Process Algebra with Timing:
Real Time and Discrete Time

by

J.C.M. Baeten and C.A. Middelburg
99/11

ISSN 0926-4515

All rights reserved
editors: prof.dr. J.C.M. Baeten
prof.dr. P.A.J. Hilbers

Reports are available at:
http://www.win.tue.nl/win/cs

Computing Science Reports 99/11
Eindhoven, September 1999
Process Algebra with Timing: Real Time and Discrete Time

J.C.M. Baeten\textsuperscript{1} and C.A. Middelburg\textsuperscript{1,2}

\textsuperscript{1} Computing Science Department, Eindhoven University of Technology
Eindhoven, the Netherlands
\textsuperscript{2} Department of Philosophy, Utrecht University
Utrecht, the Netherlands
\{josb,keesm\}@win.tue.nl

Abstract
We present real time and discrete time versions of ACP with absolute timing and relative timing. The starting-point is a new real time version with absolute timing, called ACP\textsubscript{sat}, featuring urgent actions and a delay operator. The discrete time versions are conservative extensions of the discrete time versions of ACP being known as ACP\textsubscript{dat} and ACP\textsubscript{drt}. The principal version is an extension of ACP\textsubscript{sat} with integration and initial abstraction to allow for choices over an interval of time and relative timing to be expressed. Its main virtue is that it generalizes ACP without timing and most other versions of ACP with timing in a smooth and natural way. This is shown for the real time version with relative timing and the discrete time version with absolute timing.

Keywords & Phrases: process algebra, ACP, real time, discrete time, absolute timing, relative timing, two-phase scheme, time-stamping scheme.


1 Introduction

Algebraic concurrency theories such as ACP [11, 9, 8], CCS [26, 27] and CSP [15, 23] have been extended to deal with time-dependent behaviour in various ways. First of all, timing is either absolute or relative and the time scale on which time is measured is either continuous or discrete. Besides, execution of actions and passage of time are either separated or combined. Separation corresponds to the two-phase scheme of modeling time-dependent behaviour and combination corresponds to the time-stamping scheme.

Absolute timing and relative timing have been studied in the framework of ACP for both a continuous time scale and a discrete time scale. See e.g. [1] and [6]. The versions of ACP with timing where time is measured on a continuous time scale are usually called real time versions. In the remainder of this paper, we adhere to this terminology. In the principal real time versions of ACP, viz. ACP\textsubscript{\rho} and ACP\textsubscript{\rho \sigma}, which were both introduced in [1], and ACP\textsubscript{\rho \sigma}, which was introduced in [3], execution of
actions and passage of time are combined. On the contrary, they are separated in the principal discrete time versions of ACP, viz. $\text{ACP}_{\text{dat}}$ and $\text{ACP}_{\text{drt}}$, which were both introduced in [6]. A real time version where execution of actions and passage of time are separated is $\text{ACP}_{\text{st}}$, which was introduced in [5], and [7] focuses on discrete time versions where they are combined.

Measuring time on a discrete time scale does not mean that the execution of actions is restricted to discrete points in time. In the discrete time versions of ACP, time is divided into time slices and timing of actions is done with respect to the time slices in which they are performed – within a time slice there is only the order in which actions are performed. Thus, the discrete time versions permit to consider systems at a more abstract level than the real time case, a level where time is measured with finite precision. This also occurs in practice: software components of a system are executed on processors where the measure of time is provided by a discrete clock and, in case a physical system is controlled, the state of the physical system is sampled and adjusted at discrete points in time. In any case, the abstraction made in the discrete time versions makes the time-dependent behaviour of programs better amenable to analysis.

ACP can simply be embedded in the discrete time versions $\text{ACP}_{\text{dat}}$ and $\text{ACP}_{\text{drt}}$ [6] by projecting the untimed process $a$ (for each action $a$) onto the delayable process $a$ – a delayable process $a$ is capable of performing the action $a$ in any time slice. Similarly, ACP can be embedded in the real time versions $\text{ACP}_p$ and $\text{ACP}_{p\text{r}}$ [1]. In other words, these discrete time and real time theories generalize the time free theory smoothly. Furthermore, in the discrete time case as well as the real time case, the relative time version can simply be embedded in the absolute time version extended with an initial abstraction operator to deal with relative timing.

However, the real time versions do not generalize the discrete time versions as smoothly as they generalize the time free theory. It turns out, as shown in [2], that the discrete time processes correspond to the real time processes for which the following holds: (1) if an action can be performed at some time $p \in \mathbb{R}$ such that $n < p < n + 1$ ($n \in \mathbb{N}$), it can also be performed at any other time $p' \in \mathbb{R}$ such that $n < p' < n + 1$; (2) no actions can be performed at times $p \in \mathbb{N}$. Clearly, such an embedding seriously lacks naturalness. The real time versions $\text{ACP}_p$ and $\text{ACP}_{p\text{r}}$ as well as the discrete time versions $\text{ACP}_{\text{dat}}$ and $\text{ACP}_{\text{drt}}$ are generalizations of ACP by intention. Since the real time versions were developed in advance of the discrete time versions, the former versions were not intentionally developed as generalizations of the latter versions. This explains at least partially the contrived embedding.

In this paper, we present a new real time version of ACP with absolute timing which originates from $\text{ACP}_{p\text{r}}$, a real time version introduced in [5]. In this version, which features urgent actions and a delay operator, execution of actions and passage of time are separated. We explain how execution of actions and passage of time can be combined in this version. We further add an integration operator, with which a choice over an interval of time can be expressed, and an initial abstraction operator, with which relative timing can be expressed, to this version. We show how a real time version of ACP with relative timing, which originates from $\text{ACP}_{\text{st}}$ [5], can be embedded in the extended real time version with absolute timing. We also present discrete time versions of ACP with absolute timing and relative timing which are conservative extensions of $\text{ACP}_{\text{dat}}$ and $\text{ACP}_{\text{drt}}$ [6]. We add an initial abstraction operator to the
discrete time version with absolute timing as well. Showing how the discrete time version with relative timing can be embedded in the extended discrete time version with absolute timing, can be done similarly to the real time case. We show that the extended real time version generalizes the extended discrete time version smoothly. In this case, the following holds for those real time processes that correspond to the discrete time processes: if an action can be performed at some time \( p \in \mathbb{R} \) such that \( n \leq p < n + 1 \) \((n \in \mathbb{N})\), it can also be performed at any other time \( p' \in \mathbb{R} \) such that \( n \leq p' < n + 1 \).

The main virtue of the extended real time version of ACP presented here is that it generalizes time free ACP as well as most other versions of ACP with timing in a smooth and natural way. The lack of a real time version of ACP with these characteristics was our main motivation to develop it. Different from the real time versions of [1] and [3], this version does not exclude the possibility of two or more actions to be performed consecutively at the same point in time. That is, it includes urgent actions, similar to ATP [30] and the different versions of CCS with timing [16, 28, 35]. This is useful in practice when describing and analyzing systems in which actions occur that are entirely independent. This is, for example, the case for actions that happen at different locations in a distributed system. In [1] and [3], the main idea was that it is difficult to imagine that actions are performed consecutively at the same point in time. But yet, this way of representing things is perfectly in line with modeling parallelism by interleaving. In point of fact it allows for independent actions to be handled faithfully.

In [1] and [3], ways to deal with independent actions are proposed where such actions take place at the same point in time by treating it as a special case of communication. This is, however, a real burden in the description and the analysis of the systems concerned. Of course, this does not limit the practical usefulness of ACP\(\rho\) and ACP\(\rho_{\mathcal{P}}\) for systems in which no independent actions occur. The real time versions ACP\(\mathcal{S}\rho\) and ACP\(\mathcal{S}\rho_{\mathcal{T}}\) of [5] simply do not exclude the possibility of two or more actions to be performed consecutively at the same point in time. Embedding in ACP\(\rho\) and ACP\(\rho_{\mathcal{P}}\), respectively, is obtained by extending the time domain to a domain that includes non-standard real numbers. We conjecture that the real time version presented in this paper, which originates from ACP\(\mathcal{S}\rho\), can be embedded in ACP\(\rho\) as well.

The structure of this paper is as follows. First of all, in Section 2, we present the new real time version of ACP with absolute timing. We also explain how execution of actions and passage of time can be combined in this version. Then, in Section 3, we add integration and initial abstraction to this real time version of ACP. Next, in Section 4, we first present a real time version of ACP with relative timing and then show that it can be embedded in the real time version of ACP with absolute timing presented in Sections 2 and 3. After that, in Section 5, we first present conservative extensions of the discrete time versions ACP\(\mathcal{D}\rho_{\mathcal{D}}\) and ACP\(\mathcal{D}\rho_{\mathcal{T}}\) of [6] and then show that the presented discrete time version with absolute timing can be embedded in the real time version with absolute timing presented in Sections 2 and 3. Finally, in Section 6, we make some concluding remarks.
2 Real time process algebra: absolute timing

In this section, we give the signature, axioms and term model of ACP<sup>sat</sup>, a standard real time process algebra with absolute timing. In this theory, the non-negative standard real numbers (\(\mathbb{R}_{\geq 0}\)) are used as the time domain. ACP<sup>sat</sup> originates from the theory ACP<sub>sp</sub>, presented in [5]. Unlike ACP<sub>sp</sub>, it separates execution of actions and passage of time.

In case of ACP<sup>sat</sup>, it is assumed that a theory of the non-negative real numbers has been given. Its signature has to include the constant 0 : \(\rightarrow \mathbb{R}_{\geq 0}\), the operator + : \(\mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}\), and the predicates \(\leq : \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}\) and = : \(\mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}\). In addition, this theory has to include axioms that characterize + as a commutative and associative operation with 0 as a neutral element and \(\leq\) as a total ordering that has 0 as its least element and that is preserved by +.

In ACP<sup>sat</sup>, as in the other versions of ACP with timing presented in this paper, it is assumed that a fixed but arbitrary set \(A\) of actions has been given. It is also assumed that a fixed but arbitrary communication function, i.e. a partial commutative and associative function \(\gamma : A \times A \rightarrow A\), has been given. The function \(\gamma\) is regarded to give the result of the synchronous execution of any two actions for which this is possible, and to be undefined otherwise. The weak restrictions on \(\gamma\) allow many kinds of communication between parallel processes to be modeled.

