An Axiomatization for Regular Processes in Timed Branching Bisimulation

Wan Fokkink
University of Wales Swansea
Department of Computer Science
Singleton Park, Swansea SA2 8PP, Wales
e-mail: w.j.fokkink@swan.ac.uk

Klusener introduced a timed variant of branching bisimulation. In this paper it is shown that Klusener's axioms for finite process terms, together with two standard axioms for recursion, make a complete axiomatization for regular processes modulo timed branching bisimulation.

1 Introduction

Over the years, process algebras such as CCS, CSP and ACP have been extended with a notion of time. In each of these extensions, the interaction of the silent step τ with time turns out to be deplorably complicated, see e.g. [3, 7, 15, 17, 19, 22, 24, 25]. Klusener [17, 19] extended branching bisimulation [14] to a timed setting. The intuition for branching bisimulation is that this equivalence preserves the branching structure of processes; a τ-transition is silent if and only if it does not lose possible behaviours. Klusener concluded that the same intuition in the timed case gives rise to quite a different mathematical interpretation than in the untimed case. Currently, timed branching bisimulation is used to give semantics to an extension of the specification language μCRL with time [13].

Although the definition of timed branching bisimulation is an intricate one, its axiomatization is crystal clear. One elegant axiom suffices to characterize timed branching bisimulation. Hence, performing calculations with respect to this semantics is quite feasible. In this paper, the domain where calculations can be performed is extended from finite process terms to all regular processes.

The first part of this paper considers Basic Process Algebra with deadlock in relative time, from Baeten and Bergstra [2]. In this process algebra, denoted by $\mathrm{BPA}_{\delta r}$, atomic actions are provided with a time stamp, taken from the rational numbers extended with infinity (to express livelock). A timed action $a[r]$ executes action a after exactly r time units of inaction. In this paper, the algebra $\mathrm{BPA}_{\delta r}$ is extended with recursion.

Milner [20] was the first to derive completeness of an axiomatization for regular behaviours with respect to strong bisimulation in the untimed case. Bergstra and

Klop [6] proved a similar result in a slightly different setting. They introduced two new axioms for recursion, called R1 and R2 (which are basically reformulations of axioms by Milner) and showed that R1,2 together with the standard axioms for BPA_δ suffice to completely axiomatize the algebra of untimed regular processes modulo strong bisimulation. In the first part of this paper it is shown that this result can be transposed to the timed setting. That is, R1,2 together with the standard axioms for the algebra $\mathrm{BPA}_{\delta r}$ are complete for the algebra of timed regular processes. This result is not very shocking; similar results have been obtained by Aceto and Jeffrey [1] in the regular subcalculus of Wang Yi's timed CCS [16, 26], and by Chen [8] in his extension of CCS with real time.

Bergstra and Klop [6] and Milner [21] deduced completeness results for axiomatizations for untimed regular processes with silent steps, in observational congruence. Bergstra and Klop [6, page 51] remark that their result on completeness for regular processes over BPA_δ "is not very surprising" and that they "wish to extend Milner's completeness theorem (from [20]) to the case where τ-steps are present". Similarly, the aim of this paper is to extend existing work on axiomatizations for regular processes in the timed setting to the case where τ-steps are present. Here, we consider the silent τ in Klusener's timed branching bisimulation. The conclusion of this paper is quite different from the one in [6]. There, no less than three extra axioms were needed to deal with complications regarding recursion in the presence of abstraction. This is mainly due to the fact that τ is not a 'guard' for recursion; the recursive equation $X = \tau \cdot X$ has infinitely many solutions $\tau \cdot p$ for X. However, this complication disappears in the presence of time; the recursive equation $X = \tau[r] \cdot X$ has only one solution for X (namely $\tau[r] \cdot \delta[\infty]$). A similar observation was already made by Reed and Roscoe [25], in a setting of timed CSP with topological models. As a consequence of this phenomenon, in the timed setting no extra axioms are needed to describe the interplay of recursion and the silent step. In this paper it is shown that the standard axioms for $\mathrm{BPA}_{\delta r}$, together with Klusener's characterizing axiom for timed branching bisimulation, and the axioms R1,2 for recursion, constitute a complete axiomatization for regular processes in timed branching bisimulation. The proof of this result consists of reducing terms that are timed branching bisimilar to terms that are timed strongly bisimilar, after which the completeness result from the first part of this paper can be applied.

In order to cut down notational overhead, the communication operators and the encapsulation operator from ACP are left out. Although these operators are important for the expressive power of the formalism, they are not essential for presenting the main ideas of this paper. A straightforward collection of axioms from [2], together with axioms R1,2, suffice to eliminate these operators from the syntax, see [9].[1] A stronger elimination result is proved in [10], for a class

[1] For this elimination result it is essential that the time domain consists of the rationals, instead of the reals. For example, if $X = a[1] \cdot X$ and $Y = b[\sqrt{2}] \cdot Y$, then the merge cannot be eliminated from $X \| Y$.

of timed regular processes in a setting with integration [2, 12]. This construct enables to express time dependencies, i.e., the behaviour of a process may depend on the moment in time that a previous action has been executed.

Acknowledgements. This research was carried out at the CWI in Amsterdam. It was initiated by a question from Jan Bergstra. Steven Klusener, Frits Vaandrager and anonymous referees provided valuable comments.

