1992

W. J. Fokkink

Regular processes with relative time and silent steps

Computer Science/Department of Software Technology Report CS-R9231 July

CWI is het Centrum voor Wiskunde en Informatica van de Stichting Mathematisch Centrum
CWI is the Centre for Mathematics and Computer Science of the Mathematical Centre Foundation

CWI is the research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a non-profit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands organization for scientific research (NWO).

Regular Processes with Relative Time and Silent Steps

Willem Jan Fokkink
CWI
P.O. Box 4079, 1009 AB Amsterdam,
The Netherlands

Abstract

In this paper recursion is added to real-time process algebra. An elimination theorem and a completeness result are proven for regular processes, where the time domain is restricted to the rational numbers. Finally, the alphabet is extended with the silent step τ and completeness is deduced for regular processes w.r.t. rooted branching bisimulation.

1991 Mathematics Subject Classification: 68Q50, 68Q60.
1987 CR Categories: D.3.1, F.3.1.
Key Words & Phrases: ACP, relative time, regular process, abstraction, completeness.
Note: The author is supported by the European Communities under RACE project no. 1046 (SPECS).

1 Introduction

Many extensions of process algebras with a notion of time have been given, e.g. [RR88], [MT90], [NS90]. This paper is based on the approach of Baeten and Bergstra in [BB91]. They have extended ACP with real time. Process terms are constructed from timed actions, consisting of a symbolic action together with a time stamp taken from $[0, \infty]$. This time stamp can be interpreted either absolutely or relatively.

In absolute time a timed action $a(r)$ executes action a at time r. There are identities in ACP_ρ (ACP with real time) that do not hold in untimed ACP. For example, the equality

$$a(2) \cdot (b(1) + c(3)) \;=\; a(2) \cdot c(3)$$

holds, since after executing a at time 2 the first alternative of the remaining subterm $b(1)+c(3)$ cannot be chosen anymore.

In relative time a timed action $a[r]$ executes action a exactly r time units after the previous action has been executed. So the process $a[2] \cdot b[1]$ executes a at time 2 and then b at time 3. In $\mathrm{ACP}_{r\rho}$ (ACP with relative real time) there are some new identities too, e.g.

$$a[2] \,\|\, b[1] \,\|\, c[2] \;=\; b[1] \cdot (a|c)[1]$$

At time 1 action b cannot wait any longer and has to be executed. The time stamps of a and c are shifted back one time unit, for since we work in relative time the execution of b at 1 would otherwise delay the executions of a and c by one time unit. At time 2 actions a and c are executed. To avoid a deadlock they are forced into a communication.

Report CS-R9231
ISSN 0169-118X
CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

A process is called *regular* if it has only a finite number of states. One could argue that regularity of a process implies finiteness if time is involved, because an infinite process can execute actions at an infinite number of moments in time. For example, in this view the following process would not be regular.

$$a(1) \cdot a(2) \cdot a(3) \cdot a(4) \cdot ...$$

However, in relative time this process can be represented as follows.

$$a[1] \cdot a[1] \cdot a[1] \cdot a[1] \cdot ...$$

After executing an action a, it takes exactly 1 time unit before the next action a is executed. This process is defined by the recursive specification $X = a[1] \cdot X$. We will consider this process to be regular. In general, a process is said to be regular if it has only a finite number of 'horizons', where a horizon is the state that results after executing an action, with the time set back to zero.

It turns out that the class of regular processes is not an algebra if the time domain consists of the positive real numbers. Therefore we will restrict the time domain to the positive rational numbers, in which case the regular processes do form an algebra. Furthermore, we will give a complete proof system for regular processes modulo strong bisimulation.

Klusener has studied abstraction in real-time process algebra in [Klu91a]. He has given a complete axiomatisation for closed terms w.r.t. *branching* bisimulation. In this paper we will add recursion to this timed process algebra with silent moves and give a complete axiomatisation for regular processes (cf [Mil84], [BK88]).

Acknowledgements. Jan Bergstra suggested the research of regular processes with real time. Jos Baeten and Steven Klusener are thanked for giving useful comments.

2 Relative Time

We have just argued that it is easier to reason about regular processes in relative than in absolute time. Thus we will focus on relative time. The following paragraphs contain the semantics and a proof system for $ACP_{r\rho}$.

2.1 The syntax

Assume an alphabet A of atomic actions. Let A_δ denote $A \cup \{\delta\}$, where δ is the special constant representing deadlock. A *timed* action is of the form $a[r]$ with $a \in A_\delta$ and $r \in [0, \infty]$. In the sequel $\delta[0]$ is abbreviated by δ. We also assume a communication function $| : A_\delta \times A_\delta \rightarrow A_\delta$ which is commutative and associative and has δ as zero element.

Process terms are constructed using the following operators:

- the alternative composition $+$ and the sequential composition $\cdot$
- the parallel operators $\|, \mathbin{⌊\!⌊}, |$
- the encapsulation operator ∂_H

- the (negative) time shift σ^r_-

This last operator is needed to axiomatise the left merge $\|\!\!\_$. The process $\sigma^r_-(p)$ denotes the process p that is shifted back r time units in time. For example, $\sigma^1_-(a[3])$ is equal to $a[2]$.

2.2 Ultimate delay

The *ultimate delay* $U(p)$ of a process p is the latest moment in time till which p can idle without executing an initial action. It is defined inductively as follows, where $a \in A_\delta$.

$$\begin{array}{lcll} U(a[r]) & = & r & \\ U(p+q) & = & max\{U(p), U(q)\} & \\ U(p \cdot q) & = & U(p) & \\ U(\sigma^r_-(p)) & = & U(p) \dot{-} r & \\ U(p\Box q) & = & min\{U(p), U(q)\} & \Box \in \{\|, \|\!\!\_, |\} \\ U(\partial_H(p)) & = & U(p) & \end{array}$$

2.3 Latest start time

The latest start time $L(p)$ of a process p is the latest moment in time at which p can execute an initial action. In order to define this notion formally, we first define inductively the collection $initact(p)$ of initial actions of a process p. Let $a \in A$.

$$\begin{array}{lcl} initact(\delta[r]) & = & \emptyset \\ initact(a[r]) & = & \begin{cases} \emptyset & \text{if } r \in \{0, \infty\} \\ \{a[r]\} & \text{otherwise} \end{cases} \\ initact(p+q) & = & initact(p) \cup initact(q) \\ initact(p \cdot q) & = & initact(p) \\ initact(\sigma^r_-(p)) & = & \{a[s-r] \mid a[s] \in initact(p) \wedge s > r\} \\ initact(p \|\!\!\_ q) & = & \{a[s] \in initact(p) \mid s < U(q)\} \\ initact(p|q) & = & \{c[r] \mid \exists a, b \;\; a[r] \in initact(p) \wedge b[r] \in initact(q) \wedge a|b = c \neq \delta\} \\ initact(p\|q) & = & initact(p \|\!\!\_ q + q \|\!\!\_ p + p|q) \\ initact(\partial_H(p)) & = & \{a[r] \in initact(p) \mid a \notin H\} \end{array}$$

Now we can define $L(p)$ as follows.