First, in Section 2.1, we treat BPA<sup>sat</sup>, basic standard real time process algebra with absolute timing, in which parallelism and communication are not considered. After that, in Section 2.2, BPA<sup>sat</sup> is extended to ACP<sup>sat</sup> to deal with parallelism and communication as well. Finally, we demonstrate in Section 2.3 how one can combine execution of actions and passage of time in ACP<sup>sat</sup>.

2.1 Basic process algebra

In BPA<sup>sat</sup>, we have the sort \(P\) of processes, the constants \(\tilde{a}\) (one for each \(a \in A\)), \(\tilde{\delta}\) and \(\tilde{\jmath}\), and the operators \(\sigma_{\text{abs}}\) (absolute delay), \(\cdot\) (sequential composition) and + (alternative composition). The constants \(\tilde{a}\) stand for \(a\) at time 0. Similarly, the constant \(\tilde{\delta}\) stands for a deadlock at time 0. The constant \(\tilde{\jmath}\) stands for an immediate deadlock, a process that exhibits inconsistent timing at time 0. This means that \(\tilde{\delta}\), different from \(\tilde{\delta}\), is not existing at time 0. The process \(\sigma_{\text{abs}}^p(x)\) is the process \(x\) shifted in time by \(p\). Thus, the process \(\sigma_{\text{abs}}^p(\tilde{a})\) is capable of first idling from time 0 to time \(p\) and then upon reaching time \(p\) performing action \(a\), immediately followed by successful termination. The process \(\sigma_{\text{abs}}^p(\tilde{\delta})\) is only capable of idling from time 0 to time \(p\). Time \(p\) can be reached by \(\sigma_{\text{abs}}^p(\tilde{\delta})\). This is the difference with the process \(\sigma_{\text{abs}}^p(\tilde{\delta})\), which can only idle up to, but not including, time \(p\). So \(\sigma_{\text{abs}}^p(\tilde{\delta})\) can not reach time \(p\). The process \(x \cdot y\) is the process \(x\) followed upon successful termination by the process \(y\). The process \(x + y\) is the process that proceeds with either the process \(x\) or the process \(y\), but not both. As in the untimed case, the choice is resolved upon execution of the first action, and not before. We also have the auxiliary operators \(\nu_{\text{abs}}\) (absolute time-out) and \(\nu_{\text{abs}}\) (absolute initialization). The process \(\nu_{\text{abs}}^p(x)\) is the part of \(x\) that starts to perform actions before time \(p\). The process \(\nu_{\text{abs}}^p(x)\) is the part of \(x\) that starts to perform actions at time \(p\) or later.

A real time version of ACP with absolute timing where the notation \(\tilde{a}\) was used
earlier for urgent actions is ACPS$^d$ [5], but there it always carries a time-stamp. The binary operator $\sigma_{ab}$ generalizes the unary operator $\sigma_{ab}$ of ACPS$^d$ [6] in a real time setting: for a real time process $x$ that corresponds to a discrete time process $x'$, $\sigma_{ab}(x)$ corresponds to $\sigma_{ab}(x')$. In earlier papers, including [1], [2], [3] and [5], the notations $x \gg p$ and $p \gg x$ were used instead of $\upsilon_{ab}(x)$ and $\overline{up}_{ab}(x)$, respectively. Besides, the time-out operator and the initialization operator were sometimes called the bounded initialization operator and the time shift operator, respectively.

It can be proved, using the axioms of BPA$^sat$, that each process expressed using the auxiliary operators $\upsilon_{ab}$ and $\overline{up}_{ab}$ is equal to a process expressed without them. In other words, in BPA$^sat$, all processes can be constructed from the constants using absolute delay, alternative composition and sequential composition only.

**Signature of BPA$^sat$**
The signature of BPA$^sat$ consists of the urgent action constants $\hat{a} : \rightarrow P$ (for each $a \in A$), the urgent deadlock constant $\hat{\delta} : \rightarrow P$, the immediate deadlock constant $\tilde{\delta} : \rightarrow P$, the alternative composition operator $+ : P \times P \rightarrow P$, the sequential composition operator $\cdot : P \times P \rightarrow P$, the absolute delay operator $\sigma_{ab} : \mathbb{R}_{\geq 0} \times P \rightarrow P$, the absolute time-out operator $\upsilon_{ab} : \mathbb{R}_{\geq 0} \times P \rightarrow P$, and the absolute initialization operator $\overline{up}_{ab} : \mathbb{R}_{\geq 0} \times P \rightarrow P$.

We assume that an infinite set of variables (of sort P) has been given. Given the signature of BPA$^sat$, terms of BPA$^sat$, also referred to as process expressions, are constructed in the usual way. We will generally use infix notation for binary operators. The need to use parentheses is further reduced by ranking the precedence of the binary operators. Throughout this paper we adhere to the following precedence rules: (i) the operator $\cdot$ has the highest precedence amongst the binary operators, (ii) the operator $+$ has the lowest precedence amongst the binary operators, and (iii) all other binary operators have the same precedence. We will also use the following abbreviation. Let $(t_i)_{i \in \mathcal{I}}$ be an indexed set of terms of BPA$^sat$ where $\mathcal{I} = \{i_1, \ldots, i_n\}$. Then we write \( \sum_{i \in \mathcal{I}} t_i \) for \( t_{i_1} + \ldots + t_{i_n} \). We further use the convention that \( \sum_{i \in \emptyset} t_i \) stands for \( \hat{\delta} \) if \( \mathcal{I} = \emptyset \).

We denote variables by \( x, x', y, y', \ldots \) An important convention is that we use \( a, a', b, b', \ldots \) to denote elements of \( A \cup \{\delta\} \) in the context of an equation, and elements of \( A \) in the context of an operational semantics rule. Furthermore, we use \( H \) to denote a subset of \( A \). We denote elements of \( \mathbb{R}_{\geq 0} \) by \( p, p', q, q' \) and elements of \( \mathbb{R}_{> 0} \) by \( r, r' \). We write \( A_\delta \) for \( A \cup \{\delta\} \).

**Axioms of BPA$^sat$**
The axiom system of BPA$^sat$ consists of the equations given in Tables 1 and 2.

<table>
<thead>
<tr>
<th>Equation</th>
<th>Axiom</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x + y = y + x$</td>
<td>A1</td>
</tr>
<tr>
<td>$(x + y) + z = x + (y + z)$</td>
<td>A2</td>
</tr>
<tr>
<td>$x + x = x$</td>
<td>A3</td>
</tr>
<tr>
<td>$(x + y) \cdot z = x \cdot z + y \cdot z$</td>
<td>A4</td>
</tr>
<tr>
<td>$(x \cdot y) \cdot z = x \cdot (y \cdot z)$</td>
<td>A5</td>
</tr>
<tr>
<td>$x + \hat{\delta} = x$</td>
<td>A6ID</td>
</tr>
<tr>
<td>$\hat{\delta} \cdot x = \hat{\delta}$</td>
<td>A7ID</td>
</tr>
</tbody>
</table>

Table 1: Axioms of BPA with immediate deadlock
\[ \sigma_{abs}(x) = u_{abs}^0(x) \]
\[ \sigma_{abs}(\pi_q (x)) = \sigma_{abs}^{p,q}(x) \]
\[ \sigma_{abs}(x) + \sigma_{abs}(y) = \sigma_{abs}(x + y) \]
\[ \sigma_{abs}^p(x) \cdot u_{abs}^p(y) = \sigma_{abs}^p(x \cdot y) \]
\[ \sigma_{abs}^p(x) \cdot (u_{abs}(y) + \sigma_{abs}(z)) = \sigma_{abs}^p(x \cdot u_{abs}(z)) \]
\[ \sigma_{abs}(\delta) \cdot r = \sigma_{abs}(\delta) \]
\[ \bar{\sigma} + \bar{\delta} = \bar{\sigma} \]
\[ \sigma_{abs}(x) + \bar{\delta} = \sigma_{abs}(x) \]
\[ \bar{\sigma} \cdot \bar{\delta} = \bar{\delta} \]
\[ v_{abs}^p(\delta) = \bar{\delta} \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}^p(0) = 0 \]
\[ v_{abs}(\delta) = \sigma_{abs}(\delta) \]
\[ v_{abs}(\delta) = \sigma_{abs}(\delta) \]
\[ v_{abs}(\delta) = \sigma_{abs}(\delta) \]
\[ v_{abs}^p(x + y) = v_{abs}^p(x) + v_{abs}^p(y) \]
\[ v_{abs}^p(x \cdot y) = v_{abs}^p(x) \cdot y \]
\[ v_{abs}^p(x) = v_{abs}^p(x) \]
\[ v_{abs}^p(x) = v_{abs}^p(x) \]
\[ v_{abs}^p(x) = v_{abs}^p(x) \]
\[ v_{abs}^p(x) = v_{abs}^p(x) \]
\[ v_{abs}^p(x) = v_{abs}^p(x) \]

Table 2: Additional axioms for BPA_{sat} (a \in A_\delta, p, q \geq 0, r > 0)