2 Timed Regular Processes

2.1 Basic Process Algebra

We study the formalism $\mathrm{BPA}_{\delta r}$, which stands for Basic Process Algebra with deadlock, extended with relative time. Assume an alphabet A of atomic actions, together with the special constant δ to represent deadlock. In the sequel, a and α denote elements of A and $A \cup \{\delta\}$ respectively. A *timed action* is of the form $\alpha[r]$ with $r \in \mathbb{Q} \cup \{\infty\}$. Here, ∞ denotes a special time element 'infinity' that is greater than any rational number. Moreover, we consider the binary operators alternative composition $+$ and sequential composition $\cdot$. Hence, process terms in $\mathrm{BPA}_{\delta r}$ are built from the following BNF grammar:

$$p \ ::= \ \alpha[r] \mid p + p \mid p \cdot p$$

where $\alpha \in A \cup \{\delta\}$ and $r \in \mathbb{Q} \cup \{\infty\}$.

Processes are considered in relative time. This means that a timed action $a[r]$ executes a after exactly r time units of inaction. For example, the process $a[1] \cdot b[2]$ first executes a at time 1, and then b at (absolute) time 3. Time starts at zero and never reaches infinity, so actions $a[r]$ with $r \leq 0$ or $r = \infty$ do not display any behaviour. The timed deadlock $\delta[r]$ can only idle until (relative) time r, which is illustrated by the following example.

Example 2.1 *The process $a[1] + \delta[2]$ can either execute the a at time 1, or idle until time 2, in which case it gets into a deadlock. On the other hand, the process $a[1] + \delta[1]$ will always execute the a at time 1, so that it never gets into a deadlock.*

The intuitive behaviour of process terms in $\mathrm{BPA}_{\delta r}$ is captured in the operational semantics provided in Table 1. The transition rules are taken from [12, 18]. In contrast with most semantics for timed processes in the literature, there are no idle transitions; a timed action $a[r]$ only executes the a at (relative) time r. A similar operational semantics can be found in [16].

2.2 Recursion

A *recursive specification* E is a finite set of equations $\{X_i = t_i \mid i = 1, ..., n\}$, where the X_i are recursion variables, and the t_i are process terms constructed

$$a[r] \xrightarrow{a[r]} \surd \qquad \text{if } 0 < r < \infty$$

$$\frac{x \xrightarrow{a[r]} \surd}{x + y \xrightarrow{a[r]} \surd \xleftarrow{a[r]} y + x} \qquad \frac{x \xrightarrow{a[r]} x'}{x + y \xrightarrow{a[r]} x' \xleftarrow{a[r]} y + x}$$

$$\frac{x \xrightarrow{a[r]} \surd}{x \cdot y \xrightarrow{a[r]} y} \qquad \frac{x \xrightarrow{a[r]} x'}{x \cdot y \xrightarrow{a[r]} x' \cdot y}$$

Table 1: Transition rules for $\mathrm{BPA}_{\delta r}$

from timed actions, the alternative composition, the sequential composition and the variables X_j for $j = 1, ..., n$. Intuitively, the syntactic construct $\langle X|E\rangle$ denotes a solution of X with respect to E; the precise meaning of this construct is given in Table 2. In the sequel, only *linear* recursive specifications will be considered, which consist of equations of the form

$$X = \sum_i \alpha_i[r_i] \cdot Y_i + \sum_j \beta_j[s_j].$$

Table 2 presents two standard transition rules for recursion. The expression E in these transition rules represents a linear recursive specification, which contains an equation $X = t$. Furthermore, $\langle t|E\rangle$ denotes the term t with occurrences of variables Y replaced by $\langle Y|E\rangle$.

$$\frac{\langle t|E\rangle \xrightarrow{a[r]} \surd}{\langle X|E\rangle \xrightarrow{a[r]} \surd} \qquad \frac{\langle t|E\rangle \xrightarrow{a[r]} y}{\langle X|E\rangle \xrightarrow{a[r]} y}$$

Table 2: Transition rules for recursion

The process terms of $\mathrm{BPA}_{\delta r}$ with recursion are constructed from timed actions and expressions $\langle X|E\rangle$ with E a linear recursive specification, together with the alternative and the sequential composition. So the BNF grammar of process terms is

$$p \ ::= \ \alpha[r] \mid \langle X|E\rangle \mid p + p \mid p \cdot p.$$

The operational semantics for this timed process algebra consists of the transition rules in Table 1 and Table 2.

Definition 2.2 *A process term p' is a* derivative *of process term p if p can evolve into p' in zero or more transitions.*

A process term p' is a proper derivative *of process term p if p can evolve into p' in one or more transitions.*

The process terms in $\mathrm{BPA}_{\delta r}$ with recursion are called regular, which is justified by the fact that each process terms has only finitely many derivatives, and the initial actions of these derivatives together constitute a finite collection (cf [5] for the untimed case).

2.3 Timed Strong Bisimulation

The *ultimate delay* $U(p)$ from [2] is the latest moment in time up to which process p can idle without executing an initial action. It is defined inductively as follows:

$$\begin{array}{lcl} U(\alpha[r]) & = & \max\{r, 0\} \\ U(p+q) & = & \max\{U(p), U(q)\} \\ U(p \cdot q) & = & U(p) \\ U(\langle X|E\rangle) & = & U(\langle t|E\rangle). \end{array}$$

In the last line it is assumed that E contains the equation $X = t$. The ultimate delay enables to distinguish processes that only differ in their deadlock behaviour, such as $a[1]+\delta[1]$ and $a[1]+\delta[2]$. The following notion of timed strong bisimulation [2] is a timed version of strong bisimulation [23]. The definition that is presented here stems from [18].