$$L(p) = max\{r \mid \exists a \;\; a[r] \in initact(p)\}$$

2.4 Operational semantics

Table 1 contains an operational semantics for ACP_{rp} taken from [Klu91b]. The definitions of $U(p)$ and $L(p)$ in the previous paragraphs enable to distinguish processes that only differ in their deadlock behaviour. A process can do a δ-transition if it can idle beyond the latest moment in time that it can execute an initial action:

$$U(p) > L(p) \implies p \xrightarrow{\delta[U(p)]} \surd$$

$$a, b \in A \quad r \in \langle 0, \infty \rangle$$

$$a \quad : \quad a[r] \xrightarrow{a[r]} \surd$$

$$\cdot \quad : \quad \frac{p \xrightarrow{a[r]} \surd}{p \cdot q \xrightarrow{a[r]} q} \qquad \frac{p \xrightarrow{a[r]} p'}{p \cdot q \xrightarrow{a[r]} p' \cdot q}$$

$$+ \quad : \quad \frac{p \xrightarrow{a[r]} \surd}{p + q \xrightarrow{a[r]} \surd \quad q + p \xrightarrow{a[r]} \surd} \qquad \frac{p \xrightarrow{a[r]} p'}{p + q \xrightarrow{a[r]} p' \quad q + p \xrightarrow{a[r]} p'}$$

$$\sigma^s_- \quad : \quad \frac{p \xrightarrow{a[r]} \surd \quad s < r}{\sigma^s_-(p) \xrightarrow{a[r-s]} \surd} \qquad \frac{p \xrightarrow{a[r]} p' \quad s < r}{\sigma^s_-(p) \xrightarrow{a[r-s]} p'}$$

$$\delta \quad : \quad \frac{U(p) > L(p)}{p \xrightarrow{\delta[U(p)]} \surd}$$

$$\|, \lfloor\!\lfloor \quad : \quad \frac{p \xrightarrow{a[r]} p' \quad r < U(q)}{p \| q \xrightarrow{a[r]} p' \| \sigma^r_-(q) \quad q \| p \xrightarrow{a[r]} \sigma^r_-(q) \| p' \quad p \lfloor\!\lfloor q \xrightarrow{a[r]} p' \| \sigma^r_-(q)}$$

$$\frac{p \xrightarrow{a[r]} \surd \quad r < U(q)}{p \| q \xrightarrow{a[r]} \sigma^r_-(q) \quad q \| p \xrightarrow{a[r]} \sigma^r_-(q) \quad p \lfloor\!\lfloor q \xrightarrow{a[r]} \sigma^r_-(q)}$$

If $a|b = c \neq \delta$, then

$$\|, | \quad : \quad \frac{p \xrightarrow{a[r]} p' \quad q \xrightarrow{b[r]} q'}{p \| q \xrightarrow{c[r]} p' \| q' \quad p | q \xrightarrow{c[r]} p' \| q'} \qquad \frac{p \xrightarrow{a[r]} \surd \quad q \xrightarrow{b[r]} \surd}{p \| q \xrightarrow{c[r]} \surd \quad p | q \xrightarrow{c[r]} \surd}$$

$$\frac{p \xrightarrow{a[r]} \surd \quad q \xrightarrow{b[r]} q'}{p \| q \xrightarrow{c[r]} q' \quad q \| p \xrightarrow{c[r]} q' \quad p | q \xrightarrow{c[r]} q' \quad q | p \xrightarrow{c[r]} q'}$$

$$\partial_H \quad : \quad \frac{p \xrightarrow{a[r]} \surd \quad a \notin H}{\partial_H(p) \xrightarrow{a[r]} \surd} \qquad \frac{p \xrightarrow{a[r]} p' \quad a \notin H}{\partial_H(p) \xrightarrow{a[r]} \partial_H(p')}$$

Table 1: Action rules for $\mathrm{ACP}_{r\rho}$

2.5 An axiom system

Table 2 contains an axiom system for $ACP_{r\rho}$. The axioms defining the left merge are such that the merge does not result in arbitrary interleavings. For this would result in equalities like $a[1] \parallel b[2] = a[1] \cdot b[1] + b[2] \cdot \delta$. The deadlock at the right-hand side is avoided by requiring that $a[r] \lfloor\!\lfloor q$ can only execute action $a[r]$ if $r < U(q)$.

3 Recursion

3.1 Recursive specifications

A *recursive specification* E is a set of equations $\{X = t_X \mid X \in V_E\}$, where V_E is a collection of variables and t_X a process term containing possible occurrences of variables in V_E.

A *solution* of a recursive specification E, in a certain model of $ACP_{r\rho}$, is a collection of processes $\{p_X \mid X \in V_E\}$ such that the equations $X = t_X$ become true (in the model) if the p_X are substituted for the X. The process p_X is called a *solution* for X (with respect to E).

For E a recursive specification and $X \in V_E$, the syntactic construct $\langle X|E\rangle$ denotes a solution of X. It can be regarded as some kind of variable, ranging over the collection of solutions of X. By abuse of notation $\langle X|E\rangle$ is often abbreviated by X. This construct is supplied with an operational semantics by adding the following rules to Table 1, where t_X is the right-hand side of the equation for X in V_E.

$$\frac{t_X \xrightarrow{a[r]} \surd}{X \xrightarrow{a[r]} \surd} \qquad \frac{t_X \xrightarrow{a[r]} p}{X \xrightarrow{a[r]} p}$$

With this operational semantics, the term $\langle X|E\rangle$ is in a sense the 'minimal' solution of X, i.e. if X can do an $a[r]$-transition, then each solution of X can do this $a[r]$-transition.

In this paper we will only consider *finite* recursive specifications, i.e. it is assumed that for each recursive specification E the collection V_E is finite.

3.2 Regular processes

Assume a model for $ACP_{r\rho}$. Following [BK84] we define for each process p a collection $Sub(p)$ in the model. The $Sub(p)$ are the smallest collections of processes such that the following statements are true. Let $a \in A$ and $r \in \langle 0, \infty\rangle$.

- $[\![p]\!] \in Sub(p)$.
- If $[\![a[r] \cdot p' + q]\!] \in Sub(p)$, then $[\![p']\!] \in Sub(p)$.

A process p is called *regular* if $Sub(p)$ is finite in the model.

3.3 The real-time term model

The set $\mathbb{P}_\rho$ of process expressions is constructed inductively as follows.

1. Each timed action is a process expression.

$a, b \in A_\delta$	$r, s \in [0, \infty]$	
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		$(x \cdot y) \cdot z = x \cdot (y \cdot z)$
RTA1a		$a[0] = \delta$
RTA1b		$a[\infty] = \delta[\infty]$
RTA2	$r \geq s$	$\delta[r] + \delta[s] = \delta[r]$
RTA3		$\delta[r] \cdot p = \delta[r]$
RTA4		$a[r] + \delta[r] = a[r]$
RTNS1		$\sigma^r_-(a[s]) = a[s \dot{-} r]$
RTNS2		$\sigma^r_-(x + y) = \sigma^r_-(x) + \sigma^r_-(y)$
RTNS3		$\sigma^r_-(x \cdot y) = \sigma^r_-(x) \cdot y$
RTC1	$r \neq s$	$a[r] \mid b[s] = \delta[min\{r, s\}]$
RTC2		$a[r] \mid b[r] = (a \mid b)[r]$
CM1		$x \| y = x \lfloor\!\lfloor y + y \lfloor\!\lfloor x + x \mid y$
RTCM2a	$r < U(q)$	$a[r] \lfloor\!\lfloor q = a[r] \cdot \sigma^r_-(q)$
RTCM2b	$r \geq U(q)$	$a[r] \lfloor\!\lfloor q = \delta[U(q)]$
RTCM3a	$r < U(q)$	$(a[r] \cdot p) \lfloor\!\lfloor q = a[r] \cdot (p \| \sigma^r_-(q))$
RTCM3b	$r \geq U(q)$	$(a[r] \cdot p) \lfloor\!\lfloor q = \delta[U(q)]$
CM4		$(x + y) \lfloor\!\lfloor z = x \lfloor\!\lfloor z + y \lfloor\!\lfloor z$
CM5		$(a[r] \cdot p) \mid b[s] = (a[r] \mid b[s]) \cdot p$
CM6		$a[r] \mid (b[s] \cdot q) = (a[r] \mid b[s]) \cdot q$
CM7		$(a[r] \cdot p) \mid (b[s] \cdot q) = (a[r] \mid b[s]) \cdot (p \| q)$
CM8		$(x + y) \mid z = x \mid z + y \mid z$
CM9		$x \mid (y + z) = x \mid y + x \mid z$
D1	$a \notin H$	$\partial_H(a[r]) = a[r]$
D2	$a \in H$	$\partial_H(a[r]) = \delta[r]$
D3		$\partial_H(p + q) = \partial_H(p) + \partial_H(q)$
D4		$\partial_H(p \cdot q) = \partial_H(p) \cdot \partial_H(q)$

Table 2: An axiom system for $ACP_{r\rho}$

2. If E is a finite recursive specification and $X \in V_E$, then $\langle X|E\rangle \in \mathbb{P}_\rho$.

3. If $p, q \in \mathbb{P}_\rho$, then $p\Box q \in \mathbb{P}_\rho$ for $\Box \in \{+, \cdot, \|, \lfloor\!\lfloor, |\}$.