Axioms A1-A5 are common to ACP and all real and discrete time versions of ACP. Axioms A6ID and A7ID are simple reformulations of the axioms A6 and A7 of ACP: the constant \( \delta \) has been replaced by the constant \( \bar{\delta} \) – which is introduced because the intended interpretation of \( \delta \) in ACP_{sat} differs from \( \bar{\delta} \). These axioms or similar reformulations of A6 and A7 are found in all real and discrete time versions of ACP. Axiom SAT1, and a few axioms treated later, become easier to understand by realizing that in BPA_{sat}, as well as in ACP_{sat}, the equation \( v_{abs}^0(t) = t \) is derivable for all closed terms \( t \). This equation expresses that initialization at time 0 has no effect on processes with absolute timing. To accommodate for the extension with initial abstraction in Section 3.2, we have used \( v_{abs}^0(x) \) instead of \( x \) where the former is needed in the extension. Axioms SAT1 and SAT2 point out that a time shift by 0 has no effect in case of absolute timing and that consecutive time shifts add up. Axioms SAT3-SAT5 represent the interaction of absolute delay with alternative composition and sequential composition. Axiom SAT3, called the time factorization axiom, shows that passage of time by itself can not determine a choice. Axioms SAT4 and SAT5 express that if a process terminates successfully at some point in time, it can only be followed by the part of another process that starts to perform actions at the same time or later. Axiom SAT6 is a generalization of axiom A7ID. Using axioms A6SAa and A6SAb, the equation \( t + \bar{\delta} = t \) can be derived for all closed terms \( t \) unless \( t = \bar{\delta} \) – obviously \( \bar{\delta} + \bar{\delta} = \bar{\delta} \). Axiom A7SA is another simple reformulation of axiom A7 of ACP. Axioms SAT00-SAT05 and SA01-SA15 reflect the intended meaning of the
time-out and initialization operators clearly. Axioms SAT02 and SAI2 make precise what happens if a part that starts to perform actions before the time-out time and a part that starts to perform actions at the initialization time or later, respectively, do not exist. Equations SAT03' and SAI3' given in Table 3 are derivable from the axioms of BPA\textsuperscript{sat}. In BPA\textsuperscript{sat} and ACP\textsuperscript{sat}, and also in the further extension with initial abstraction, axiom SAT02 can be replaced by equation SAT03' just as well. In BPA\textsuperscript{sat} and ACP\textsuperscript{sat}, but not in the further extension with initial abstraction, axioms SAI0a, SAI1 and SAI3 together can be replaced by the equations SAI1'' and SAI3'' given in Table 3. The absolute initialization operator could have been added later with the addition of the absolute abstraction operator. However, having it available in BPA\textsuperscript{sat} and ACP\textsuperscript{sat} makes it possible to express interesting properties of real time processes with absolute timing such as the properties presented in Lemmas 1 and 3 below.

We can prove that the auxiliary operators $v_{abs}$ and $\overline{v}_{abs}$ can be eliminated in closed terms of BPA\textsuperscript{sat}. We can also prove that sequential compositions in which the form of the first operand is not $\overline{\alpha}$ ($\alpha \in \mathcal{A}$) and alternative compositions in which the form of the first operand is $\sigma_{abs}(t)$ can be eliminated in closed terms of BPA\textsuperscript{sat}. The terms that remain after exhaustive elimination are called the basic terms over BPA\textsuperscript{sat}. Because of this elimination result, we are permitted to use induction on the structure of basic terms over BPA\textsuperscript{sat} to prove statements for all closed terms of BPA\textsuperscript{sat}.

Examples We give some examples of a closed term of BPA\textsuperscript{sat} and the corresponding basic term:

$$
\begin{align*}
\sigma_{abs}^5(\alpha) \cdot \sigma_{abs}^4(\beta) &= \sigma_{abs}^5(\alpha \cdot \beta) \\
\sigma_{abs}^2(\alpha) \cdot (\sigma_{abs}^2(\beta) + \sigma_{abs}^4(\beta)) &= \sigma_{abs}^2(\alpha \cdot (\beta + \sigma_{abs}^4(\beta))) \\
v_{abs}(\sigma_{abs}^2(\alpha) + \sigma_{abs}^4(\beta)) &= \sigma_{abs}^2(\alpha + \sigma_{abs}^4(\beta)) \\
v_{abs}^5(\sigma_{abs}^2(\alpha) + \sigma_{abs}^4(\beta)) &= \sigma_{abs}^2(\alpha + \sigma_{abs}^4(\beta)) \\
\end{align*}
$$

The following lemmas are also useful in proofs. They are, for example, used in the proof of Theorem 12 (embedding of ACP\textsuperscript{sat} in ACP\textsuperscript{satI}). These lemmas, as most other lemmas in this paper, call for proofs by induction on the structure of basic terms. The proofs are generally straightforward, but long and tedious. For that reason, we will present for each such proof only one of the cases to be treated. The selected case is usually typical of the proof and relatively hard. We write $\equiv$ to indicate that the induction hypothesis of the proof is used.

Lemma 1 In BPA\textsuperscript{sat} and ACP\textsuperscript{sat}, as well as in the further extensions with restricted integration and initial abstraction introduced in Section 3:

1. the equation $t = v_{abs}(t) + \overline{v}_{abs}(t)$ is derivable for all closed terms $t$ such that $t = \overline{v}_{abs}(t)$ and $t = t + \sigma_{abs}^5(\delta)$;
2. the equations \( t = \nu^p_{abs}(t) \) and \( \overline{\nu}^p_{abs}(t) = \sigma^p_{abs}(\delta) \) are derivable for all closed terms \( t \) such that \( t = \overline{\nu}^0_{abs}(t) \) and \( t \neq t + \sigma^p_{abs}(\delta) \).

**Proof.** It is straightforward to prove both 1 and 2 by induction on the structure of \( t \).

1. We present only the case that \( t \) is of the form \( \sigma^q_{abs}(t') \). The other cases are similar, but simpler, and do not require case distinction.

Case \( p \leq q \):
\[
\sigma^q_{abs}(t') + \sigma^p_{abs}(\delta) \xrightarrow{\text{ Sat2}} \sigma^q_{abs}(\tau') + \delta \xrightarrow{\text{ Sat3}} \nu^p_{abs}(\sigma^q_{abs}(t')) + \overline{\nu}^p_{abs}(\sigma^q_{abs}(t'))
\]

Case \( p > q \):
\[
\sigma^q_{abs}(t') + \sigma^p_{abs}(\delta) \xrightarrow{\text{ Sat2}} \sigma^q_{abs}(\sigma^{q-p}(t')) + \sigma^p_{abs}(\sigma^{p-q}(\delta)) \xrightarrow{\text{ Sat3}} \sigma^q_{abs}(\tau' + \sigma^{p-q}(\delta)) \xrightarrow{\text{ Sat3}} \sigma^q_{abs}(\nu^p_{abs}(t')) + \overline{\nu}^p_{abs}(\sigma^q_{abs}(t'))
\]

In applying \( \text{ Sat3} \) we assume that \( t' = \overline{\nu}^0_{abs}(t') \). In case of \( \text{ BPA}^{\text{ sat}} \), \( \text{ ACP}^{\text{ sat}} \) and \( \text{ ACP}^{\text{ sat}} \) with integration, this equation is derivable for all closed terms \( t' \). The assumption is also justified in case of extension with initial abstraction. In that case, we are permitted, because of elimination results presented in Section 3.2, to consider here only closed terms of the form \( \sigma^q_{abs}(t') \) where no initial abstraction occurs in \( t' \).

2. Observe that \( \overline{\nu}^p_{abs}(t) = \sigma^p_{abs}(\delta) \) follows immediately from \( t = \nu^p_{abs}(t) \) by axiom \( \text{ SI3} \). So it suffices to prove only \( t = \nu^p_{abs}(t) \). Again, we present only the case that \( t \) is of the form \( \sigma^q_{abs}(t') \).

Case \( p \leq q \):
\[
\sigma^q_{abs}(t') \xrightarrow{\text{ Sat2}} \sigma^q_{abs}(\sigma^{q-p}(t')) + \delta \xrightarrow{\text{ Sat3}} \nu^p_{abs}(\sigma^q_{abs}(t')) + \sigma^p_{abs}(\sigma^{q-p}(t'))
\]

Case \( p > q \):
\[
\sigma^q_{abs}(t') \neq \sigma^q_{abs}(t') + \sigma^p_{abs}(\delta) \xrightarrow{\text{ Sat2, Sat3}} t' \neq t + \sigma^p_{abs}(\delta)
\]

By the induction hypothesis, \( \sigma^q_{abs}(t') = \sigma^q_{abs}(\nu^p_{abs}(t')) \xrightarrow{\text{ Sat3}} \nu^p_{abs}(\sigma^q_{abs}(t')) \)

From Lemma 1 we readily conclude the following.

**Corollary 2** In \( \text{ BPA}^{\text{ sat}} \) and \( \text{ ACP}^{\text{ sat}} \), as well as in the further extensions with restricted integration and initial abstraction introduced in Section 3, the equation \( \sigma^p_{abs}(t) \cdot \tau = \sigma^p_{abs}(t) \cdot \overline{\nu}^p_{abs}(t') \) is derivable for all closed terms \( t \) and \( t' \) such that \( t' = \overline{\nu}^0_{abs}(t') \).

**Lemma 3** In \( \text{ BPA}^{\text{ sat}} \) and \( \text{ ACP}^{\text{ sat}} \), as well as in the further extensions with restricted integration and initial abstraction introduced in Section 3, for each \( p \in \mathbb{R}_{>0} \) and each closed term \( t \), there exists a closed term \( t' \) such that \( \overline{\nu}^p_{abs}(t) = \sigma^p_{abs}(t') \) and \( t' = \overline{\nu}^0_{abs}(t') \).

In subsequent proofs, we write \( t_{[p]} \) for a fixed but arbitrary closed term \( t' \) that fulfills these conditions.

**Proof.** It is straightforward to prove this by induction on the structure of \( t \). We present only the case that \( t \) is of the form \( \sigma^q_{abs}(t'') \). Again, the other cases are similar, but simpler, and do not require case distinction.

Case \( p \leq q \):
\[
\nu^p_{abs}(\sigma^q_{abs}(t'')) \xrightarrow{\text{ Sat3}} \sigma^q_{abs}(t'') \xrightarrow{\text{ Sat2}} \sigma^q_{abs}(\sigma^{q-p}(t'')) \quad \text{and}
\]
\[
\sigma^q_{abs}(t'') \xrightarrow{\text{ Sat2}} \sigma^q_{abs}(\sigma^{q-p}(t'')) \xrightarrow{\text{ Sat1}} \overline{\nu}^0_{abs}(\sigma^{q-p}(t''))
\]

Case \( p > q \):
\[
\sigma^q_{abs}(t'') \xrightarrow{\text{ Sat2}} \sigma^q_{abs}(\sigma^{q-p}(t'')) \xrightarrow{\text{ Sat1}} \sigma^q_{abs}(\sigma^{q-p}(t'')) \xrightarrow{\text{ Sat2}} \sigma^p_{abs}(t''_{[p-q]}) \quad \text{and}
\]
\[
t''_{[p-q]} = \overline{\nu}^0_{abs}(t''_{[p-q]})
\]

In applying \( \text{ Sat3} \) we assume that \( t'' = \overline{\nu}^0_{abs}(t'') \). As described in the previous proof, this assumption is justified in all cases.
Lemma 1 indicates that a process that is able to reach time \( p \) can be regarded as being the alternative composition of the part that starts to perform actions before \( p \) and the part that starts to perform actions at \( p \) or later. Lemma 3 shows that the part of a process that starts to perform actions at time \( p \) or later can always be regarded as a process shifted in time by \( p \).

