the following processes, which are graphically presented in Figure 1.6:

$$P_1^t = (\texttt{delay}(\delta);a;b)+(\texttt{delay}(\delta);a;c) \qquad P_2^t = (\tau;\texttt{delay}(\delta);a;b)+(\tau;\texttt{delay}(\delta);a;c)$$

$$P_3^t = \texttt{delay}(\delta)\,;\,a\,;\,(\tau\,;\,b+\tau\,;\,c) \qquad P_4^t = \texttt{delay}(\delta)\,;\,(\tau\,;\,a\,;\,b+\tau\,;\,a\,;\,c)$$

are not weakly timed bisimilar.

In order to get $P_1^t \sim_g P_2^t$, we have that the transition $P_1^t \xrightarrow{\delta} a;b+a;c$ can be simulated by P_2^t, by applying Rule 6, *aging* it without resolving the non-determinism, to obtain $P_2^t \overset{\delta}{\rightsquigarrow} \tau;a;b+\tau;a;c$. Besides, in order to simulate the transition $P_2^t \xrightarrow{\tau} \texttt{delay}(\delta);a;b$, we have that Rules 3 and 4 allow P_1^t to remove one of the two branches after the timed transitions.

On the other hand, we also have $P_1^t \sim_g P_3^t$ because the transition $P_3^t \xrightarrow{\delta} a\,;\,(\tau\,;\,b+\tau\,;\,c)$ may be simulated by P_1^t just by performing the transition $P_1^t \xrightarrow{\delta} a;b+a;c$. So we obtain the transition systems P_1 and P_3 of the previous section. The same result holds for P_1^t and P_4^t because when one of them performs a timed transition the other can execute the same transition, and in this way they evolve into the processes P_1 and P_2 of the previous section. Besides, P_2^t and P_3^t are globally timed bisimilar, since in order to simulate the transition $P_2^t \xrightarrow{\tau} \delta\,;a;b$, P_3^t can execute a τ transition, by applying Rule 4. Using similar reasonings one can check that the pairs $\langle P_2^t, P_4^t\rangle$ and $\langle P_3^t, P_4^t\rangle$ are global timed bisimilar.

4. EXAMPLE: A FAULTY CHANNEL

In this section we summarize the main ideas presented in this paper by means of a simple example: a faulty channel. We will consider a faulty channel which when retransmitting a received message can either

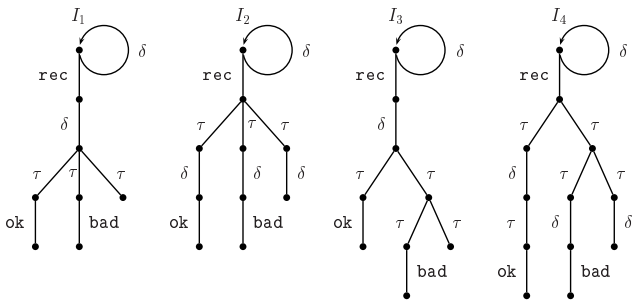

Figure 1.7 Different Implementations of the faulty channel.

correctly send it, or send it but with errors, or lose it. After a more formal specification of the channel we will present several processes which can be considered as correct implementations of the specification. We would like that all of these implementations were equivalent. Nevertheless, they are not under weak timed bisimulation, but they are under our notion of global timed bisimulation.

4.1 THE CHANNEL

We want to get a faulty channel fulfilling the following properties:

1. The channel will eventually receive one message.
2. Then the channel will try to send the message to the final receiver. Three situations can happen:
   - The message is successfully sent to the receiver.
   - The message is sent but the content is corrupted.
   - The message is lost.

   After sending the message the channel gets blocked, that is, we assume that this is a *one-time use* channel.
3. The channel will try to retransmit the message each δ units of time after it was received.

4.2 THE IMPLEMENTATIONS

From the previous specification, the implementer has several choices in order to develop a correct implementation. For example, (s)he can decide to implement the delay between the reception of the message

and its sending either right after the system has taken a decision about losing the message or not, or just before. In Figure 1.7 we present some of the possibilities. There `rec` denotes the reception of the message, and `ok` and `bad` represents the right and the incorrect sending of the message, respectively. Let us see which are the differences between these implementations.