4. If $p \in \mathbb{P}_\rho$ and $H \subseteq A$, then $\partial_H(p) \in \mathbb{P}_\rho$.

5. If $p \in \mathbb{P}_\rho$ and $r \in [0, \infty]$, then $\sigma^r_-(p) \in \mathbb{P}_\rho$.

We consider process expressions in $\mathbb{P}_\rho$ modulo *(strong) bisimulation.*

Definition 3.1 *Two process expressions* p_0, q_0 *are said to be* strongly bisimilar, *notation* $p_0 \leftrightarrow q_0$, *if there exists a symmetric binary* bisimulation relation $\mathcal{R}$ *on* $\mathbb{P}_\rho$ *such that*

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

2. *if* $p \xrightarrow{a[r]} p'$ *and* $p\mathcal{R}q$, *then* $q \xrightarrow{a[r]} q'$ *for some process* q' *with* $p'\mathcal{R}q'$.

3. *if* $p \xrightarrow{a[r]} \surd$ *and* $p\mathcal{R}q$, *then* $q \xrightarrow{a[r]} \surd$.

Strong bisimulation is a congruence on $\mathbb{P}_\rho$. It is easy to see that $\mathbb{P}_\rho/\leftrightarrow$ is a model for $ACP_{r\rho}$ (cf [Gla87]). We will refer to it as the *real-time term model.*

3.4 Linear recursive specifications

A recursive specification E is called *linear* if it is of the form

$$\{X = \sum_i a_i[r_i] \cdot X_i + \sum_j b_j[s_j] \mid X \in V_E\}$$

It is easy to see for each model that all its regular processes are solutions of (finite) linear specifications. However, the converse need not be true; there may be an equality $p = q$ in the model, where p is a solution of a linear specification and q is not regular.

One can prove that the following proposition does hold in the model $\mathbb{P}_\rho$ (cf [BW90]).

Proposition 3.2 *A process is regular iff it is a solution of a linear recursive specification.*

3.5 Two axioms dealing with recursion

We add the following two axioms to Table 2, concerning regular processes. Let E be a linear recursive specification of the form $\{X_i = T_i(X_1, ..., X_n) \mid i = 1, ..., n\}$.

R1	$p_i = \langle X_i\|E\rangle \quad i = 1, ..., n \implies p_1 = T_1(p_1, ..., p_n)$
R2	$p_i = T_i(p_1, ..., p_n) \quad i = 1, ..., n \implies p_1 = \langle X_1\|E\rangle$

The axiom R1 induces equalities like $\langle X|X = a[r] \cdot X\rangle = a[r] \cdot \langle X|X = a[r] \cdot X\rangle$. And R2 tells us that indeed $\langle X|E\rangle$ is equal to each solution of X.

4 An Algebra of Regular Processes

Unfortunately the collection of regular processes is not a subalgebra of the real-time term model. The following counterexample is due to Jan Bergstra.

Example 4.1 *Consider the processes $X = a[1] \cdot X$ and $Y = b[\sqrt{2}] \cdot Y$. Then $X \| Y$ consists of the following sequence of atomic actions:*

$$a[1] \cdot b[\sqrt{2}-1] \cdot a[2-\sqrt{2}] \cdot b[2\sqrt{2}-2] \cdot a[3-2\sqrt{2}] \cdot a[1] \cdot b[3\sqrt{2}-4] \cdot a[5-3\sqrt{2}] \cdot ...$$

Thus for each pair $m, n \in \mathbb{N}$ with $m - n\sqrt{2} \in \langle 0, 1\rangle$ the timed action $a[m - n\sqrt{2}]$ will be executed by $X\|Y$. They form an infinite collection of timed actions. However, clearly a regular process can execute only finitely many different timed actions. So $X\|Y$ is not regular.

4.1 Rational time

If the time domain is restricted to the rational numbers, then it turns out that the regular processes do form an algebra. Still, even for simple processes their merge can become quite complicated. For example, consider the processes $X = a[\frac{1}{5}] \cdot X$ and $Y = b[\frac{1}{3}] \cdot Y$. Then

$$\begin{aligned} X\|Y = & \; a[\tfrac{1}{5}] \cdot (X \| \sigma_{-}^{\frac{1}{5}}(Y)) \; = \; a[\tfrac{1}{5}] \cdot b[\tfrac{1}{3} - \tfrac{1}{5}] \cdot (\sigma_{-}^{\frac{1}{3}-\frac{1}{5}}(X) \| Y) \\ = & \; a[\tfrac{1}{5}] \cdot b[\tfrac{1}{3} - \tfrac{1}{5}] \cdot a[\tfrac{2}{5} - \tfrac{1}{3}] \cdot (X \| \sigma_{-}^{\frac{2}{5}-\frac{1}{3}}(Y)) \; = \; ... \\ = & \; a[\tfrac{1}{5}] \cdot b[\tfrac{1}{3} - \tfrac{1}{5}] \cdot a[\tfrac{2}{5} - \tfrac{1}{3}] \cdot a[\tfrac{1}{5}] \cdot b[\tfrac{2}{3} - \tfrac{3}{5}] \cdot a[\tfrac{4}{5} - \tfrac{2}{3}] \cdot (a|b)[\tfrac{1}{5}] \cdot X \| Y \end{aligned}$$

In the sequel the time domain is restricted to the rational numbers, resulting in ACP_{rq} (ACP with relative rational time). Let $\mathbb{P}_q$ be the *rational-time* term model. The collection of regular processes in $\mathbb{P}_q$ will be denoted by $\mathbb{R}_q$.

Theorem 4.2 *$\mathbb{R}_q$ is a subalgebra of $\mathbb{P}_q$.*

Proof Let E and E' be linear recursive specifications.

$$E = \{X_i = \sum_k a_{ik}[r_{ik}] \cdot X_{i_k} + \sum_l a'_{il}[r'_{il}] \mid i = 1, ..., M\}$$

$$E' = \{Y_j = \sum_m b_{jm}[s_{jm}] \cdot Y_{j_m} + \sum_n b'_{jn}[s'_{jn}] \mid j = 1, ..., N\}$$

Then X_1 and Y_1 are regular processes. It is easy to see that $X_1 + Y_1$, $X_1 \cdot Y_1$ and $\sigma_{-}^{r}(X_1)$ are regular processes (i.e. are solutions of linear recursive specifications). We now prove that $X_1 \| Y_1$ is regular.

Let $\{r_1, ..., r_\alpha\}$ be the collection of time stamps in $\langle 0, \infty\rangle$ that occur in E and E' (so it consists of all the $r_{ik}, r'_{il}, s_{jm}, s'_{jn}$ that are unequal to 0 and ∞). We can assume that this collection is not empty, for else clearly X_1 and Y_1 are equal to either δ or $\delta[\infty]$ and we are done. Define a set Q of rational numbers in $[0, \infty\rangle$ as follows.

$$Q = \{k_1 \cdot r_1 + ... + k_\alpha \cdot r_\alpha \mid k_i \in \mathbb{Z}\} \cap [0, \infty\rangle$$

Since the r_i are rational numbers, it follows that there is a rational number $t_0 > 0$ such that

$$Q = \{0, t_0, 2t_0, 3t_0, 4t_0, ...\}$$

Let $R = max\{r_1, ..., r_\alpha\}$. Define $\mathcal{P}$ to be the following finite collection of processes. Define

$$\mathcal{P} = \{X_i,\ Y_j,\ \sigma_-^t(X_i),\ \sigma_-^t(Y_j),\ X_i \| Y_j,\ \sigma_-^t(X_i)\|Y_j,\ X_i\|\sigma_-^t(Y_j)\}$$

where t ranges over $t_0, 2t_0, ..., R$ and i over $1, ..., M$ and j over $1, ..., N$.