**Semantics of BPA\textsuperscript{sat}**

A real time transition system over \( A \) consists of a set of states \( S \), a root state \( p \in S \) and four kinds of relations on states:

- a binary relation \( \langle \cdot, p \rangle \xrightarrow{a} \langle \cdot, p \rangle \) for each \( a \in A \), \( p \in \mathbb{R}_{\geq 0} \),
- a unary relation \( \langle \cdot, p \rangle \xrightarrow{\sqrt{\cdot}} \langle \cdot, p \rangle \) for each \( a \in A \), \( p \in \mathbb{R}_{\geq 0} \),
- a binary relation \( \langle \cdot, p \rangle \xrightarrow{\cdot} \langle \cdot, q \rangle \) for each \( r \in \mathbb{R}_{> 0} \), \( p, q \in \mathbb{R}_{\geq 0} \) where \( q = p + r \),
- a unary relation \( \text{ID}(\cdot, p) \) for each \( p \in \mathbb{R}_{\geq 0} \);

satisfying

1. if \( \langle s, p \rangle \xrightarrow{r + r'} \langle s', q \rangle \), \( r, r' > 0 \), then there is a \( s'' \) such that \( \langle s, p \rangle \xrightarrow{r} \langle s'', p + r \rangle \) and \( \langle s'', p + r \rangle \xrightarrow{r'} \langle s', q \rangle \);
2. if \( \langle s, p \rangle \xrightarrow{r} \langle s'', p + r \rangle \) and \( \langle s'', p + r \rangle \xrightarrow{r'} \langle s', q \rangle \), then \( \langle s, p \rangle \xrightarrow{r + r'} \langle s', q \rangle \).

The four kinds of relations are called action step, action termination, time step and immediate deadlock relations, respectively. We write RTTS(\( A \)) for the set of all real time transition systems over \( A \).

We shall associate a transition system \( TS(t) \) in RTTS(\( A \)) with a closed term \( t \) of BPA\textsuperscript{sat} by taking the set of closed terms of BPA\textsuperscript{sat} as set of states, the closed term \( t \) as root state, and the action step, action termination, time step and immediate deadlock relations defined below using rules in the style of Plotkin [31]. A semantics given in this way is called a structured operational semantics. On the basis of these rules, the operators of BPA\textsuperscript{sat} can also be directly defined on the set of real time transition systems in a straightforward way. Note that the relations can in this case be explained as follows:

\[
\langle t, p \rangle \xrightarrow{a} \langle t', p \rangle : \text{process } t \text{ is capable of first performing action } a \text{ at time } p \text{ and then proceeding as process } t';
\]

\[
\langle t, p \rangle \xrightarrow{\sqrt{\cdot}} \langle \sqrt{\cdot}, p \rangle : \text{process } t \text{ is capable of first performing action } a \text{ at time } p \text{ and then terminating successfully;}
\]

\[
\langle t, p \rangle \xrightarrow{\cdot} \langle t', q \rangle : \text{process } t \text{ is capable of first idling from time } p \text{ to time } q \text{ and then proceeding as process } t';
\]

\[
\text{ID}(t, p) : \text{process } t \text{ is not capable of reaching time } p.
\]

The rules for the operational semantics have the form \( \frac{p_1, \ldots, p_m, s}{c_1, \ldots, c_n, s} \), where \( s \) is optional. They are to be read as "if \( p_1 \) and \( \ldots \) and \( p_m \) then \( c_1 \) and \( \ldots \) and \( c_n \), provided \( s'' \). As usual, \( p_1, \ldots, p_m \) and \( c_1, \ldots, c_n \) are called the premises and the conclusions, respectively. The conclusions are positive formulas of the form \( \langle t, p \rangle \xrightarrow{a} \langle t', p \rangle \), \( \langle t, p \rangle \xrightarrow{\sqrt{\cdot}} \langle t, p \rangle \xrightarrow{\cdot} \langle t', q \rangle \) or \( \text{ID}(t, p) \), where \( t \) and \( t' \) are open terms of BPA\textsuperscript{sat}. The premises are positive formulas of the above forms or negative formulas of the form \( \lnot \text{ID}(t, p) \). The rules are actually rule schemas. The optional \( s \) is a side-condition restricting the actions over which \( a, b \) and \( c \) range and the non-negative real numbers over which \( p, q \) and \( r \) range.
The signature of BPA\textsuperscript{sat} together with the rules that will be given constitute according to the definitions of [8] a term deduction system in \textit{panth} format that is \textit{stratifiable}. Within the framework of term deduction systems, the instances of the rule schemas that satisfy the stated side-conditions should be taken as the rules under consideration. For the rest, we continue to use the word rule in the broader sense.

A structured operational semantics of BPA\textsuperscript{sat} is described by the rules given in Table 4.

<table>
<thead>
<tr>
<th>(\text{ID}(\delta, p))</th>
<th>(\text{ID}(\delta, r))</th>
<th>(\text{ID}(\tilde{a}, 0))</th>
<th>(\text{ID}(\tilde{a}, r))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x, p) (\xrightarrow{a} (x', p))</td>
<td>(\sigma_{\text{abs}}^0(x, p) \xrightarrow{a} (x', p))</td>
<td>(\tilde{a}, 0) (\xrightarrow{a} (\sqrt{0}, 0))</td>
<td>(\sigma_{\text{abs}}^0(x, p) \xrightarrow{a} (\sqrt{0}, p))</td>
</tr>
<tr>
<td>(x, p) (\xrightarrow{a} (x', p))</td>
<td>(\sigma_{\text{abs}}^r(x, p) \xrightarrow{a} (x', p))</td>
<td></td>
<td>(x, p) (\xrightarrow{a} (\sqrt{p}, p))</td>
</tr>
<tr>
<td>((x, p) \vdash \sigma_{\text{abs}}^r(x, p) \xrightarrow{a} (x', p))</td>
<td>(\sigma_{\text{abs}}^r(x, p) \xrightarrow{a} (\sigma_{\text{abs}}^r(x', p) \vdash r))</td>
<td></td>
<td>(\sigma_{\text{abs}}^r(x, p) \xrightarrow{a} (\sqrt{p}, r))</td>
</tr>
<tr>
<td>(\sigma_{\text{abs}}^q(x, p) \vdash \tau \xrightarrow{a} (x, p \vdash r))</td>
<td>(\sigma_{\text{abs}}^q(x, p) \vdash \tau \xrightarrow{a} (\sigma_{\text{abs}}^q(x', p) \vdash q + r))</td>
<td>(\text{ID}(x, p))</td>
<td>(\sigma_{\text{abs}}^q(x, p) \vdash \tau \xrightarrow{a} (\sigma_{\text{abs}}^q(x, p) \vdash q + r))</td>
</tr>
<tr>
<td>(x, p) (\xrightarrow{a} (x', p))</td>
<td>(\sigma_{\text{abs}}^q(x, p) \vdash \tau \xrightarrow{a} (x', p))</td>
<td></td>
<td>(x, p) (\xrightarrow{a} (\sqrt{p}, p))</td>
</tr>
<tr>
<td>((x, p) \vdash (x + y, p) \xrightarrow{a} (x', p))</td>
<td>((x, y) \vdash (x, p) \xrightarrow{a} (x', p))</td>
<td></td>
<td>((x, p) \vdash (x + y, p) \xrightarrow{a} (\sqrt{p}, p))</td>
</tr>
<tr>
<td>((x, y, p) \vdash \tau \xrightarrow{a} (x, p \vdash (y + x, p + r)))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, p \vdash (y + x, p + r)))</td>
<td></td>
<td>((x, y, p) \vdash \tau \xrightarrow{a} (x', p))</td>
</tr>
<tr>
<td>((x, x') \vdash \tau \xrightarrow{a} (x, p \vdash (x, y, p + r)))</td>
<td>((x', y) \vdash \tau \xrightarrow{a} (x, p \vdash (x, y, p + r)))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, y) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, x') \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x', p))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x', p))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, p \vdash (x, y, p + r)))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, p \vdash \tau \xrightarrow{a} (x', p))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
<tr>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td>((x, p) \vdash \tau \xrightarrow{a} (x, y, p + r))</td>
<td></td>
<td>((x, p) \vdash \tau \xrightarrow{a} (\sqrt{p}, (y, p + q)))</td>
</tr>
</tbody>
</table>

Table 4: Rules for operational semantics of BPA\textsuperscript{sat} (\(a \in A, r > 0, p, q \geq 0\))
These rules are easy to understand. We will only explain the rules for the absolute delay operator (\(a_{abs}\)). The first pair of rules expresses that the action related capabilities of a process \(a_{abs}^p(x)\) at time \(p\) include those of process \(x\) at time \(p\). The second pair of rules expresses that the action related capabilities of a process \(a_{abs}^p(x)\) at time \(p + r\) include those of process \(x\) at time \(p\) shifted in time by \(r\) \((p \geq 0, r > 0)\). The third pair of rules expresses that the time related capabilities of a process \(a_{abs}^p(x)\) at time \(p + q\) include those of process \(x\) at time \(p\) shifted in time by \(q\) \((q \geq 0)\). The fourth pair of rules expresses that a process \(a_{abs}^p(x)\) can idle from any time \(p \geq 0\) to any time \(q < r\) and that it can also idle to time \(r\) provided that process \(x\) can reach time 0.