Definition 2.3 *Two process expressions p_0, q_0 are* strongly bisimilar*, notation $p_0 \leftrightarrows q_0$, if there exists a symmetric, binary* timed strong bisimulation relation *$\mathcal{B}$ on processes such that*

1. *$p_0 \mathcal{B} q_0$.*
2. *If $p \xrightarrow{a[r]} p'$ and $p\mathcal{B}q$, then $q \xrightarrow{a[r]} q'$ for some process q' with $p'\mathcal{B}q'$.*
3. *If $p \xrightarrow{a[r]} \surd$ and $p\mathcal{B}q$, then $q \xrightarrow{a[r]} \surd$.*
4. *If $p\mathcal{B}q$, then $U(p) = U(q)$.*

Process terms are considered modulo timed strong bisimulation equivalence.

2.4 Axiom System

Timed strong bisimulation is a congruence, which means that if $p \leftrightarrows p'$ and $q \leftrightarrows q'$, then $p+q \leftrightarrows p'+q'$ and $p \cdot q \leftrightarrows p' \cdot q'$. This property follows from the *path* format of Baeten and Verhoef [4]. They proved that if a collection of transition rules is within this format, and if these rules are 'well-founded', then the strong bisimulation equivalence it induces on the algebra of closed terms is always a congruence. (Fokkink and van Glabbeek [11] showed that the well-foundedness

requirement can be omitted.) In order to apply the path format, the inductive rules that define the predicates $U(p) = r$ for $r \in \mathbb{Q}_{>0} \cup \{\infty\}$ are to be incorporated as transition rules. Then the notion of timed strong bisimulation in Definition 2.3 agrees with the general notion of strong bisimulation in the presence of predicates, as defined in [4]. Thus, the transition rules in Table 1 and 2 meet the restrictions of the path format, so that the timed strong bisimulation relation they induce is a congruence.

Table 3 contains an axiom system for $\mathrm{BPA}_{\delta r}$ with recursion. Axioms A1-5 are the standard axioms from BPA, axioms TA6-9 are taken from [18], and axioms R1,2 for recursion stem from [6]. In these last two axioms, E denotes a linear recursive specification of the form $\{X_i = t_i \mid i = 1, ..., n\}$. The axiom R1 induces equalities such as $\langle X|X = a[r]\cdot X\rangle = a[r]\cdot\langle X|X = a[r]\cdot X\rangle$. The axiom R2 (or the *Recursive Specification Principle*) implies that each linear recursive specification has only one solution.

A1				$x + y$	$=$	$y + x$	
A2				$(x + y) + z$	$=$	$x + (y + z)$	
A3				$x + x$	$=$	x	
A4				$(x + y) \cdot z$	$=$	$x \cdot z + y \cdot z$	
A5				$(x \cdot y) \cdot z$	$=$	$x \cdot (y \cdot z)$	
TA6	$r \leq s$	$\Longrightarrow$	$\alpha[s] + \delta[r]$	$=$	$\alpha[s]$		
TA7			$\delta[r] \cdot x$	$=$	$\delta[r]$		
TA8	$r \leq 0$	$\Longrightarrow$	$\alpha[r]$	$=$	$\delta[0]$		
TA9			$\alpha[\infty]$	$=$	$\delta[\infty]$		
R1	$y_i = \langle X_i\|E\rangle$	$i = 1, ..., n$	$\Longrightarrow$	$y_1 = t_1[y_1/X_1, ..., y_n/X_n]$			
R2	$y_i = t_i[y_1/X_1, ..., y_n/X_n]$	$i = 1, ..., n$	$\Longrightarrow$	$y_1 = \langle X_1\|E\rangle$			

Table 3: Axioms for timed regular processes

The axiomatization for $\mathrm{BPA}_{\delta r}$ with recursion is sound with respect to timed strong bisimulation, i.e, if two process terms can be equated by the axioms, then they are timed strongly bisimilar. Since timed strong bisimulation is a congruence relation, this can be verified by checking soundness of the separate axioms. It is not hard to see that the axioms A1-5 and TA6-9 and R1 are sound with respect to timed strong bisimulation. A detailed proof of the soundness of R2 can be found in [9].

2.5 Completeness

This section is devoted to proving that the axiomatization for $\mathrm{BPA}_{\delta r}$ with recursion is complete with respect to timed strong bisimulation, i.e., if two process

terms are timed strongly bisimilar, then they can be equated by the axioms. First, each linear specification will be reduced to a normal form, by means of the axioms. Next, it will be shown that if two normal forms are bisimilar, then they are syntactically equal modulo α-conversion (i.e., modulo renaming of variables). This proves completeness.

Let $E = \{X_i = t_i \mid i = 1, ..., n\}$ be a linear specification. The process term $\langle X_1|E\rangle$ is reduced to normal form in several steps.

Step 1: Removal of redundant deadlocks

- First, replace each summand of terms t_i of the form $\alpha[r]$ with $r \leq 0$ by $\delta[0]$, and each summand of t_i of the form $a[\infty]$ by $\delta[\infty]$.

- Next, replace each summand of terms t_i of the form $\delta[r] \cdot X$ by $\delta[r]$.

- Finally, remove each summand $\delta[r]$ from terms t_i for which there is summand $a[s] \cdot X$ or $\alpha[s]$ in this t_i with $r \leq s$.

Step 2: Identification of bisimilar variables

If $\langle X_j|E\rangle \leftrightarrows \langle X_k|E\rangle$ with $j < k$, then rename X_k into X_j in all the terms t_i.

Step 3: Removal of double edges

If an expression $a[r]$ or $a[r] \cdot X_j$ occurs in a term t_i more than once, then remove all but one of the occurrences of this expression in t_i.