If $t \geq R$, then for each i either $\sigma_-^t(X_i) = \delta[\infty]$ (if some r_{ik} or r'_{il} is equal to ∞) or $\sigma_-^t(X_i) = \delta$ (otherwise). And similarly for processes $\sigma_-^t(Y_j)$. Thus each process of the form $\sigma_-^t(X_i)$, $\sigma_-^t(Y_j)$, $\sigma_-^t(X_i)\|Y_j$ or $X_i\|\sigma_-^t(Y_j)$ with $t \in Q$ is equal to a process in $\mathcal{P}$.

Now we show that for each $p \in \mathcal{P}$ there are processes $p_k \in \mathcal{P}$ such that

$$p = \sum_k c_k[t_k] \cdot p_k + \sum_l d_l[u_l]$$

(where the sums over k and l are finite). Then it follows that $X_1\|Y_1$ is regular, because by replacing the processes q in these equations by variables Z_q, we get a finite linear recursive specification $\tilde{E}$ such that

$$\text{R2} \vdash X_1\|Y_1 = \langle Z_{X_1\|Y_1} | \tilde{E} \rangle$$

For processes $\sigma_-^t(X_i)$ we have the following equality.

$$\sigma_-^t(X_i) = \sum_{\{k | r_{ik} > t\}} a_{ik}[r_{ik} - t] \cdot X_{i_k} + \sum_{\{l | r'_{il} > t\}} a'_{il}[r'_{il} - t]$$

There are similar equations for processes $\sigma_-^t(Y_j)$ and X_i and Y_j. Now consider the process $\sigma_-^t(X_i)\|Y_j$.

$$\begin{aligned}
\sigma_-^t(X_i) \mathbin{\|\!_} Y_j &= \sum_{\{k | 0 < r_{ik} - t < U(Y_j)\}} a_{ik}[r_{ik} - t] \cdot (X_{i_k} \| \sigma_-^{r_{ik}-t}(Y_j)) \\
&+ \sum_{\{l | 0 < r'_{il} - t < U(Y_j)\}} a'_{il}[r'_{il} - t] \cdot \sigma_-^{r'_{il}-t}(Y_j) \\
Y_j \mathbin{\|\!_} \sigma_-^t(X_i) &= \sum_{\{m | s_{jm} + t < U(X_i)\}} b_{jm}[s_{jm}] \cdot (\sigma_-^{s_{jm}+t}(X_i) \| Y_j) \\
&+ \sum_{\{n | s'_{jn} + t < U(X_i)\}} b'_{jn}[s'_{jn}] \cdot \sigma_-^{s'_{jn}+t}(X_i) \\
\sigma_-^t(X_i) | Y_j &= \sum_{\{(k,m) | r_{ik} - t = s_{jm}\}} (a_{ik} | b_{jm})[s_{jm}] \cdot (X_{i_k} \| Y_{j_m}) \\
&+ \sum_{\{(k,n) | r_{ik} - t = s'_{jn}\}} (a_{ik} | b'_{jn})[s'_{jn}] \cdot X_{i_k}
\end{aligned}$$

$$+ \sum_{\{(l,m)|r'_{il}-t=s_{jm}\}} (a'_{il}|b_{jm})[s_{jm}] \cdot Y_{jm}$$

$$+ \sum_{\{(l,n)|r'_{il}-t=s'_{jn}\}} (a'_{il}|b'_{jn})[s'_{jn}]$$

We can deduce similar equations for processes $X_i \| \sigma^t_-(Y_j)$ and $X_i \| Y_j$.

Thus it follows that $X_1 \| Y_1$ is regular. It is now easy to prove that $X_1 \llfloor Y_1$ and $X_1 | Y_1$ are regular. □

4.2 Soundness

Consider the model $\mathbb{R}_q$. It is easy to see that the axioms of ACP_{rq} and R1 are sound w.r.t. rooted branching bisimulation. We will now show that R2 is sound.

In order to do so we extend the syntax with a projection operator π_r, where $\pi_r(p)$ denotes the process p that is stopped at time r. Its action rules are

$$\frac{p \xrightarrow{a[s]} \surd \quad s < r}{\pi_r(p) \xrightarrow{a[s]} \surd} \qquad \frac{p \xrightarrow{a[s]} p' \quad s < r}{\pi_r(p) \xrightarrow{a[s]} \pi_{r-s}(p')}$$

Its ultimate delay and initial-actions set are defined by

$$\begin{aligned} U(\pi_r(p)) &= min\{U(p), r\} \\ initact(\pi_r(p)) &= \{a[s] \in initact(p) \mid s < r\} \end{aligned}$$

The projection operator is axiomatised by

RTPR1a	$s < r$	$\pi_r(a[s]) = a[s]$
RTPR1b	$s < r$	$\pi_r(a[s] \cdot p) = a[s] \cdot \pi_{r-s}(p)$
RTPR2a	$s \geq r$	$\pi_r(a[s]) = \delta[r]$
RTPR2b	$s \geq r$	$\pi_r(a[s] \cdot p) = \delta[r]$
RTPR3		$\pi_r(x + y) = \pi_r(x) + \pi_r(y)$

It is straightforward to check the soundness of these axioms w.r.t. $\underline{\leftrightarrow}$.

Proposition 4.3 R2 *is sound w.r.t. strong bisimulation.*

Proof. Let $E = \{X_i = T_i(X_1, ..., X_n) \mid i = 1, ..., n\}$ be a linear specification and let $p_1, ..., p_n$ be processes such that $p_i = T_i(p_1, ..., p_n)$ for $i = 1, ..., n$. We need to prove that $p_i \underline{\leftrightarrow} X_i$ for each i.

Since R1 is sound, it follows that $X_i \underline{\leftrightarrow} T_i(X_1, ..., X_n)$. And using an induction argument to the number of times that R2 is applied in the deductions $p_i = T_i(p_1, ..., p_n)$, we can assume that $p_i \underline{\leftrightarrow} T_i(p_1, ..., p_n)$.

If E does not contain time numbers from $\langle 0, \infty \rangle$, then for each i either

$$p_i \underline{\leftrightarrow} T_i(p_1, ..., p_n) \underline{\leftrightarrow} \delta \underline{\leftrightarrow} T_i(X_1, ..., X_n) \underline{\leftrightarrow} X_i$$

or

$$p_i \underline{\leftrightarrow} T_i(p_1, ..., p_n) \underline{\leftrightarrow} \delta[\infty] \underline{\leftrightarrow} T_i(X_1, ..., X_n) \underline{\leftrightarrow} X_i$$

and we are done. So we can assume that some r_0 is the smallest time number in $\langle 0, \infty\rangle$ occurring in E. We prove by induction to $[\frac{r}{r_0}]$ (which denotes the smallest whole number $\leq \frac{r}{r_0}$) that $\pi_r(p_i) \leftrightarrow \pi_r(X_i)$ for all $r \in \langle 0, \infty\rangle$.

First assume that $[\frac{r}{r_0}] = 0$, i.e. $0 < r < r_0$. Then

$$\pi_r(p_i) \leftrightarrow \pi_r(T_i(p_1, ..., p_n)) \overset{\text{RTPR1-3}}{=} \delta[r]$$

$$\pi_r(X_i) \leftrightarrow \pi_r(T_i(X_1, ..., X_n)) \overset{\text{RTPR1-3}}{=} \delta[r]$$

Since RTPR1-3 are sound, we can conclude that $\pi_r(p_i) \leftrightarrow \delta[r] \leftrightarrow \pi_r(X_i)$.