By identifying bisimilar processes we obtain our preferred model of BPA\(^{sat}\). One process is (strongly) bisimilar to another process means that if one of the processes is capable of doing a certain step, i.e. performing a certain action at a certain time or idling from a certain time to another, and next going on as a certain subsequent process then the other process is capable of doing the same step and next going on as a process bisimilar to the subsequent process. More precisely, a bisimulation on RTTS(A) is a symmetric binary relation \(R\) on the set of states \(S\) such that:

1. if \(R(s,t)\) and \(\langle s,p \rangle \xrightarrow{a} \langle s',p \rangle\), then there is a \(t'\) such that \(\langle t,p \rangle \xrightarrow{a} \langle t',p \rangle\) and \(R(s',t')\);
2. if \(R(s,t)\), then \(\langle s,p \rangle \xrightarrow{\langle \triangle \rangle} \langle s',p \rangle\) iff \(\langle t,p \rangle \xrightarrow{\langle \triangle \rangle} \langle s',p \rangle\);
3. if \(R(s,t)\) and \(\langle s,p \rangle \xleftarrow{a} \langle s',q \rangle\), then there is a \(t'\) such that \(\langle t,p \rangle \xleftarrow{a} \langle t',q \rangle\) and \(R(s',t')\);
4. if \(R(s,t)\), then \(\text{ID}(s,p)\) iff \(\text{ID}(t,p)\).

We say that two closed terms \(s\) and \(t\) are bisimilar, written \(s \cong t\), if there exists a bisimulation \(R\) such that \(R(s,t)\). It is known from [8] that if a term deduction system in panth format is stratifiable, bisimulation equivalence is a congruence for the operators in the signature concerned. Consequently, bisimulation equivalence is a congruence for the operators of BPA\(^{sat}\). For this reason, the operators of BPA\(^{sat}\) can be defined on the set of bisimulation equivalence classes. We can prove that this results in a model for BPA\(^{sat}\), i.e. all equations derivable in BPA\(^{sat}\) hold. As in connection with the other axiomatizations presented in this paper, we leave it as an open problem whether the axioms of BPA\(^{sat}\) form a complete axiomatization for this model.

2.2 Algebra of communicating processes

In ACP\(^{sat}\), we have, in addition to sequential and alternative composition, parallel composition of processes. The process \(x \parallel y\) is the process that proceeds with the processes \(x\) and \(y\) in parallel. Furthermore, we have the encapsulation operators \(\partial_H\) (one for each \(H \subseteq A\)) which turns all urgent actions \(\tilde{a}\), where \(a \in H\), into \(\tilde{a}\). As in ACP, we also have the auxiliary operators \(\parallel\) (left merge) and \(\mid\) (communication merge) to get a finite axiomatization of the parallel composition operator. The processes \(x \parallel y\) and \(x \mid y\) are the same except that \(x \parallel y\) must start to perform actions by performing an action of \(x\). The processes \(x \parallel y\) and \(x \mid y\) are the same except that \(x \parallel y\) must start to perform actions by performing an action of \(x\) and an action of \(y\) synchronously. In case of ACP\(^{sat}\), an additional auxiliary operator \(\nu_{abs}\) (absolute urgent initialization)
is needed. The process $\nu_{abs}(x)$ is the part of process $x$ that starts to perform actions at time 0.

The operator $\nu_{abs}$ of ACP$^{sat}$ is simply the operator $\nu_{abs}$ of ACP$^{dat}$ [4] lifted to the real time setting.

**Signature of ACP$^{sat}$** The signature of ACP$^{sat}$ is the signature of BPA$^{sat}$ extended with the parallel composition operator $\|: P \times P \rightarrow P$, the left merge operator $\llcap: P \times P \rightarrow P$, the communication merge operator $\ll merge: P \times P \rightarrow P$, for each $H \subseteq A$, and the absolute urgent initialization operator $\nu_{abs}: P \rightarrow P$.

**Axioms of ACP$^{sat}$** The axiom system of ACP$^{sat}$ consists of the axioms of BPA$^{sat}$ and the equations given in Table 5.

Axioms CM1, CM4, CM8, CM9, D3 and D4 are common to ACP and all real and discrete time versions of ACP. Axioms CF1SA, CF2SA, CM2SA, CM3SA, CM5SA-CM7SA, D1SA and D2SA are simple reformulations of the axioms CP1, CF2, CM2, CM3, CM5-CM7, D1 and D2 of ACP: constants $a$ ($a \in A_\delta$) have been replaced by constants $\check{a}$, and in addition to that certain variables $x$ have been replaced by $x + \delta$ in CM2SA and CM3SA. Recall that $x + \delta = \delta$ if $x \neq \delta$, and $\delta + \delta = \delta$. This means that $x + \delta$ never stands for $\delta$. Axioms SACM1 and SACM2 represent the interaction of absolute delay with left merge. Axiom SACM2 shows that if two parallel processes start to perform actions by performing an action of one of them and that process starts to perform actions at a certain time, only the part of the other process proceeds that starts to perform actions at the same time or later. What happens if such a part does not exist, is reflected more clearly by a generalization of axiom SACM1 than by that axiom itself. This generalization, which is derivable from the axioms of ACP$^{sat}$, is equation SACM1' given in Table 6. Note that a term of the form $\nu_{abs}(\check{y})$ stands for an arbitrary process that starts to perform actions before time $p$; and that a term of the form $\sigma_{\nu_{abs}}(\nu_{abs}(z) + \delta)$ stands for an arbitrary process that starts to perform actions at time $p$ or deadlocks at time $p$. So equation SACM1 expresses that if the process that would perform the first action can only do so after the ultimate starting time to start performing actions or to deadlock for the other process, the result will be a deadlock at this ultimate starting time. Note that in case of sequential processes, the process that would first perform actions can always do so, irrespective of the ultimate starting time for the other process. This difference is apparent from equation SAT4', given in Table 6, which is derivable from the axioms of ACP$^{sat}$ and the standard initialization axioms SI13 and SI16 (Table 14, page 21). Axioms SACM3-SACM5 represent the interaction of absolute delay with communication merge. Axioms SACM4 and SACM5 are similar to axioms SACM1 and SACM2. Axiom SACM3 is needed as well because communication merge requires that both processes concerned start performing actions at the same time. Axiom SACM5 is simpler than axiom SACM2 just because of the left distributivity of the communication merge (axiom CM9). Equations SACM3' and SACM4' given in Table 6 generalize axioms SACM3 and SACM4 like equation SACM1' generalizes axiom SACM1. Axiom SAD represents the (lack of) interaction of absolute delay with encapsulation. Axioms SAU0-SAU4 reflect the intended meaning of the urgent initialization operator clearly.

We can prove that the operators $\|, \llcap, \partial_H$ and $\nu_{abs}$ can be eliminated in closed terms of ACP$^{sat}$. Because of the elimination result for BPA$^{sat}$, we are permitted to
\[
\begin{align*}
\tilde{a} \parallel \tilde{b} &= \tilde{c} \quad \text{if } \gamma(a, b) = c \quad \text{CF1SA} \\
\tilde{a} \parallel \tilde{b} &= \tilde{d} \quad \text{if } \gamma(a, b) \text{ undefined} \quad \text{CF2SA} \\
x \parallel y &= (x \parallel y \parallel y) + x \parallel y \quad \text{CM1} \\
\tilde{d} \parallel x &= \tilde{d} \quad \text{CMID1} \\
x \parallel \tilde{d} &= \tilde{d} \quad \text{CMID2} \\
\tilde{a} \parallel (x + \tilde{d}) &= \tilde{a} \cdot (x + \tilde{d}) \quad \text{CM2SA} \\
\tilde{a} \cdot x \parallel (y + \tilde{d}) &= \tilde{a} \cdot (x \parallel (y + \tilde{d})) \quad \text{CM3SA} \\
\sigma^p_{abs}(x) \parallel (\nu_{abs}(y) + \tilde{d}) &= \tilde{d} \quad \text{SACM1} \\
\sigma^p_{abs}(x) \parallel (\nu^p_{abs}(y) + \sigma^p_{abs}(z)) &= \sigma^p_{abs}(x \parallel z) \quad \text{SACM2} \\
(x + y) \parallel z &= x \parallel z + y \parallel z \quad \text{CM4} \\
\tilde{d} \parallel x &= \tilde{d} \quad \text{CMID3} \\
x \parallel \tilde{d} &= \tilde{d} \quad \text{CMID4} \\
\tilde{a} \cdot x \parallel \tilde{b} &= (\tilde{a} \parallel \tilde{b}) \cdot x \quad \text{CM5SA} \\
\tilde{a} \parallel \tilde{b} \cdot x &= (\tilde{a} \parallel \tilde{b}) \cdot x \quad \text{CM6SA} \\
\tilde{a} \cdot x \parallel \tilde{b} \cdot y &= (\tilde{a} \parallel \tilde{b}) \cdot (x \parallel y) \quad \text{CM7SA} \\
(\nu_{abs}(x) + \tilde{d}) \parallel \sigma^p_{abs}(y) &= \tilde{d} \quad \text{SACM3} \\
\sigma^p_{abs}(x) \parallel (\nu_{abs}(y) + \tilde{d}) &= \tilde{d} \quad \text{SACM4} \\
\sigma^p_{abs}(x) \parallel (\sigma^p_{abs}(y) + \sigma^p_{abs}(z)) &= \sigma^p_{abs}(x \parallel y) \quad \text{SACM5} \\
(x + y) \parallel z &= x \parallel z + y \parallel z \quad \text{CM8} \\
x \parallel (y + z) &= x \parallel y + x \parallel z \quad \text{CM9} \\
\partial_H(\tilde{d}) &= \tilde{d} \quad \text{D0} \\
\partial_H(\tilde{a}) &= \tilde{a} \quad \text{if } a \notin H \quad \text{D1SA} \\
\partial_H(\tilde{a}) &= \tilde{d} \quad \text{if } a \in H \quad \text{D2SA} \\
\partial_H(\sigma^p_{abs}(x)) &= \sigma^p_{abs}(\partial_H(x)) \quad \text{SAD} \\
\partial_H(x + y) &= \partial_H(x) + \partial_H(y) \quad \text{D3} \\
\partial_H(x \cdot y) &= \partial_H(x) \cdot \partial_H(y) \quad \text{D4} \\
\nu_{abs}(\tilde{d}) &= \tilde{d} \quad \text{SAU0} \\
\nu_{abs}(\tilde{a}) &= \tilde{a} \quad \text{SAU1} \\
\nu_{abs}(\sigma^p_{abs}(x)) &= \tilde{d} \quad \text{SAU2} \\
\nu_{abs}(x + y) &= \nu_{abs}(x) + \nu_{abs}(y) \quad \text{SAU3} \\
\nu_{abs}(x \cdot y) &= \nu_{abs}(x) \cdot y \quad \text{SAU4}
\end{align*}
\]

Table 5: Additional axioms for ACP\textsuperscript{sat} \((a, b \in A, c \in A, p \geq 0, r > 0)\)

\[
\begin{align*}
\sigma^p_{abs}(x) \cdot (\nu^p_{abs}(y) + \sigma^p_{abs}(\nu_{abs}(x) + \tilde{d})) &= \sigma^p_{abs}(x \cdot \tilde{d}) \quad \text{SAT4'} \\
\sigma^p_{abs}(x) \parallel (\nu^p_{abs}(y) + \sigma^p_{abs}(\nu_{abs}(x) + \tilde{d})) &= \sigma^p_{abs}(\tilde{d}) \quad \text{SACM1'} \\
\sigma^p_{abs}(\nu_{abs}(x) + \tilde{d}) \parallel \sigma^p_{abs}(y) &= \sigma^p_{abs}(\tilde{d}) \quad \text{SACM3'} \\
\sigma^p_{abs}(x) \parallel \sigma^p_{abs}(\nu_{abs}(y) + \tilde{d}) &= \sigma^p_{abs}(\tilde{d}) \quad \text{SACM4'}
\end{align*}
\]

Table 6: Some derivable equations \((p \geq 0, r > 0)\)

use induction on the structure of basic terms over BPA\textsuperscript{sat} to prove statements for all closed terms of ACP\textsuperscript{sat}.