Step 4: Removal of redundant variables

Let the collection $dep(X_1)$ of variables in E that occur in the 'dependency graph' of X_1 be defined as follows:

$$X_1 \in dep(X_1),$$
$$X_i \in dep(X_1) \text{ and } X_j \text{ occurs in a } t_i \implies X_j \in dep(X_1).$$

If $X_j \notin dep(X_1)$, then remove the equation $X_j = t_j$ from E.

Thus the construction of the normal form of $\langle X_1|E\rangle$ has been finished. Step 1 is provable from R1,2 and TA6-9, Step 3 from R1,2 plus A3, and Step 4 from R1,2. We show that Step 2 can be proved from R1,2+A3. Let $\tilde{E}$ be the specification that results after identifying bisimilar variables in E. Let $X_{i(j)}$ denote the bisimilar variable that has been substituted for X_j in $\tilde{E}$, for $j = 1, ..., n$.

Lemma 2.4 R1,2+A3 $\vdash \langle X_1|E\rangle = \langle X_1|\tilde{E}\rangle$.

Proof. Let T_j denote the process $t_j[\langle X_{i(1)}|\tilde{E}\rangle/X_1, ..., \langle X_{i(n)}|\tilde{E}\rangle/X_n]$. It is easy to see that $\langle X_j|E\rangle \leftrightarrows T_j$ for $j = 1, ..., n$. Since $\langle X_j|E\rangle \leftrightarrows \langle X_{i(j)}|E\rangle$, this implies $T_j \leftrightarrows T_{i(j)}$ for $j = 1, ..., n$.

So if T_j has a subterm $a[r] \cdot \langle X_k|\tilde{E}\rangle$, then $T_{i(j)}$ has a subterm $a[r] \cdot \langle X_l|\tilde{E}\rangle$ where $\langle X_k|\tilde{E}\rangle$ and $\langle X_l|\tilde{E}\rangle$ are bisimilar. Since variables have been identified in $\tilde{E}$, it follows that $k = l$. By the same argument, each subterm $a[r] \cdot \langle X_k|\tilde{E}\rangle$ of $T_{i(j)}$ is also a subterm of T_j. Similarly, T_j has a subterm $a[r]$ if and only if $T_{i(j)}$ has a subterm $a[r]$. Finally, since $U(T_j) = U(T_{i(j)})$, Step 1 in the reduction to normal form ensures that T_j has a subterm $\delta[r]$ if and only if $T_{i(j)}$ has a subterm $\delta[r]$. Thus A3 $\vdash T_j = T_{i(j)}$.

Then $\langle X_{i(j)}|\tilde{E}\rangle \stackrel{\text{R1}}{=} T_{i(j)} \stackrel{\text{A3}}{=} T_j$. This holds for all j, so $\langle X_{i(1)}|\tilde{E}\rangle, ..., \langle X_{i(n)}|\tilde{E}\rangle$ is a solution for E. Then R2 implies $\langle X_j|E\rangle = \langle X_{i(j)}|\tilde{E}\rangle$ for $j = 1, ..., n$. □

The next proposition implies that A1-5+TA6-9+R1,2 constitutes a complete axiomatization for regular processes modulo timed strong bisimulation.

Proposition 2.5 *If two normal forms $\langle X_1|E\rangle$ and $\langle Y_1|E'\rangle$ are bisimilar, then they are syntactically equal modulo α-conversion.*

Proof. Let

$$\begin{aligned} E &= \{X_i = t_i \mid i = 1, ..., m\}, \\ E' &= \{Y_j = u_j \mid j = 1, ..., n\}. \end{aligned}$$

A relation ϕ between variables X_i and Y_j is defined as follows:

- $X_1 \phi Y_1$,
- for $i, j > 1$, $X_i \phi Y_j$ holds iff $\langle X_i|E\rangle \leftrightarrows \langle Y_j|E'\rangle$.

We derive several properties for the relation ϕ.

- For each X_i there exists a unique Y_j such that $X_i \phi Y_j$ holds, and vice versa.

  By Step 4, X_i is in the dependency graph of X_1. In other words, $\langle X_i|E\rangle$ is a derivative of $\langle X_1|E\rangle$. Then bisimilarity of $\langle X_1|E\rangle$ and $\langle Y_1|E'\rangle$ yields that there is a derivative of $\langle Y_1|E'\rangle$ that is bisimilar with $\langle X_i|E\rangle$. Since E' is linear, this derivative is of the form $\langle Y_j|E'\rangle$. So by definition of ϕ we have $X_i \phi Y_j$.

  Suppose that $X_i \phi Y_j$ and $X_i \phi Y_k$. Then $\langle Y_j|E'\rangle \leftrightarrows \langle X_i|E\rangle \leftrightarrows \langle Y_k|E'\rangle$, so Step 2, identification of bisimilar variables, ensures that $j = k$.

  By the symmetric argument it follows that for each Y_j there exists a unique X_i such that $X_i \phi Y_j$ holds. So we can conclude that ϕ constitutes a bijection between the variables in E and the variables in E'.

- If $\phi(X_i) = Y_j$, then $\phi(t_i) = u_j$, and vice versa.

Suppose that

$$t_i = \sum_k a_k[r_k] \cdot X_{i_k} + \sum_l \alpha_l[s_l].$$

with $0 < r_k < \infty$ and $0 \leq s_l \leq \infty$.

Since $\langle X_i|E\rangle \leftrightarrows \langle Y_j|E'\rangle$, u_j has a subterm $a_k[r_k]\cdot Y_{j_k}$ with $\langle X_{i_k}|E\rangle \leftrightarrows \langle Y_{j_k}|E'\rangle$ for each k. Moreover, since in Step 1 redundant deadlocks have been removed, and expressions $a[\infty]$ have been renamed into $\delta[\infty]$, u_j has a subterm $\alpha_l[s_l]$ for each l.