Now suppose that we have proven the case for $[\frac{r}{r_0}] \leq n$ and let $[\frac{r}{r_0}] = n + 1$. Assume that

$$T_i(X_1, ..., X_n) = \sum_j a_j[s_j] \cdot X_{i_j} + \sum_k b_k[t_k]$$

We can assume that there is an s_j or a t_k that is smaller than r, for otherwise $\pi_r(p_i) \leftrightarrow \delta[r] \leftrightarrow \pi_r(X_i)$ and we are done. Then

$$\pi_r(p_i) \leftrightarrow \pi_r(T_i(p_1, ..., p_n)) \overset{\text{RTPR1-3}}{=} \sum_{\{j|s_j<r\}} a_j[s_j] \cdot \pi_{r-s_j}(p_{i_j}) + \sum_{\{k|t_k<r\}} b_k[t_k]$$

$$\pi_r(X_i) \leftrightarrow \pi_r(T_i(X_1, ..., X_n)) \overset{\text{RTPR1-3}}{=} \sum_{\{j|s_j<r\}} a_j[s_j] \cdot \pi_{r-s_j}(X_{i_j}) + \sum_{\{k|t_k<r\}} b_k[t_k]$$

By induction we have $\pi_{r-s_j}(p_{i_j}) \leftrightarrow \pi_{r-s_j}(X_{i_j})$ for all j. Thus $\pi_r(p_i) \leftrightarrow \pi_r(X_i)$.

It is left to the reader to check that $\pi_r(p_i) \leftrightarrow \pi_r(X_i)$ for $r \in \langle 0, \infty\rangle$ induces $p_i \leftrightarrow X_i$. □

4.3 Completeness

We now prove that ACP$_{rq}$+R1,2 is a complete axiomatisation for $\mathbb{R}_q$. First each solution of a linear specification is reduced to a normal form, and it is shown that this reduction is provable in ACP$_{rq}$+R1,2. Finally, it is proven that if two solutions of linear specifications are bisimilar, then their normal forms are syntactically equal modulo α-conversion (i.e. modulo renaming of variables). Since each regular process is equal to a solution of a linear specification, this induces completeness.

Let $E = \{T_i(X_1, ..., X_n) \mid i = 1, ..., n\}$ be a linear specification. The process $\langle X|E\rangle$ is brought to normal form by reducing E. This reduction is described in several steps.

Step 1: Removal of redundant deadlocks

- First replace each expression in $T_i(X_1, ..., X_n)$ of the form $a[0]$ by $\delta[0]$ and each expression of the form $a[\infty]$ by $\delta[\infty]$.

- Now replace each expression in $T_i(X_1, ..., X_n)$ of the form $\delta[r] \cdot X$ by $\delta[r]$.

- Finally, remove each expression $\delta[r]$ from $T_i(X_1, ..., X_n)$ for which there is an expression $a[s] \cdot X$ or $a[s]$ in $T_i(X_1, ..., X_n)$ with $r \leq s$.

Step 2: Identification of bisimilar variables

If $\langle X_j|E\rangle \leftrightarrow \langle X_k|E\rangle$ with $j \neq k$, then rename all occurrences of X_k in the $T_i(X_1,...,X_n)$ into X_j.

Step 3: Removal of double edges

If an expression $a[r]$ or $a[r]\cdot X$ occurs in $T_i(X_1,...,X_n)$ more than once, then remove all but one of the occurrences of this expression in $T_i(X_1,...,X_n)$.

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:

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

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

Thus we have constructed the normal form of $\langle X_1|E\rangle$. Clearly Step 1 is provable by R1,2 and RTA1-4, while Step 3 follows from R1,2 plus A3 and Step 4 from R2. We now show that Step 2 can be deduced from R1,2+A3.

Let $\tilde{E}$ be the specification that results after identifying all bisimilar variables in E. Let $X_{i(j)}$ denote the (bisimilar) variable that has been substituted for X_j in $\tilde{E}$ ($j = 1,...,n$).

Proposition 4.4 $\mathrm{R1,2+A3} \vdash \langle X_j|E\rangle = \langle X_{i(j)}|\tilde{E}\rangle \qquad j = 1,...,n$

Proof. Abbreviate the process $\tilde{T}_j(\langle X_{i(1)}|\tilde{E}\rangle,...,\langle X_{i(n)}|\tilde{E}\rangle)$ to $\tilde{T}_j$. Then

$$\langle X_j|E\rangle \leftrightarrow \langle X_j|\tilde{E}\rangle \leftrightarrow \tilde{T}_j \qquad j = 1,...,n$$

This together with $\langle X_j|E\rangle \leftrightarrow \langle X_{i(j)}|E\rangle$ implies $\tilde{T}_j \leftrightarrow \tilde{T}_{i(j)}$.

Fix a j. If $\tilde{T}_j$ has a subterm $a[r]$, then $\tilde{T}_j \xrightarrow{a[r]} \surd$ (because in Step 1 all redundant deadlocks have been removed), and so $\tilde{T}_{i(j)} \xrightarrow{a[r]} \surd$. It follows that $\tilde{T}_{i(j)}$ has a subterm $a[r]$.

Similarly, if $\tilde{T}_j$ has a subterm $a[r]\cdot\langle X_k|\tilde{E}\rangle$, then $\tilde{T}_{i(j)}$ must have a subterm $a[r]\cdot\langle X_l|\tilde{E}\rangle$ with $\langle X_k|\tilde{E}\rangle \leftrightarrow \langle X_l|\tilde{E}\rangle$. Since bisimilar variables have been identified in $\tilde{E}$, we have $k = l$.

By the same argument it follows that each subterm $a[r]$ or $a[r]\cdot\langle X_k|\tilde{E}\rangle$ of $\tilde{T}_{i(j)}$ is a subterm of $\tilde{T}_j$. Thus $\mathrm{A3} \vdash \tilde{T}_j = \tilde{T}_{i(j)}$.

Then $\langle X_{i(j)}|\tilde{E}\rangle \stackrel{\mathrm{R1}}{=} \tilde{T}_{i(j)} = \tilde{T}_j$. This holds for all j, so then $\langle X_{i(1)}|\tilde{E}\rangle,...,\langle X_{i(n)}|\tilde{E}\rangle$ is a solution for E. Now R2 implies $\langle X_{i(j)}|\tilde{E}\rangle = \langle X_j|\tilde{E}\rangle$ for $j = 1,...,n$. □

Proposition 4.5 *If two normal forms $\langle X_1|E\rangle$ and $\langle Y_1|E\rangle$ are bisimilar, then they are syntactically equivalent modulo α-conversion.*

Proof. Let

$$\begin{array}{rcl} E & = & \{X_i = T_i(X_1,...,X_m) \mid i = 1,...,m\} \\ E' & = & \{Y_j = S_j(Y_1,...,Y_n) \mid j = 1,...,n\} \end{array}$$

We now inductively construct a mapping ϕ from the variables of E to those of E' such that $\langle X_i|E\rangle \leftrightarrow \langle \phi(X_i)|E'\rangle$ for each i, and if $\phi(X_i) = Y_j$, then $\phi \circ T_i(X_1,...,X_m) \equiv S_j(Y_1,...,Y_n)$.

Put $\phi(X_1) = Y_1$. Now suppose that we have already defined $\phi(X_i) = Y_j$ for some i. Let

$$T_i(X_1,...,X_m) = \sum_k a_k[r_k] \cdot X_{i_k} + \sum_l b_l[s_l].$$

Since $\langle X_i|E\rangle \leftrightarrow \langle Y_j|E'\rangle$, clearly $S_j(Y_1,...,Y_n)$ has a subterm $b_l[s_l]$ for each l and a subterm $a_k[r_k] \cdot Y_{j_k}$ with $\langle X_{i_k}|E\rangle \leftrightarrow \langle Y_{j_k}|E'\rangle$ for each k.

If $\phi(X_{i_k})$ has already been defined, then by induction $\langle \phi(X_{i_k})|E'\rangle \leftrightarrow \langle X_{i_k}|E\rangle \leftrightarrow \langle Y_{j_k}|E'\rangle$. Since bisimilar variables have been identified in Step 2, it follows that $\phi(X_{i_k}) \equiv Y_{j_k}$. If $\phi(X_{i_k})$ has not yet been defined, then put $\phi(X_{i_k}) = Y_{j_k}$.