**Examples** We give some examples of a closed term of ACP\textsuperscript{sat} and the corresponding basic term (in case \(\gamma(a, b)\) and \(\gamma(a, c)\) are undefined):

13
The structured operational semantics of $ACP^{sat}$ is described by the rules for BPA$^{sat}$ and the rules given in Table 7.

These rules are easy to understand. We will only mention that the first two rules for the parallel composition operator ($\parallel$) express that a process $x$ loses all its action related capabilities at time $p$ if it is put in parallel with a process $y$ that can not reach time $p$, and that the last rule for this operator expresses that in such cases the parallel composition can not reach time $p$ either. As in the case of BPA$^{sat}$, we obtain a term deduction system in panth format that is stratifiable, so bisimulation equivalence is also a congruence for the additional operators of $ACP^{sat}$. Therefore, these operators
can be defined on the set of bisimulation equivalence classes as well. As in the case of \( \text{BPA}^{\text{sat}} \), we can prove that this results in a model for \( \text{ACP}^{\text{sat}} \).

### 2.3 Time-stamped actions

The real time versions \( \text{ACP}_p \), \( \text{ACP}_{rp} \) and \( \text{ACP}_{p\sigma} \) \([1, 3]\) feature time-stamped actions – and thus combine execution of actions and passage of time. The time-stamped actions defined below are more closely related to the ones in \( \text{ACP}_{p\sigma} \) \([5]\), the real time version of \( \text{ACP} \) from which \( \text{ACP}^{\text{sat}} \) originates. This is because \( \text{ACP}_{p\sigma} \), like \( \text{ACP}^{\text{sat}} \) and unlike \( \text{ACP}_p \), \( \text{ACP}_{rp} \) and \( \text{ACP}_{p\sigma} \), does not exclude the possibility of two or more actions to be performed consecutively at the same point in time.

Time-stamped actions are defined in terms of urgent actions and the delay operator in Table 8. We also define a time-stamped version of immediate deadlock. In \( \text{ACP}_p \)

\[
\begin{align*}
\tilde{a}(p) &= \sigma_{\text{abs}}^a(\tilde{a}) \\
\tilde{b}(p) &= \sigma_{\text{abs}}^b(\tilde{b})
\end{align*}
\]

Table 8: Definitions of time-stamped actions and immediate deadlock \((a \in A_d, p \geq 0)\)

and \( \text{ACP}_{rp} \), which exclude the possibility of two or more actions to be performed consecutively at the same point in time, there is no reason to distinguish, for instance, between the processes \( a(p+r) \cdot \delta(p+r) \) and \( a(p+r) \cdot b(p) \) \((r > 0)\). In \( \text{ACP}^{\text{sat}} \), unlike in \( \text{ACP}_{p\sigma} \), distinction is made in comparable cases by introducing immediate deadlock to deal with timing inconsistencies. Therefore, we have to introduce time-stamped immediate deadlock here.

We now consider the signature of \( \text{BPA}^{\text{sat}} \) with time-stamped actions, i.e. the signature of \( \text{BPA}^{\text{sat}} \), but with the urgent action constants \( \tilde{a} \), the urgent deadlock constant \( \tilde{\delta} \), the immediate deadlock constant \( \tilde{\delta} \) and the delay operator \( \sigma_{\text{abs}} \) replaced by the time-stamped action constants \( \tilde{a}(p) \) and the time-stamped deadlock constants \( \tilde{\delta}(p) \) and \( \tilde{\delta}(p) \). From the axioms of \( \text{BPA}^{\text{sat}} \) and the definitions of time-stamped actions and immediate deadlock, we can easily derive the equations given in Table 9 for closed terms. Axioms A1-A5 from Table 1 and the equations from Table 9 together can be considered to form the axioms of \( \text{BPA}^{\text{sat}} \) with time-stamped actions. The differences with the axioms of \( \text{BPAsp}_\delta \) in \([5]\) are all due to the different treatment of timing inconsistencies. Extension of this version to \( \text{ACP} \) is left to the reader.

### 3 Extension of \( \text{ACP}^{\text{sat}} \)

In this section, we describe the extension of \( \text{ACP}^{\text{sat}} \) with integration and initial abstraction. The extension with integration is needed to be able to embed discrete time process algebras, as exemplified in Section 5.3. The extension with initial abstraction is needed to be able to embed process algebras with relative timing, as illustrated in Section 4.3.

Integration and initial abstraction are both variable binding operators. Following e.g. \([20]\), we will introduce variable binding operators by a declaration of the form

\[
f : S_{11}, \ldots, S_{1k} \cdot S_1 \times \ldots \times S_{n1}, \ldots, S_{nk} \cdot S_n \rightarrow S.
\]

Hereby is indicated that \( f \) combines an operator \( f^* : ((S_{11} \times \ldots \times S_{1k}) \rightarrow S_1) \times \ldots \times ((S_{n1} \times \ldots \times S_{nk}) \rightarrow S_n) \rightarrow S \).
$S_n \rightarrow S$ with $\lambda$-calculus-like functional abstraction, binding $k_i$ variables ranging over $S_{i1}, \ldots, S_{ik_i}$ in the $i$th argument ($0 \leq i \leq n$). Applications of $f$ have the following form: $f(x_{i1}, \ldots, x_{ik_i}, t_1, \ldots, x_{n1}, \ldots, x_{nk_n}, t_n)$, where each $x_{ij}$ is a variable of sort $S_{ij}$ and each $t_i$ is a term of sort $S_i$.

Integration requires a more extensive theory of the non-negative real numbers than the minimal theory sketched at the beginning of Section 2 (page 4). In the first place, it has to include a theory of sets of non-negative real numbers that makes it possible to deal with set membership and set equality. Besides, the theory should cover suprema of sets of non-negative real numbers.

First, in Section 3.1, $ACP^{sat}$ is extended with integration. After that, in Section 3.2, initial abstraction is added. Finally, some useful additional axioms, derivable for closed terms, are given in Section 3.3.

### 3.1 Integration

We add the integration operator $\int$ to $ACP^{sat}$. It provides for alternative composition over a continuum of alternatives. That is, $\int_{v \in V} P$, where $v$ is a variable ranging over $\mathbb{R}_{\geq 0}$, $V \subseteq \mathbb{R}_{\geq 0}$ and $P$ is a term that may contain free variables, proceeds as one of the alternatives $P[p/v]$ for $p \in V$. The resulting theory is called $ACP^{sat\int}$. Obviously, we could first have added integration to $BPA^{sat}$, resulting in $BPA^{sat\int}$, and then have extended $BPA^{sat\int}$ to deal with parallelism and communication.

**Signature of $ACP^{sat\int}$** The signature of $ACP^{sat\int}$ is the signature of $ACP^{sat}$ extended with the *integration* (variable-binding) operator $\int : \mathcal{P}(\mathbb{R}_{\geq 0}) \times \mathbb{R}_{\geq 0} \rightarrow \mathcal{P}$.

We assume that an infinite set of *time variables* ranging over $\mathbb{R}_{\geq 0}$ has been given, and denote them by $v, w, \ldots$. Furthermore, we use $V, W, \ldots$ to denote subsets of $\mathbb{R}_{\geq 0}$. We denote terms of $ACP^{sat\int}$ by $P, Q, \ldots$ We will use the following notational convention. We write $\int_{v \in V} P$ for $\int(V, v \cdot P)$.

**Axiom system of $ACP^{sat\int}$** The axiom system of $ACP^{sat\int}$ consists of the axioms of $ACP^{sat}$ and the equations given in Table 10.
\[
\begin{align*}
\int_{\emptyset} P &= \delta & \text{INT1} \\
\int_{\emptyset}(p) P &= P[p/v] & \text{INT2} \\
\int_{\emptyset} E V \cup W P &= \int_{\emptyset} E V P + \int_{\emptyset} E W P & \text{INT3} \\
V \neq \emptyset \Rightarrow \int_{\emptyset} E V R &= R & \text{INT4} \\
(V \in V \cdot P[p/v] = Q[p/v]) \Rightarrow \int_{\emptyset} E V P &= \int_{\emptyset} E V Q & \text{INT5} \\
V \neq \emptyset \Rightarrow \int_{\emptyset} E V \sigma_{\text{abs}}(\delta) &= \sigma_{\text{sup \ V}}(\delta) & \text{INT6} \\
V \neq \emptyset, \sup V \notin V \Rightarrow \int_{\emptyset} E V \sigma_{\text{abs}}(\delta) &= \sigma_{\text{sup \ V}}(\delta) & \text{INT7} \\
\sup V \in V \Rightarrow \int_{\emptyset} E V \sigma_{\text{abs}}(\delta) &= \sigma_{\text{sup \ V}}(\delta) & \text{INT8} \\
\int_{\emptyset} E V \sigma_{\text{abs}}(P) &= \sigma_{\text{abs}}(\int_{\emptyset} E V P) \quad \text{if } p \neq v & \text{INT9} \\
\int_{\emptyset} E V (P + Q) &= \int_{\emptyset} E V P + \int_{\emptyset} E V Q & \text{INT10} \\
\int_{\emptyset} E V (P \cdot R) &= (\int_{\emptyset} E V P) \cdot R & \text{INT11} \\
\int_{\emptyset} E V (P \uplus R) &= (\int_{\emptyset} E V P) \uplus R & \text{INT12} \\
\int_{\emptyset} E V (P | R) &= (\int_{\emptyset} E V P) | R & \text{INT13} \\
\int_{\emptyset} E V (R | P) &= R | (\int_{\emptyset} E V P) & \text{INT14} \\
\int_{\emptyset} E V \theta_H(P) &= \theta_H(\int_{\emptyset} E V P) & \text{INT15} \\
\nu_{\text{abs}}(\int_{\emptyset} E V P) &= \int_{\emptyset} E V \nu_{\text{abs}}(P) \quad \text{if } p \neq v & \text{SAI06} \\
\nu_{\text{abs}}(\int_{\emptyset} E V P) &= \int_{\emptyset} E V \nu_{\text{abs}}(P) \quad \text{if } p \neq v & \text{SAI16} \\
\nu_{\text{abs}}(\int_{\emptyset} E V P) &= \int_{\emptyset} E V \nu_{\text{abs}}(P) & \text{SAU05}
\end{align*}
\]