Since $\langle X_{i_k}|E\rangle \leftrightarrows \langle Y_{j_k}|E'\rangle$, the definition of ϕ yields $\phi(X_{i_k}) = Y_{j_k}$. In Step 3 double edges have been removed, so subterms $a_k[r_k] \cdot X_{i_k}$ and $a_k[r_k] \cdot Y_{j_k}$ and $\alpha_l[s_l]$ occur in t_i and u_j only once. Hence, $\phi(t_i) = u_j$ and $\phi(u_j) = t_i$.

So we conclude that ϕ constitutes an α-conversion between $\langle X_1|E\rangle$ and $\langle Y_1|E'\rangle$. □

Corollary 2.6 A1-5+TA6-9+R1,2 *form a complete axiomatization for $BPA_{\delta r}$ with recursion, modulo timed strong bisimulation.*

3 Abstraction

The previous section treated $\text{BPA}_{\delta r}$ with recursion modulo timed strong bisimulation. In this section the alphabet is extended with a special constant τ, to obtain $\text{BPA}_{\delta\tau r}$ with recursion, and process terms are considered modulo rooted timed branching bisimulation. In the sequel, a and α will represent elements from $A \cup \{\tau\}$ and $A \cup \{\delta, \tau\}$, respectively.

3.1 Time Shift

In order to define timed branching bisimulation, the syntax is extended with the *time shift* operator $(r)p$, which takes a rational number r and a process term p. The process term $(r)p$ denotes the behaviour of p that is shifted r units in time. Its ultimate delay is defined by

$$U((r)p) \;=\; \max\{U(p) + r, 0\}$$

The transition rules and axioms for the time shift are given in Table 4. Using axioms TS1-4, this operator can be eliminated from all process terms.

3.2 Timed Branching Bisimulation

The operational semantics consists of the transition rules in Table 1 and Table 2 and Table 4. The definition of timed strong bisimulation is adapted to that of timed *branching* bisimulation from Klusener [19]. In untimed branching bisimulation, a τ-transition is invisible if it does not lose possible behaviours, or in

$$\frac{x \xrightarrow{a[r]} \surd \quad r+s>0}{(s)x \xrightarrow{a[r+s]} \surd} \qquad \frac{x \xrightarrow{a[r]} x' \quad r+s>0}{(s)x \xrightarrow{a[r+s]} x'}$$

TS1	$s>0 \implies$	$(r)\alpha[s]$	$=$	$\alpha[r+s]$
TS2		$(r)\delta[0]$	$=$	$\delta[r]$
TS3		$(r)(x+y)$	$=$	$(r)x+(r)y$
TS4		$(r)(x \cdot y)$	$=$	$(r)x \cdot y$

Table 4: Transition rules and axioms for the time shift

other words, $\tau p+q$ is equivalent to p if q is semantically included in p. The same intuition is used to define timed branching bisimulation. In the latter semantics, a transition $p \xrightarrow{\tau[r]} p'$ of a process p can be matched with the passing of time in a process q if $U(q) > r$ and $p' \underline{\leftrightarrow}_b (-r)q$. That is, under these conditions the $\tau[r]$-transition in p and idling beyond r in q result in equivalent behaviours. However, this same intuition gives rise to a mathematical interpretation that is quite different from the untimed case. This is shown by the following examples.

Example 3.1 *In the untimed setting,* $\tau(a+b)+a \underline{\leftrightarrow}_b a+b$. *However,*

$$\tau[1] \cdot (a[1]+b[1]) + a[2] \not\underline{\leftrightarrow}_b a[2]+b[2].$$

Not executing the τ *at 1 in the process on the left means a decision that the* a*, and not the* b*, will be executed at 2.*

Example 3.2 *In the untimed setting,* $\tau a+b \not\underline{\leftrightarrow}_b a+\tau b$. *However,*

$$\tau[1] \cdot a[1] + b[2] \underline{\leftrightarrow}_b a[2] + \tau[1] \cdot b[1].$$

In both processes it is decided at time 1 whether the a *or the* b *will be executed at 2.*

In the definition of timed branching bisimulation, an auxiliary definition will be needed. Suppose that two processes p and q are timed branching bisimilar, and that p can execute an a-action. Unlike timed strong bisimulation, it may not be the case that q can execute the same initial a-action. Possibly, q will first execute a number of τ-actions, which can all be matched with idling in p. Finally, the resulting state q' can execute the a-action. This will be denoted as "$q \Rightarrow q'$ equivalent with p".

Definition 3.3 *Let* $\mathcal{B}$ *be a binary relation on processes. For* p *a process and* $r \geq 0$*, the relation "*$q \Rightarrow_r q'$ $\mathcal{B}$-equivalent with p*" is defined inductively as follows.*

1. *If* $U(q) \geq r$, *and* $(-t)p\mathcal{B}(-t)q$ *for* $0 \leq t \leq r$, *then* $q \Rightarrow_r (-r)q$ $\mathcal{B}$*-equivalent with* p.

2. *If* $q \xrightarrow{\tau[s]} q'$, *and* $(-t)p\mathcal{B}(-t)q$ *for* $0 \leq t < s$, *and* $q' \Rightarrow_{r-s} q''$ $\mathcal{B}$*-equivalent with* $(-s)p$, *then* $q \Rightarrow_r q''$ $\mathcal{B}$*-equivalent with* p.

Definition 3.4 *Two process terms* p_0 *and* q_0 *are* timed branching bisimilar, *denoted by* $p_0 \underline{\leftrightarrow}_b q_0$, *if there exists a symmetric binary relation* $\mathcal{B}$ *on processes such that*

1. $p_0 \mathcal{B} q_0$.