In Step 3 double edges have been removed, so subterms $b_l[s_l]$ and $a_k[r_k] \cdot X_{i_k}$ resp. $a_k[r_k] \cdot Y_{j_k}$ occur in $T_i(X_1,...,X_m)$ resp. $S_j(Y_1,...,Y_n)$ only once. Thus $\phi \circ T_i(X_1,...,X_m) \equiv S_j(Y_1,...,Y_n)$.

By Step 4 each variable is in the dependency graph of X_1, so ϕ is defined for all variables in E.

It is easy to see that ϕ is bijective, since by a symmetric construction one can define its inverse. □

Thus we have proven that ACP$_{rq}$+R1,2 is a complete axiomatisation for $\mathbb{R}_q$. Now a 'timed' Approximation Induction Principle for regular processes can be deduced.

Corollary 4.6 *If* ACP$_{rq}$+R1+RTPR1-3 $\vdash \pi_r(p) = \pi_r(q)$ *for all* $r \in \langle 0, \infty\rangle$, *then* ACP$_{rq}$+ R1, 2 $\vdash p = q$.

Proof. ACP$_{rq}$+R1+RTPR1-3 is sound, so if $\pi_r(p) = \pi_r(q)$ is provable in this axiom system, then $\pi_r(p) \leftrightarrow \pi_r(q)$ for each $r \in \langle 0, \infty\rangle$. This implies $p \leftrightarrow q$. Now completeness gives ACP$_{rq}$ + R1, 2 $\vdash p = q$. □

5 Abstraction

We now add the silent step $\tau[r]$ to the syntax. Abstraction together with real time was studied in [Klu91a]. The operational semantics and proof theory that we will define in this chapter have been taken from that paper; they are based on the (untimed) *branching* bisimulation that was introduced in [GW89].

In the previous section the model $\mathbb{P}_q$ modulo strong bisimulation has been studied. In the following paragraphs we will consider the model $\mathbb{P}_{q\tau}/\leftrightarrow_{rb}$; the syntax is extended with the special constant τ, the *positive* time shift σ^r_+ and the *abstraction operator* τ_I, while processes are considered modulo *rooted branching* bisimulation. As before one can deduce that the regular processes of this model are exactly the solutions of linear specifications and that they form an algebra. We will refer to this algebra as $\mathbb{R}_{q\tau}$.

5.1 Branching bisimulation

We add the silent step τ as a special constant to the alphabet A. The operational semantics still consists of the action rules from Table 1, where $a, b \in A_\tau$. Only, the definition of strong bisimulation is adapted to that of *branching* bisimulation.

Definition 5.1 *Two processes p_0, q_0 are called* branching bisimilar, *notation $p_0 \leftrightarrow_b q_0$, if there exists a symmetric binary* branching bisimulation relation $\mathcal{R}$ *between processes such that*

1. $p_0 \mathcal{R} q_0$.
2. *if $p \xrightarrow{a[r]} p'$ and $p\mathcal{R}q$, then either*

   $a = \tau$ *and* $U(q) > r$ *and* $p'\mathcal{R}\sigma^r_-(q)$

   or $\exists q' \; \exists s < r$ *such that* $q \xrightarrow{\tau[s]} q'$ *and* $\sigma^s_-(p)\mathcal{R}q'$

   or $\exists q'$ *such that* $q \xrightarrow{a[r]} q'$ *and* $p'\mathcal{R}q'$
3. *if $p \xrightarrow{a[r]} \surd$ and $p\mathcal{R}q$, then either*

   $\exists q' \; \exists s < r$ *such that* $q \xrightarrow{\tau[s]} q'$ *and* $\sigma^s_-(p)\mathcal{R}q'$

   or $q \xrightarrow{a[r]} \surd$

The intuition behind this definition is that a timed τ-transition can be omitted if it does not lose any possible behaviours. For example:

$$a[2] + \tau[1] \cdot a[1] \leftrightarrow_b a[2]$$

because it does not make any difference if the τ is executed at 1 or not; in both cases the a will be executed at 2, after which the process terminates successfully. But

$$a[2] + \tau[1] \cdot b[1] \not\leftrightarrow_b a[2] + b[2]$$

because at time 1 it is decided whether the a or the b will be executed at 2.

The requirement in point 1 of the definition that $U(q) > r$ avoids equivalences like $a[r] + \tau[s] \cdot \delta \leftrightarrow_b a[r]$ for $r \leq s$.

5.2 Zeno processes

A process is called *Zeno* if it can execute an infinite number of actions in a finite amount of time.

Branching bisimulation is not an equivalence relation if Zeno processes are involved. For example, consider a recursive specification of the form

$$E = \{X_i = \tau[2^{-i}] \cdot X_{i+1} \mid i = 1, 2, 3, \ldots\}$$

The process $\langle X_1|E\rangle$ executes infinitely many τ's before 1. One can verify that $\langle X_1|E\rangle \leftrightarrow_b a[1]$ for each $a \in A_{\delta\tau}$. But clearly $a[1] \not\leftrightarrow_b b[1]$ if $a \neq b$.

However, since $\mathbb{P}_{q\mathcal{T}}$ contains only finite recursive specifications, it is easy to see that this model does not contain Zeno processes. It can be proven that branching bisimulation does constitute an equivalence relation on $\mathbb{P}_{q\mathcal{T}}$.

5.3 Rooted branching bisimulation

Unfortunately branching bisimulation is not a congruence. For example:

$$c[2] + a[2] + \tau[1] \cdot a[1] \not\leftrightarrow_b c[2] + a[2]$$

We need a rootedness condition.

Definition 5.2 *Two processes are called* rooted branching bisimilar, *notation* $p \leftrightarrow_{rb} q$, *if the following requirements hold.*

1. *if* $p \xrightarrow{a[r]} p'$, *then there is a* q' *such that* $q \xrightarrow{a[r]} q'$ *and* $p' \leftrightarrow_b q'$
2. *if* $q \xrightarrow{a[r]} q'$, *then there is a* p' *such that* $p \xrightarrow{a[r]} p'$ *and* $p' \leftrightarrow_b q'$
3. *if* $p \xrightarrow{a[r]} \surd$, *then* $q \xrightarrow{a[r]} \surd$
4. *if* $q \xrightarrow{a[r]} \surd$, *then* $p \xrightarrow{a[r]} \surd$

It can be proven that rooted branching bisimulation is a congruence on $\mathbb{P}_{q\mathcal{T}}$ (cf [Klu91a]).

5.4 The communication function for τ

In untimed process algebra the communication function is defined such that $\tau | a = \delta$ for all $a \in A_{\delta\mathcal{T}}$. However, here this would result in equivalences like

$$a[1] \cdot b[1] \cdot c[1] \leftrightarrow_{rb} (a[1] \cdot c[2]) \parallel b[2] \leftrightarrow_{rb} (a[1] \cdot \tau[1] \cdot c[1]) \parallel b[2] \leftrightarrow_{rb} a[1] \cdot \delta[1]$$

This example shows that in process algebra with real time the communication function for the silent step is to be defined by $\tau | a = a$ for all $a \in A_{\delta\mathcal{T}}$.