Table 10: Axioms for integration \((p \geq 0, v \text{ not free in } R)\)

Axioms INT1-INT5 are the crucial axioms of integration. They reflect the informal explanation that \(\int_{\emptyset} E V P\) proceeds as one of the alternatives \(P[p/v]\) for \(p \in V\). The remaining axioms are all easily understood by realizing that \(\int\) stands for an infinite alternative composition.

We can prove that the auxiliary operators \(\nu_{\text{abs}}\) and \(\overline{\nu}_{\text{abs}}\), as well as sequential compositions in which the form of the first operand is not \(\overline{a}\) \((a \in A)\) and alternative compositions in which the form of the first operand is \(\sigma(\overline{a}, t)\), can be eliminated in closed terms of BPA\text{sat}I with a restricted form of integration. Basically, this restriction means that in terms of the form \(\int_{\emptyset} E V P, V\) is an interval of which the bounds are given by linear expressions over time variables and \(P\) is of the form \(\sigma_{\text{abs}}(\overline{a})\) or \(\sigma_{\text{abs}}(\overline{a}) \cdot t\) \((a \in A_{\delta})\). This restricted form of integration is essentially the same as prefixed integration from [24] (see also [19]). The terms that remain after exhaustive elimination are called the basic terms over BPA\text{sat} with restricted integration. We can also prove that the operators \(\|, \uplus, |, \theta_H\) and \(\nu_{\text{abs}}\) can be eliminated in closed terms of ACP\text{sat} with restricted integration. Because of these elimination results, we are permitted to use induction on the structure of basic terms over BPA\text{sat} with restricted integration to prove statements for all closed terms of ACP\text{sat} with restricted integration.

**Examples** We give some examples of a closed term of ACP\text{sat} with restricted integration and the corresponding basic term:

\[
\begin{align*}
\int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\nu_{\text{abs}}(\sigma_{\text{abs}}(\overline{a})) \| \sigma_{\text{abs}}(\overline{b}) \cdot \sigma_{\text{abs}}(\overline{c}))) &= \sigma_{\text{abs}}(\delta) \\
\int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{a}) + \int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{b}) &= \int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{a} + \overline{b}) \\
(\int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{a})) | (\int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{b})) &= \int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{c}) \quad \text{if } \gamma(a, b) = c \\
(\int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{a})) | (\int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{b})) &= \int_{\emptyset} E V^{[4,5,5,1]} \sigma_{\text{abs}}(\overline{a}) \quad \text{if } \gamma(a, b) \text{ undefined}
\end{align*}
\]
Semantics of \( \text{ACP}^{\text{satI}} \) The structured operational semantics of \( \text{ACP}^{\text{satI}} \) is described by the rules for \( \text{ACP}^{\text{sat}} \) and the rules given in Table 11.

\[
\begin{array}{c}
\langle P[q/v], p \rangle \xrightarrow{\alpha} \langle P', p \rangle, \quad q \in V \\
\langle \int_{v \in V} P, p \rangle \xrightarrow{\alpha} \langle P', p \rangle, \\
\langle P[q/v], p \rangle \trans{r} \langle P[q/v], p + r \rangle, \quad q \in V \\
\langle \int_{v \in V} P, p \rangle \trans{r} \langle \int_{v \in V} P, p + r \rangle \\
\end{array}
\]

Table 11: Rules for integration \( (a \in A, \quad r > 0, \quad p, q \geq 0) \)

The rules for integration are simple generalizations of the rules for alternative composition to the infinite case. We can reformulate them as rules of a term deduction system in panth format that is stratifiable, so bisimulation equivalence is also a congruence for the integration operator. Hence, this operator can be defined on the set of bisimulation equivalence classes as well. As in the case of \( \text{BPA}^{\text{sat}} \) and \( \text{ACP}^{\text{sat}} \), we can prove that this results in a model for \( \text{ACP}^{\text{satI}} \). We will call this model \( M_A \).

3.2 Initial abstraction

We add the initial abstraction operator \( \sqrt{\cdot} \) to \( \text{ACP}^{\text{satI}} \). It provides for (simple) parametric timing: \( \sqrt{v} \cdot F \), where \( v \) is a variable ranging over \( \mathbb{R}_{\geq 0} \) and \( F \) is a term that may contain free variables, proceeds as \( F[p/v] \) if initialized at time \( p \in \mathbb{R}_{\geq 0} \). This means that \( \sqrt{v} \cdot F \) denotes a function \( f : \mathbb{R}_{\geq 0} \rightarrow P \) that satisfies \( f(p) = \sqrt{p}(f(p)) \) for all \( p \in \mathbb{R}_{\geq 0} \). In the resulting theory, called \( \text{ACP}^{\text{satIV}} \), the sort \( P \) of processes is replaced by the sort \( P^* \) of parametric time processes. Of course, it is also possible to add the initial abstraction operator to \( \text{ACP}^{\text{sat}} \), resulting in a theory \( \text{ACP}^{\text{satIV}} \).

Signature of \( \text{ACP}^{\text{satIV}} \) The signature of \( \text{ACP}^{\text{satIV}} \) is the signature of \( \text{ACP}^{\text{satI}} \) extended with the initial abstraction (variable-binding) operator \( \sqrt{\cdot} : \mathbb{R}_{\geq 0} \rightarrow P^* \rightarrow P^* \).

We now use \( x, y, \ldots \) to denote variables of sort \( P^* \). Terms of \( \text{ACP}^{\text{satIV}} \) are denoted by \( F, G, \ldots \). We will use the following notational convention. We write \( \sqrt{\cdot} \cdot F \) for \( \sqrt{v} \cdot F \).

Axiom system of \( \text{ACP}^{\text{satIV}} \) The axiom system of \( \text{ACP}^{\text{satIV}} \) consists of the axioms of \( \text{ACP}^{\text{satI}} \) and the equations given in Table 12.

Axioms SIA1 and SIA2 are similar to the \( \alpha^{-} \) and \( \beta^{-} \)-conversion rules of \( \lambda \)-calculus. Axiom SIA3 points out that multiple initial abstractions can simply be replaced by one. Axiom SIA4 shows that processes with absolute timing can be treated as special cases of processes with parametric timing: they do not vary with different initialization times. Axiom SIA5 is an extensionality axiom. Axiom SIA6 expresses that in case a process performs an action and then proceeds as another process, the initialization time of the latter process is the time at which the action is performed. Notice that the equation \( \sqrt{\cdot} \cdot x = \sqrt{x} \cdot \sqrt{0} \) is a special case of axiom SIA6. The related equation \( \sigma^p_{\text{abs}}(x) = \sigma^p_{\text{abs}}(\sqrt{0}(x)) \) follows immediately from axioms SAT1 and SAT2 (Table 2, page 6). Axioms SIA7-SIA17 become easier to understand by realizing that \( \sqrt{\cdot} \cdot F \) denotes a function \( f : \mathbb{R}_{\geq 0} \rightarrow P \) such that \( f(p) = \sqrt{p}(f(p)) \) for all \( p \in \mathbb{R}_{\geq 0} \). This is reflected by the equation.
\[ v \cdot G = \sqrt{v} \cdot G[v/w] \]  
SIA1
\[ \overline{v}_{\text{abs}} (\sqrt{v} \cdot F) = \overline{v}_{\text{abs}} (F[p/v]) \]  
SIA2
\[ \sqrt{v} \cdot (\sqrt{w} \cdot F) = \sqrt{v} \cdot F[v/w] \]  
SIA3
\[ G = \sqrt{v} \cdot G \]  
SIA4
\[ (\forall p \in \mathbb{R}_{\geq 0} \bullet \overline{v}_{\text{abs}} (x) = \overline{v}_{\text{abs}} (y)) \Rightarrow x = y \]  
SIA5
\[ \sigma_{\text{abs}} (\overline{a}) \cdot x = \sigma_{\text{abs}} (\overline{a}) \cdot \overline{v}_{\text{abs}} (x) \]  
SIA6
\[ \sigma_{\text{abs}} (\sqrt{v} \cdot F) = \sigma_{\text{abs}} (F[0/v]) \]  
SIA7
\[ (\sqrt{v} \cdot F) + G = \sqrt{v} \cdot (F + \overline{v}_{\text{abs}} (G)) \]  
SIA8
\[ (\sqrt{v} \cdot F) \cdot G = \sqrt{v} \cdot (F \cdot G) \]  
SIA9
\[ \nu_{\text{abs}} (\sqrt{v} \cdot F) = \sqrt{v} \cdot \nu_{\text{abs}} (F) \]  
SIA10
\[ (\sqrt{v} \cdot F) \parallel G = \sqrt{v} \cdot (F \parallel \overline{v}_{\text{abs}} (G)) \]  
SIA11
\[ (\sqrt{v} \cdot F) \mid G = \sqrt{v} \cdot (F \mid \overline{v}_{\text{abs}} (G)) \]  
SIA12
\[ G \mid (\sqrt{v} \cdot F) = \sqrt{v} \cdot (\overline{v}_{\text{abs}} (G) \mid F) \]  
SIA13
\[ \delta_{H} (\sqrt{v} \cdot F) = \sqrt{v} \cdot \delta_{H} (F) \]  
SIA14
\[ \nu_{\text{abs}} (\sqrt{v} \cdot F) = \sqrt{v} \cdot \nu_{\text{abs}} (F) \]  
SIA15
\[ \int_{v \in V} (\sqrt{w} \cdot F) = \sqrt{w} \cdot (\int_{v \in V} F) \]  
SIA16

Table 12: Axioms for standard initial abstraction \((p \geq 0, v \text{ not free in } G)\)

\[ \sqrt{v} \cdot F = \sqrt{v} \cdot \overline{v}_{\text{abs}} (F) \]  
SIA1

which can be derived – without using axioms SIA7-SIA17 – for all closed terms \(F\) by means of the extensionality axiom.