I_1 works as follows: The system waits until a message arrives, then it waits for a time δ. Finally, it decides non-deterministically either to send the message correctly, or incorrectly, or not to send the message. I_2 receives the message and then it non-deterministically decides what will happen with the message. Afterwards, regardless of the previous decision, it waits a time δ, and then it produces the corresponding result.I_3 is a simple variation of I_1. In this case the implementer first distinguishes two cases: either the message is sent correctly or not. If this is the case we have a new non-deterministic choice either to (incorrectly) send the message or to lose it. I_4 presents a similar variation with respect to I_2.

It is clear that all the previous systems are correct implementations of the specification. Additionally, should we consider all these implementations to be equivalent? Under weak timed bisimulation the answer is negative. For example, I_1 and I_2 are not equivalent. Consider the state of I_1 which is reached after executing the action `rec` and allowing to pass a time δ. If we want to simulate this state with I_2, it must first execute `rec` and then execute one of the τ actions, in order to finally allow the passing of time. It is clear that the state reached by I_1 is not weakly timed bisimilar to any of the possible states reached by I_2. We can find a similar situation in the rest of the cases. Actually, for any $i \neq j$ we have $I_i \not\sim I_j$. Nevertheless, it seems that no external observer should be able to distinguish these implementations: all of them can lose or (correctly or incorrectly) send the message, and all of them wait a time δ between the reception and the sending process. Global timed bisimulation solves this problem. Combining the reasonings used in the previous section, it can be proved that for any i, j we have $I_i \sim_g I_j$.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a timed extension of global bisimulation. This new semantics can be considered as a suitable framework to study the semantics of timed non-deterministic processes. Along the paper we have shown that global timed bisimulation equalize different kinds of non-determinism that are distinguished by weak bisimulation. In particular we have shown how to get associativity of pure τ choices, which in our opinion is a very desirable property.

One line for future work consists in relating our notion of global timed bisimulation with other alternative semantics. For the untimed case we have shown in [3] that (must) testing semantics can be alternatively defined as a kind of global bisimulation (where some rules to deal with *mixed* choices must be added). It would be interesting to study which new rules (for generating static and dynamic transitions) must be added to the notion of global timed bisimulation presented in this paper in order to get other semantics, in particular testing-like semantics like *must* and *may* testing [4, 7], or friendly testing [1, 2].

Another work under development consists in dealing with probabilistic information. We want our work to be consistent with LOTOS, so we have taken as starting points [10, 6] where a probabilistic and a probabilistic-timed version of LOTOS have been defined, respectively.

Acknowledgments. This research has been partially supported by the CICYT project TIC 97-0669-C03-01.

References

[1] D. de Frutos-Escrig, L.F. Llana-Díaz, and M. Núñez. Friendly testing as a conformance relation. In *Formal Description Techniques for Distributed Systems and Communication Protocols (X), and Protocol Specification, Testing, and Verification (XVII)*, pages 283–298. Chapman & Hall, 1997.

[2] D. de Frutos-Escrig, L.F. Llana-Díaz, and M. Núñez. An invitation to friendly testing. *Journal of Computer Science and Technology*, 13(6):531–545, 1998.

[3] D. de Frutos-Escrig, N. López, and M. Núñez. Global bisimulations. Technical Report SIP 91/99, Dept. de Sistemas Informáticos y Programación. Universidad Complutense de Madrid, 1999.

[4] R. de Nicola and M.C.B. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[5] H. Erdogmus, R. Johnston, and M. Ferguson. On the operational semantics of nondeterminism and divergence. *Theoretical Computer Science*, 159:271–317, 1996.

[6] C. Gregorio and M. Núñez. Specifying and verifying the Alternating Bit Protocol with Probabilistic-Timed LOTOS. In *COST 247 International Workshop on Applied Formal Methods in System Design*, pages 38–50, 1996.

[7] M. Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.

[8] L. Llana, D. de Frutos, and M. Núñez. Testing semantics for urgent timed algebras. In *3rd AMAST Workshop on Real-Time Systems*, pages 33–46, 1996.

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

[10] M. Núñez and D. de Frutos. Testing semantics for probabilistic LOTOS. In *Formal Description Techniques VIII*, pages 365–380. Chapman & Hall, 1995.

[11] J. Quemada, editor. *Final Committee Draft on enhancements to LOTOS*. ISO/IEC JTC1/SC21/WG7 project 1.21.20.2.3, 1998.