5.5 The abstraction operator

As usual we add the abstraction operator τ_I to the syntax, where $I \subseteq A$. Its action rules are

$\dfrac{p \xrightarrow{a[r]} \surd \quad a \notin I}{\tau_I(p) \xrightarrow{a[r]} \surd}$	$\dfrac{p \xrightarrow{a[r]} p' \quad a \notin I}{\tau_I(p) \xrightarrow{a[r]} \tau_I(p')}$
$\dfrac{p \xrightarrow{a[r]} \surd \quad a \in I}{\tau_I(p) \xrightarrow{\tau[r]} \surd}$	$\dfrac{p \xrightarrow{a[r]} p' \quad a \in I}{\tau_I(p) \xrightarrow{\tau[r]} \tau_I(p')}$

We define

$$\begin{aligned} U(\tau_I(p)) &= U(p) \\ initact(\tau_I(p)) &= \{a[r] \in initact(p) \mid a \notin I\} \cup \{\tau[r] \mid \exists a \in I \; a[r] \in initact(p)\} \end{aligned}$$

The abstraction operator is axiomatised by

TI1	$a \notin I$	$\tau_I(a[r]) = a[r]$
TI2	$a \in I$	$\tau_I(a[r]) = \tau[r]$
TI3		$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$
TI4		$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$

It can be proven that each process term is equal to a term that does not contain any occurrences of the abstraction operator. For example, we have

$$\text{TI1-4+R1,2} \vdash \tau_I(\langle X|E\rangle) = \langle X|\tau_I(E)\rangle$$

where $\tau_I(E)$ denotes the collection of equations that results from replacing all occurrences of atoms $a \in I$ in E by τ.

5.6 The positive time shift

The *positive* time shift σ_+^r is defined in order to give a complete axiomatisation for rooted branching bisimulation. The expression $\sigma_+^r(p)$ denotes the process p that is shifted forward r time units in time. Its action rules are

$$\frac{p \xrightarrow{a[s]} \surd}{\sigma_+^r(p) \xrightarrow{a[r+s]} \surd} \qquad \frac{p \xrightarrow{a[s]} p'}{\sigma_+^r(p) \xrightarrow{a[r+s]} p'}$$

We define

$$\begin{aligned} U(\sigma_+^r(p)) &= U(p) + r \\ initact(\sigma_+^r(p)) &= \{a[r+s] \mid a[s] \in initact(p)\} \end{aligned}$$

The positive time shift is axiomatised by

RTPS1	$\sigma_+^r(a[s])$	$=$	$a[r+s]$
RTPS2	$\sigma_+^r(x+y)$	$=$	$\sigma_+^r(x) + \sigma_+^r(y)$
RTPS3	$\sigma_+^r(x \cdot y)$	$=$	$\sigma_+^r(x) \cdot y$

5.7 An axiom for closed terms

Using the intuition for branching bisimulation that we have given in Paragraph 5.1, we can express rooted branching bisimulation equivalence in only one axiom.

	$r, s \in \langle 0, \infty \rangle$	
RTT	$U(p) \leq s \wedge U(q) > 0$	$a[r] \cdot (p + \tau[s] \cdot q) = a[r] \cdot (p + \sigma_+^s(q))$

The requirement $U(p) \leq s$ ensures that executing $\tau[s]$ does not lose possible behaviours, while $U(q) > 0$ avoids the equality $a[r] \cdot (p + \tau[s] \cdot \delta) = a[r] \cdot (p + \delta[s])$ for $U(p) \leq s$. This last equality has to be avoided, since it would give rise to equations like

$$a[1] \cdot b[1] \cdot c[1] = (a[1] \cdot (c[2] + \tau[1] \cdot \delta)) \parallel b[2] = a[1] \cdot (b[1] \cdot c[1] + b[1] \cdot \delta)$$

It has been proven in [Klu91a] that the axiom system of ACP_{rq} extended with RTPS1-3 and RTT is a complete axiomatisation for closed terms modulo rooted branching bisimulation.

5.8 Two axioms for recursion

In order to get a complete axiomatisation for regular processes w.r.t. rooted branching bisimulation, two axioms have to be added to R1,2. The axioms are very similar to axioms for regular processes with silent steps in the untimed case, that have been introduced in [BK88].

Assume a linear specification $E = \{X_i = T_i(X_1, ..., X_n) \mid i = 1, ..., n\}$. Let $r > 0$ and $i \neq 1$. The specification E_{-i} denotes E without the equation for X_i.

In axiom R3 it is assumed that the time stamps occurring in $T_i'(X_1, ..., X_n)$ are all $\leq r$, and that the time stamp 0 does not occur in $T_j(X_1, ..., X_n)$. The polynomial $T_j^r(X_1, ...X_n)$ denotes $T_j(X_1, ..., X_n)$ with the time stamps increased by r (i.e. each timed action $a[s]$ is replaced by $a[s+r]$).

R3	$\langle X_1 \mid E_{-i},\ X_i = T_i'(X_1, ..., X_n) + \tau[r] \cdot X_j \rangle \ =$
	$\langle X_1 \mid E_{-i},\ X_i = T_i'(X_1, ..., X_n) + T_j^r(X_1, ..., X_n) \rangle$
R4	$\langle X_1 \mid E_{-i},\ X_i = \tau[r] \cdot X_i \rangle \ = \ \langle X_1 \mid E_{-i},\ X_i = \delta[\infty] \rangle$

Note that in the model $\mathbb{R}_{q\tau}$ the axiom RTT is induced by R3.

5.9 Soundness

Consider the model $\mathbb{R}_{q\tau}$. It is easy to see that the axioms of ACP_{rq} and TI1-4 and R1,3,4 are sound w.r.t. rooted branching bisimulation. In order to prove the soundness of R2, we can extend the syntax with the projection operator π_r and repeat the proof from Paragraph 4.2. The following property of rooted branching bisimulation is essential in this proof.

$$\forall r \in \langle 0, \infty \rangle \ \ \pi_r(p) \leftrightarrow_{rb} \pi_r(q) \quad \Longrightarrow \quad p \leftrightarrow_{rb} q$$

It is left to the reader to check the validity of this statement.

The definition of branching bisimulation that we use here differs slightly from the one given in [Klu91a]; in that definition it is the case that if $p \leftrightarrow_b q$ and p can execute an action a with $a \neq \tau$, then after a certain number of τ-transitions q will be able to execute a too. This bisimulation induces a rooted branching bisimulation $\leftrightarrow_{rb}^*$ that does not have the property just mentioned, which is shown by the following example.

Example 5.3 *Consider the processes* $\tau[r] \cdot \delta[\infty]$ *and* $X = \tau[r] \cdot X$. *Unlike* $\tau[r] \cdot \delta[\infty]$, *the process* X *will never execute a* δ*-transition, so* $\tau[r] \cdot \delta[\infty] \not\leftrightarrow_b^* X$. *However, we have*

$$\begin{array}{lcccll} \pi_s(\tau[r] \cdot \delta[\infty]) & \leftrightarrow_{rb}^* & \delta[s] & \leftrightarrow_{rb}^* & \pi_s(X) & \textit{for } s \in \langle 0, r] \\ \pi_s(\tau[r] \cdot \delta[\infty]) & \leftrightarrow_{rb}^* & \tau[r] \cdot \delta[s-r] & \leftrightarrow_{rb}^* & \pi_s(X) & \textit{for } s \in \langle r, \infty \rangle \end{array}$$

5.10 Completeness

As in Paragraph 4.3, we prove completeness by reducing each solution of a linear specification to a normal form and showing that if two normal forms are bisimilar, then they are syntactically equivalent.

Let $E = \{X_i = T_i(X_1, ..., X_n) \mid i = 1, ..., n\}$ be a linear specification. We reduce $\langle X_1|E\rangle$ to normal form in several steps.

Step 1: Removal of redundant deadlocks

Replace expressions of the form $a[0]$ in $T_i(X_1, ..., X_n)$ by $\delta[0]$ and expressions of the form $a[\infty]$ by $\delta[\infty]$. Then replace expressions of the form $\delta[r] \cdot X$ by $\delta[r]$. Finally, remove expressions $\delta[r]$ from $T_i(X_1, ..., X_n)$ for which there is an expression $a[s] \cdot X$ or $a[s]$ in $T_i(X_1, ..., X_n)$ with $r \leq s$.

Step 2: Root unwinding

Add an equation $X_{\text{root}} = T_1(X_1, ..., X_n)$ to E, where X_{root} does not yet occur in E.