2. *If* $p\mathcal{B}q$ *and* $p \xrightarrow{a[r]} p'$, *then there exists a* q' *and an* $s < r$, *such that* $q \Rightarrow_s q'$ $\mathcal{B}$*-equivalent with* p, *and*

   - *either there exists a* q'' *such that* $q' \xrightarrow{a[r-s]} q''$ *with* $p'\mathcal{B}q''$,
   - *or* $a = \tau$ *and* $U(p') > 0$ *and* $p'\mathcal{B}(s-r)q'$.

3. *If* $p\mathcal{B}q$ *and* $p \xrightarrow{a[r]} \surd$, *then there exists a* q' *and an* s, *such that* $q \Rightarrow_s q'$ $\mathcal{B}$*-equivalent with* p, *and* $q' \xrightarrow{a[r-s]} \surd$.

4. *If* $p\mathcal{B}q$ *and* $U(p) > r$, *then there exists a* q' *such that* $q \Rightarrow_r q'$ $\mathcal{B}$*-equivalent with* p.

Similar to the untimed case, timed branching bisimulation is not a congruence. For example, $a[2] \underline{\leftrightarrow}_b \tau[1] \cdot a[1]$, but $a[2] + b[2] \not\underline{\leftrightarrow}_b \tau[1] \cdot a[1] + b[2]$. A rootedness condition is needed.

Definition 3.5 *Two process terms* p *and* q *are* rooted timed branching bisimilar, *denoted by* $p \underline{\leftrightarrow}_{rb} q$, *if*

1. $p \xrightarrow{a[r]} p'$ *if and only if* $q \xrightarrow{a[r]} q'$ *with* $p' \underline{\leftrightarrow}_b q'$.

2. $p \xrightarrow{a[r]} \surd$ *if and only if* $q \xrightarrow{a[r]} \surd$.

Rooted timed branching bisimulation is a congruence.

3.3 One Axiom for Abstraction

Using the intuition for branching bisimulation, rooted timed branching bisimulation equivalence is expressed in one axiom TT, from [19]. Surprisingly, a complete axiomatization is obtained for $\mathrm{BPA}_{\delta\tau r}$ with recursion by adding only this axiom to the axiom system.

TT	$U(x) \leq r \ \wedge \ U(y) > 0$	$\Longrightarrow$	$\alpha[s] \cdot (x + \tau[r] \cdot y)$	$=$	$\alpha[s] \cdot (x + (r)y)$

We derive an equation, which will be needed later on, to exemplify the use of axiom TT.

Equation 3.6 $\langle X|X = \tau[r] \cdot X\rangle \;=\; \tau[r] \cdot \delta[\infty]$.

Proof. The case $r \leq 0$ is easy, because then both terms equal $\delta[0]$. We focus on the case $r > 0$.

$$\begin{array}{ll} & \tau[r] \cdot \tau[r] \cdot \delta[\infty] \\ \overset{\text{TA6}}{=} & \tau[r] \cdot (\delta[0] + \tau[r] \cdot \delta[\infty]) \\ \overset{\text{TT}}{=} & \tau[r] \cdot (\delta[0] + (r)\delta[\infty]) \\ \overset{\text{TS1}}{=} & \tau[r] \cdot (\delta[0] + \delta[\infty]) \\ \overset{\text{TA6}}{=} & \tau[r] \cdot \delta[\infty] \end{array}$$

So according to axiom R2, $\tau[r] \cdot \delta[\infty]$ equals $\langle X|X = \tau[r] \cdot X\rangle$. □

3.4 Completeness

Since the axioms A1-5 and TA6-9 and R1,2 are sound with respect to timed strong bisimulation, it follows that they are also sound with respect to rooted timed branching bisimulation, because this last equivalence relates more process terms. Furthermore, it is not hard to see that TT and TS1-4 are also sound. This section is devoted to proving that the axiomatization is complete. We will show that if two solutions of linear specifications are rooted timed branching bisimilar, then they can be made timed strongly bisimilar, by the introduction of extra τ-steps. Then completeness of the axioms for rooted timed branching bisimulation follows from the completeness of the axioms for timed strong bisimulation.

Proposition 3.7 *If two linear specifications are rooted timed branching bisimilar, then they are provably equal to two linear specifications that are timed strongly bisimilar.*

Proof. Let $\langle X_1|E\rangle \underline{\leftrightarrow}_{rb} \langle Y_1|E'\rangle$. If the collection Q of positive rationals that occur as a time stamp in E or E' is empty, then both $\langle X_1|E\rangle$ and $\langle Y_1|E'\rangle$ equal $\delta[0]$ or $\delta[\infty]$, so that they are timed strongly bisimilar. In that case we are done, so we may assume that Q is non-empty. Q is finite, and it contains only rational numbers, so then there is a greatest rational R_0 such that r/R_0 is a natural number for all $r \in Q$.

First, we apply root unwinding to the linear specifications E and E'. Let these specifications contain equations $X_1 = t_1$ and $Y_1 = u_1$, respectively. Add an equation $X_{\text{root}} = t_1$ to E, where X_{root} does not yet occur in E, to obtain $\bar{E}$. Even so, add an equation $Y_{\text{root}} = u_1$ to E', where Y_{root} does not yet occur in E', to obtain $\bar{E}'$. The term $\langle X_1|E\rangle$ is provably equal to $\langle X_{\text{root}}|\bar{E}\rangle$, and the term $\langle Y_1|E'\rangle$ is provably equal to $\langle Y_{\text{root}}|\bar{E}'\rangle$, by axioms R1,2.