Step 3: Adding τ-steps

Let the equation of a variable $X_i \not\equiv X_{\text{root}}$ in E be given by

$$X_i = \sum_j a_j[r_j] \cdot X_{i_j} + \sum_k b_k[s_k]$$

Let t_0 be the smallest time number that occurs in this equation. If there is an r_j or s_k that is greater than t_0, then replace this equation in E by the following two equations:

$$X_i = \sum_{\{j|r_j=t_0\}} a_j[t_0] \cdot X_{i_j} + \sum_{\{k|s_k=t_0\}} b_k[t_0] + \tau[t_0] \cdot Y$$

$$Y = \sum_{\{j|r_j>t_0\}} a_j[r_j - t_0] \cdot X_{i_j} + \sum_{\{k|s_k>t_0\}} b_k[s_k - t_0]$$

where Y is a variable that does not yet occur in E.

Repeat this procedure until the equations in E for variables unequal to X_{root} have all become of the form

$$X_i = \sum_j a_j[r] \cdot X_{i_j} + \sum_k b_k[r]$$

Step 4: Identification of bisimilar variables

If $\langle X_j|E\rangle \leftrightarrow_b \langle X_k|E\rangle$ with $X_j \not\equiv X_k$ and $X_j, X_k \not\equiv X_{\text{root}}$, then rename all occurrences of X_k at the right-hand side of equations from E into X_j.

Step 5: Removal of double edges

If an expression $a[r]$ or $a[r] \cdot X$ occurs in a $T_i(X_1, ..., X_n)$ more than once, then remove all but one of the occurrences of this expression in $T_i(X_1, ..., X_n)$.

Step 6: Removal of τ-loops

Let X be a variable, with $X \not\equiv X_{\text{root}}$, for which the equation in E is of the form $X = \tau[r] \cdot X$. Then replace this equation in E by $X = \delta[\infty]$.

Step 7: Removal of redundant τ-steps

Suppose that there is an equation of the form $X_i = \tau[r] \cdot X_j$ in E with $X_i \not\equiv X_{root}$. Let the equation for X_j in E be as follows:

$$X_j = \sum_k a_k[s] \cdot X_{j_k} + \sum_l b_l[s]$$

Then replace the equation for X_i in E by

$$X_i = \sum_k a_k[s+r] \cdot X_{j_k} + \sum_l b_l[s+r]$$

The removal of τ-loops in the previous step ensures that this is a finite reduction.

Step 8: Removal of redundant variables

If $X_i \notin dep(X_{root})$, then remove the equation $X_i = T_i(X_1, ..., X_n)$ from E.

Thus we have constructed the normal form $\langle X_{root}|\tilde{E}\rangle$ of $\langle X_1|E\rangle$. Step 1 can be proven by R1,2+RTA1-4, Steps 2 and 8 by R2, Step 3 by R2,3, Step 5 by R1,2+A3, Step 6 by R4 and Step 7 by R3. We now show that Step 4 is provable.

Let Q be the collection of time stamps in $\langle 0, \infty\rangle$ that occur in E. We can assume that $Q \neq \emptyset$. Since Q is finite, and since it contains only rational numbers, there is a greatest common divisor t_0.

Reduce the specifications E and $\tilde{E}$ as follows. Consider an equation

$$X_i = \sum_j a_j[r] \cdot X_{i_j} + \sum_k b_k[r]$$

with $X_i \not\equiv X_{root}$. If $r = \infty$, then replace this equation by

$$X_i = \tau[t_0] \cdot X_i$$

And if $t_0 < r < \infty$, then replace it by the following two equations:

$$X_i = \tau[t_0] \cdot Y$$

$$Y = \sum_j a_j[r - t_0] \cdot X_{i_j} + \sum_k b_k[r - t_0]$$

where the variable Y does not yet occur in E or $\tilde{E}$.

Repeat this procedure until all equations in E and E' for variables unequal to X_{root} have become of the form

$$X_i = \sum_j a_j[t_0] \cdot X_{i_j} + \sum_k b_k[t_0]$$

The resulting specifications are denoted by E^* and $\tilde{E}^*$. By axioms R3,4 we have $\langle X_{root}|E\rangle = \langle X_{root}|E^*\rangle$ and $\langle X_{root}|\tilde{E}\rangle = \langle X_{root}|\tilde{E}^*\rangle$.

Since $\langle X_{root}|E\rangle \leftrightarrow_b \langle X_{root}|\tilde{E}\rangle$, it is clear that $\langle X_{root}|E^*\rangle \leftrightarrow_b \langle X_{root}|\tilde{E}^*\rangle$. By the construction of E^* and $\tilde{E}^*$, the branching bisimulation relation between $\langle X_{root}|E^*\rangle$ and $\langle X_{root}|\tilde{E}^*\rangle$ must be a strong bisimulation relation. Thus $\langle X_{\text{root}}|E^*\rangle \leftrightarrow \langle X_{\text{root}}|\tilde{E}^*\rangle$. Then the completeness result from the previous section gives $\langle X_{\text{root}}|E^*\rangle = \langle X_{\text{root}}|\tilde{E}^*\rangle$, and so finally

$$\langle X_{\text{root}}|E\rangle = \langle X_{\text{root}}|E^*\rangle = \langle X_{\text{root}}|\tilde{E}^*\rangle = \langle X_{\text{root}}|\tilde{E}\rangle$$

Similarly as in Proposition 4.5, it can be proven that if $\langle X_{\text{root}}|E\rangle$ and $\langle Y_{\text{root}}|E'\rangle$ are two normal forms that are rooted branching bisimilar, then they are syntactically equivalent modulo α-conversion.

Thus ACP_{rq}+TI1-4+R1-4 is a complete axiomatisation for regular processes modulo rooted branching bisimulation.

Similarly as in Corollary 4.6, one can now deduce an Approximation Induction Principle for regular processes with silent steps.

Corollary 5.4 *If* $\text{ACP}_{\text{rq}} + \text{RTT} + \text{R1} + \text{RTPR1-3} \vdash \pi_r(p) = \pi_r(q)$ *for all* $r \in \langle 0, \infty\rangle$*, then* $\text{ACP}_{rq} + \text{R1-4} \vdash p = q$.

References

[BB91] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Journal of Formal Aspects of Computing Science*, 3(2):142–188, 1991.

[BK84] J.A. Bergstra and J.W. Klop. The algebra of recursively defined processes and the algebra of regular processes. In J. Paredaens, editor, *Proceedings* 11^{th} *ICALP,* Antwerp, LNCS 172, pages 82–95. Springer-Verlag, 1984.

[BK88] 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.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process algebra.* Cambridge Tracts in Theoretical Computer Science 18. Cambridge University Press, 1990.

[Gla87] R.J. van Glabbeek. Bounded nondeterminism and the approximation induction principle in process algebra. In F.J. Brandenburg, G. Vidal-Naquet, and M. Wirsing, editors, *Proceedings STACS 87*, LNCS 247, pages 336–347. Springer-Verlag, 1987.

[GW89] R.J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics. In G.X. Ritter, editor, *Information Processing 89*, pages 613–618. North-Holland, 1989.

[Klu91a] 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 of the REX workshop "Real-Time: Theory in Practice",* Mook, LNCS 600. Springer-Verlag, 1991.

[Klu91b] A.S. Klusener. Completeness in real time process algebra. In J.C.M. Baeten and J.F. Groote, editors, *Proceedings CONCUR 91,* Amsterdam, LNCS 527, pages 376–392. Springer-Verlag, 1991.

[Mil84] R. Milner. A complete inference system for a class of regular behaviours. *Journal of Computer and System Sciences*, 28:439–466, 1984.

[MT90] F. Moller and C. Tofts. A temporal calculus of communicating systems. In J.C.M. Baeten and J.W. Klop, editors, *Proceedings CONCUR 90,* Amsterdam, LNCS 458, pages 401–415. Springer-Verlag, 1990.

[NS90] X. Nicollin and J. Sifakis. ATP: An algebra for timed processes. Technical Report RT-C26, IMAG, Laboratoire de Génie informatique, Grenoble, 1990.

[RR88] M. Reed and A.W. Roscoe. A timed model for communicating sequential processes. *Theoretical Computer Science*, 58:249–261, 1988.