Next, the equations in $\bar{E}$ and $\bar{E}'$, except for the root equations, are saturated with $\tau[R_0]$-steps. For example, consider the equation for a variable Z in E, with $Z \neq X_{\rm root}$:

$$Z = \sum_{i \in I} a_i[r_i] \cdot V_i + \sum_{j \in J} \alpha_j[s_j].$$

If the time stamps r_i and s_j are not all $\leq R_0$, then such an equation is adapted as follows.

- Suppose that at least one of the time stamps r_i or s_j is a rational number (so unequal to ∞) greater than R_0. Replace the equation for Z by the following two equations:

$$Z = \sum_{\{i \in I | r_i = R_0\}} a_i[R_0] \cdot V_i + \sum_{\{j \in J | s_j = R_0\}} \alpha_j[R_0] + \tau[R_0] \cdot W$$

$$W = \sum_{\{i \in I | r_i > R_0\}} a_i[r_i - R_0] \cdot V_i + \sum_{\{j \in J | s_j > R_0\}} \alpha_j[s_j - R_0]$$

  where the variable W does not yet occur in $\bar{E}$. Note that for each time stamp r in the equation for W, either $r = \infty$ or r/R_0 is a natural number, owing to our choice of R_0.

  This adaptation of the equation for Z can be derived using axiom TT.

- If none of the time stamps r_i or s_j is a rational number greater than R_0, and so at least one of them equals ∞, then replace the equation for Z by:

$$Z = \tau[R_0] \cdot Z.$$

  This adaptation of the equation for Z can be derived using Equation 3.6.

Repeat this procedure until none of the equations in $\bar{E}$ for variables unequal to $X_{\rm root}$, and none of the equations in $\bar{E}'$ for variables unequal to $Y_{\rm root}$, contain time stamps greater than R_0. This procedure terminates, because positive time stamps are getting smaller and smaller. The resulting specifications, which contain no positive time stamps other than R_0, are denoted by $\tilde{E}$ and $\tilde{E}'$. The equalities $\langle X_{\rm root}|E\rangle = \langle X_{\rm root}|\tilde{E}\rangle$ and $\langle Y_{\rm root}|E\rangle = \langle Y_{\rm root}|\tilde{E}'\rangle$ can be derived from the axioms.

Since $\langle X_{\rm root}|\bar{E}\rangle \mathrel{\underline{\leftrightarrow}}_{rb} \langle Y_{\rm root}|\bar{E}'\rangle$, it follows that $\langle X_{\rm root}|\tilde{E}\rangle \mathrel{\underline{\leftrightarrow}}_{rb} \langle Y_{\rm root}|\tilde{E}'\rangle$. The rooted timed branching bisimulation relation between $\langle X_{\rm root}|\tilde{E}\rangle$ and $\langle Y_{\rm root}|\tilde{E}'\rangle$ is a timed strong bisimulation relation. Namely, owing to the rootedness condition, initial transitions $\langle X_{\rm root}|\tilde{E}\rangle \stackrel{a[r]}{\longrightarrow} p'$ are matched with initial transitions $\langle Y_{\rm root}|\tilde{E}'\rangle \stackrel{a[r]}{\longrightarrow} q'$, and vice versa. Moreover, the construction of $\tilde{E}$ and $\tilde{E}'$ ensures that non-initial transitions in the transitions systems of $\langle X_{\rm root}|\tilde{E}\rangle$ and $\langle Y_{\rm root}|\tilde{E}'\rangle$ have labels of the form $a[R_0]$, so such transitions $p \stackrel{a[R_0]}{\longrightarrow} p'$ in the one transition system are matched with transitions $q \stackrel{a[R_0]}{\longrightarrow} q'$ in the other. Furthermore, the terms $\langle X_{\rm root}|\bar{E}\rangle$ and $Y_{\rm root}|\tilde{E}'\rangle$ have the same ultimate delay, and

if proper derivatives of these terms are related, then these derivatives have the same ultimate delay 0 or R_0. Hence, $\langle X_{\rm root}|\tilde{E}\rangle \leftrightarrow \langle Y_{\rm root}|\tilde{E'}\rangle$. $\Box$

Proposition 3.7, together with the completeness result for timed strong bisimulation, immediately yields the desired completeness result for rooted timed branching bisimulation.

Corollary 3.8 A1-5+TA6-9+TT+TS1-4+R1,2 *is a complete axiomatization for* $BPA_{\delta\tau r}$ *with recursion, modulo rooted timed branching bisimulation.*

Proof. Suppose that $\langle X_1|E\rangle \leftrightarrow_{rb} \langle Y_1|E'\rangle$. According to Proposition 3.7, we can derive $\langle X_1|E\rangle = \langle X_{\rm root}|\tilde{E}\rangle$ and $\langle Y_1|E'\rangle = \langle Y_{\rm root}|\tilde{E'}\rangle$ with $\langle X_{\rm root}|\tilde{E}\rangle \leftrightarrow \langle Y_{\rm root}|\tilde{E'}\rangle$. Then the completeness result for timed strong bisimulation, Corollary 2.6, yields $\langle X_{\rm root}|\tilde{E}\rangle = \langle Y_{\rm root}|\tilde{E'}\rangle$. Hence,

$$\langle X_1|E\rangle \;=\; \langle X_{\rm root}|\tilde{E}\rangle \;=\; \langle Y_{\rm root}|\tilde{E'}\rangle \;=\; \langle Y_1|E'\rangle. \qquad \Box$$

References

[1] L. Aceto and A. Jeffrey. A complete axiomatization of timed bisimulation for a class of timed regular behaviours. *Theoretical Computer Science*, 152(2):251–268, 1995.

[2] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3(2):142–188, 1991.

[3] J.C.M. Baeten and J.A. Bergstra. Discrete time process algebra with abstraction. In H. Reichel, editor, *10th Conference on Fundamentals of Computation Theory (FCT'95),* Dresden, *LNCS 965*, pages 1–15. Springer-Verlag, 1995.

[4] J.C.M. Baeten and C. Verhoef. A congruence theorem for structured operational semantics with predicates. In E. Best, editor, *Proceedings 4th Conference on Concurrency Theory (CONCUR'93),* Hildesheim, *LNCS 715*, pages 477–492. Springer-Verlag, 1993.

[5] J.A. Bergstra and J.W. Klop. The algebra of recursively defined processes and the algebra of regular processes. In J. Paredaens, editor, *Proceedings 11th International Colloquium on Automata, Languages and Programming (ICALP'84),* Antwerp, *LNCS 172*, pages 82–95. Springer-Verlag, 1984.

[6] J.A. Bergstra and J.W. Klop. A complete inference system for regular processes with silent moves. In F.R. Drake and J.K. Truss, editors, *Proceedings Logic Colloquium 1986,* Hull, pages 21–81. North-Holland, 1988.

[7] L. Chen. A model for real-time process algebras. In A.M. Borzyszkowski and S. Sokołowski, editors, *Proceedings 18th Symposium on Mathematical Foundations of Computer Science (MFCS'93),* Gdansk, *LNCS 711*, pages 372–381. Springer-Verlag, 1993.

[8] L. Chen. Axiomatising real-timed processes. In S. Brooks, M. Main, A. Melton, M. Mislove, and D. Schmidt, editors, *Proceedings 9th Conference on Mathematical Foundations of Programming Semantics (MFPS'93),* New Orleans, *LNCS 802*, pages 215–229. Springer-Verlag, 1993.

[9] W.J. Fokkink. Regular processes with rational time and silent steps. Report CS-R9231, CWI, Amsterdam, 1992. Available at http://www.cwi.nl/cwi/publications/reports/CS-1992.html.

[10] W.J. Fokkink. An elimination theorem for regular behaviours with integration. In E. Best, editor, *Proceedings 4th Conference on Concurrency Theory (CONCUR'93),* Hildesheim, *LNCS 715*, pages 432–446. Springer-Verlag, 1993.

[11] W.J. Fokkink and R.J. van Glabbeek. Ntyft/ntyxt rules reduce to ntree rules. *Information and Computation*, 126(1):1–10, 1996.

[12] W.J. Fokkink and A.S. Klusener. An effective axiomatization for real time ACP. *Information and Computation*, 122(2):286–299, 1995.

[13] J.F. Groote. The syntax and semantics of timed μCRL. Report SEN-R9709, CWI, Amsterdam, 1997. Available at http://www.cwi.nl/cwi/publications/reports/SEN-1997.html.

[14] R.J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics. *Journal of the ACM*, 43(3):555–600, 1996.

[15] C. Ho-Stuart, H.S.M. Zedan, and M. Fang. Congruent weak bisimulation with dense real-time. *Information Processing Letters*, 46(2):55–61, 1993.

[16] U. Holmer, K.G. Larsen, and Y. Wang. Deciding properties of regular real timed processes. In K.G. Larsen and A. Skou, editors, *Proceedings 3rd Workshop on Computer Aided Verification (CAV'91),* Aalborg, *LNCS 575*, pages 432–442. Springer-Verlag, 1991.

[17] A.S. Klusener. Abstraction in real time process algebra. In J.W. de Bakker, C. Huizing, W.P. de Roever, and G. Rozenberg, editors, *Proceedings REX Workshop "Real Time: Theory in Practice",* Mook, *LNCS 600*, pages 325–352. Springer-Verlag, 1991.

[18] A.S. Klusener. Completeness in real time process algebra. In J.C.M. Baeten and J.F. Groote, editors, *Proceedings 2nd Conference on Concurrency Theory (CONCUR'91),* Amsterdam, *LNCS 527*, pages 376–392. Springer-Verlag, 1991.

[19] A.S. Klusener. The silent step in time. In W.R. Cleaveland, editor, *Proceedings 3rd Conference on Concurrency Theory (CONCUR'92),* Stony Brook, *LNCS 630*, pages 421–435. Springer-Verlag, 1992.

[20] R. Milner. A complete inference system for a class of regular behaviours. *Journal of Computer and System Sciences*, 28(3):439–466, 1984.

[21] R. Milner. A complete axiomatisation for observational congruence of finite-state behaviours. *Information and Computation*, 81(2):227–247, 1989.

[22] F. Moller and C. Tofts. Behavioural abstraction in TCCS. In W. Kuich, editor, *Proceedings 19th International Colloquium on Automata, Languages and Programming (ICALP'92),* Vienna, *LNCS 623*, pages 559–570. Springer-Verlag, 1992.

[23] D.M.R. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Proceedings 5th GI Conference,* Karlsruhe, *LNCS 104*, pages 167–183. Springer-Verlag, 1981.

[24] J. Quemada, D. de Frutos, and A. Azcorra. TIC: A TImed Calculus. *Formal Aspects of Computing*, 5(3):224–252, 1993.

[25] G.M. Reed and A.W. Roscoe. A timed model for communicating sequential processes. *Theoretical Computer Science*, 58(1/3):249–261, 1988.

[26] Y. Wang. *A calculus of real time systems.* PhD thesis, Chalmers University of Technology, Göteborg, 1991.