The elimination results for \(\text{ACP}^{\text{satIv}}\) with the restricted form of integration mentioned in Section 3.1 are essentially the same as the ones for \(\text{ACP}^{\text{satI}}\) with the restricted form of integration. Besides, all closed terms of \(\text{ACP}^{\text{satIv}}\) with this restricted form of integration can be written in the form \(\sqrt{v} \cdot F\) where \(F\) is a basic term over \(\text{BPA}^{\text{sat}}\) with restricted integration.

**Examples** We give some examples of a closed term of \(\text{ACP}^{\text{satIv}}\) with restricted integration, the corresponding term of the form \(\sqrt{v} \cdot F\) where \(F\) is a basic term and, if possible, the corresponding basic term without initial abstraction:

\[ \sqrt{v} \cdot \nu_{\text{abs}}^{\mathbb{R}_{>0}} (\sqrt{w} \cdot \sigma_{\text{abs}} (\overline{a})) = \sqrt{v} \cdot \sigma_{\text{abs}}^{\mathbb{R}_{>0}} (\overline{a}) \]  
\[ \sqrt{v} \cdot \nu_{\text{abs}}^{\mathbb{R}_{>0}} (\sqrt{w} \cdot \sigma_{\text{abs}} (\overline{a})) = \sqrt{v} \cdot \sigma_{\text{abs}}^{\mathbb{R}_{>0}} (\overline{a}) \]  
\[ \overline{v}_{\text{abs}} (\sqrt{v} \cdot \nu_{\text{abs}}^{\mathbb{R}_{>0}} (\sqrt{w} \cdot \sigma_{\text{abs}} (\overline{a}))) = \nu_{\text{abs}} (\sqrt{v} \cdot \sigma_{\text{abs}}^{\mathbb{R}_{>0}} (\overline{a})) = \sigma_{\text{abs}}^{\mathbb{R}_{>0}} (\overline{a}) \]  
\[ \overline{v}_{\text{abs}}^{\mathbb{R}_{>0}} (\sqrt{v} \cdot \nu_{\text{abs}}^{\mathbb{R}_{>0}} (\sqrt{w} \cdot \sigma_{\text{abs}} (\overline{a}))) = \nu_{\text{abs}} (\sqrt{v} \cdot \int_{v \in [6,6.1]} \sigma_{\text{abs}} (\overline{a})) = \int_{v \in [6,6.1]} \sigma_{\text{abs}} (\overline{a}) \]

On the basis of the rules for its operational semantics, the operators of \(\text{ACP}^{\text{satIv}}\) can also be directly defined on real time transition systems in a straightforward way. In the following, we will describe a model of \(\text{ACP}^{\text{satIv}}\) in terms of these operators.

**Semantics of \(\text{ACP}^{\text{satIv}}\)** We have to extend \(\text{RTTS}(A)\) to the function space

\[ \text{RTTS}^{*}(A) = \{ f : \mathbb{R}_{\geq 0} \rightarrow \text{RTTS}(A) \mid \forall p \in \mathbb{R}_{\geq 0} \bullet f(p) = \overline{v}_{\text{abs}} (f(p)) \} \]

of real time transition systems with parametric timing. We use \(f, g, \ldots\) to denote elements of \(\text{RTTS}^{*}(A)\). In Table 13, the constants and operators of \(\text{ACP}^{\text{satIv}}\) are
defined on \( \text{RTTS}^*(A) \). We use \( \lambda \)-notation for functions – here \( t \) is a variable ranging over \( \mathbb{R}_{\geq 0} \). We write \( f(t) \ast g \) for the real time transition system obtained from \( f(t) \) by replacing \( (s, p) \xrightarrow{a} (s', p) \) by \( (s, p) \xrightarrow{a} (s', p) \), where \( s' \) is the root state of \( g(p) \), whenever \( s \) is reachable from the root state of \( f(t) \).

<table>
<thead>
<tr>
<th>Table 13: Definition of operators on ( \text{RTTS}^* ) (( \phi : \mathbb{R}<em>{\geq 0} \rightarrow \text{RTTS}^*(A) ), ( a \in A_S, p \in \mathbb{R}</em>{\geq 0} ))</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \dot{\delta} = \lambda t \cdot \delta )</td>
</tr>
<tr>
<td>( \dot{a} = \lambda t \cdot \dot{v}_{\text{abs}}(\dot{a}) )</td>
</tr>
<tr>
<td>( \sigma_{\text{abs}}^p(f) = \lambda t \cdot \dot{v}<em>{\text{abs}}^t(\sigma</em>{\text{abs}}^p(f(0))) )</td>
</tr>
<tr>
<td>( f + g = \lambda t \cdot (f(t) + g(t)) )</td>
</tr>
<tr>
<td>( f \cdot g = \lambda t \cdot (f(t) \ast g) )</td>
</tr>
<tr>
<td>( \nu_{\text{abs}}^p(f) = \lambda t \cdot \dot{v}<em>{\text{abs}}^t(\nu</em>{\text{abs}}^p(f(t))) )</td>
</tr>
<tr>
<td>( \nu_{\text{abs}}^0(f) = f(p) )</td>
</tr>
<tr>
<td>( f \parallel g = \lambda t \cdot (f(t) \parallel g(t)) )</td>
</tr>
<tr>
<td>( f \parallel g = \lambda t \cdot (f(t) \parallel g(t)) )</td>
</tr>
<tr>
<td>( f</td>
</tr>
<tr>
<td>( \partial_H(f) = \lambda t \cdot \partial_H(f(t)) )</td>
</tr>
<tr>
<td>( \nu_{\text{abs}}(f) = \lambda t \cdot \dot{v}<em>{\text{abs}}^t(\nu</em>{\text{abs}}(f(t))) )</td>
</tr>
<tr>
<td>( \int_{v \in V}(f) = \lambda t \cdot \int_{v \in V}(f(t)) )</td>
</tr>
<tr>
<td>( \sqrt{\phi} = \lambda t \cdot \dot{v}_{\text{abs}}^t(\sqrt{\phi(t)}) )</td>
</tr>
</tbody>
</table>

We say that \( f, g \in \text{RTTS}^*(A) \) are bisimilar if for all \( p \in \mathbb{R}_{\geq 0} \), there exists a bisimulation \( R \) such that \( R(f(p), g(p)) \). We obtain a model of \( \text{ACP}^{\text{satIv}} \) by defining all operators on the set of bisimulation equivalence classes. We will call this model \( M_A^* \). Notice that \( f \in \text{RTTS}^*(A) \) corresponds to a process that can be written with the constants and operators of \( \text{ACP}^{\text{satI}} \) only iff \( \nu_{\text{abs}}^\phi(f) = f \). In fact, \( M_A \) is isomorphic to a subalgebra of \( M_A^* \).

### 3.3 Standard initialization axioms

In Table 14, some equations concerning initialization and time-out are given that hold in the model \( M_A^* \), and that are derivable for closed terms of \( \text{ACP}^{\text{satIv}} \). We will use these axioms in proofs in subsequent sections. Notice that the very useful equation \( \nu_{\text{abs}}^p(\nu_{\text{abs}}^p(x)) = \nu_{\text{abs}}^p(x) \) is a special case of axiom SI2. We can easily prove, using equation SIA1 (page 19) and the standard initialization axioms, that initial abstraction distributes over \( _, \parallel, [\_ \_ ] \) and \( _ \square_ \):

\[
(\sqrt{\_ v \cdot F}) \square (\sqrt{\_ v \cdot F'}) = \sqrt{\_ v \cdot (F \square F')} \quad \text{DISTR}\square
\]

for \( \square = _, \parallel, [\_ \_ ] \). Using this fact shortens many of the calculations needed in the proof of Theorem 6 (embedding of \( \text{ACP}^{\text{satI}} \) in \( \text{ACP}^{\text{satv}} \)).

### 4 Real time process algebra: relative timing

In this section, we give the signature, axioms and term model of \( \text{ACP}^{\text{satI}} \), a standard real time process algebra with relative timing. \( \text{ACP}^{\text{satI}} \) originates from the theory
ACPst, presented in [5]. Like ACPst, it separates execution of actions and passage of time.

First, in Section 4.1, we treat BPA"!, basic standard real time process algebra with relative timing, in which parallelism and communication are not considered. After that, in Section 4.2, BPA"! is extended to ACP"! to deal with parallelism and communication as well. Finally, we show in Section 4.3 how ACP"! can be embedded in ACPsat\lor.

### 4.1 Basic process algebra

In BPA"!, we have the constants \( \tilde{a} \) and \( \tilde{\delta} \) instead of \( a \) and \( \delta \), and the operator \( \sigma_{rel} \) (relative delay) instead of \( \sigma_{abs} \) (absolute delay). The constants \( \tilde{a} \) and \( \tilde{\delta} \) stand for an urgent action without any delay and a deadlock without any delay, respectively. The process \( \sigma_{rel}^p(x) \) is the process \( x \) delayed for a period of time \( p \). We also have relative counterparts of the absolute time-out and initialization operators: \( \nu_{rel} \) (relative time-out) and \( \nu_{rel} \) (relative initialization). The process \( \nu_{rel}^p(x) \) is the part of \( x \) that starts to perform actions after a period of time shorter than \( p \). The process \( \nu_{rel}^p(x) \) is the part of \( x \) that starts to perform actions after a period of time longer than or equal to \( p \).

The notation \( \tilde{a} \) for urgent actions in case of relative timing was also used in ACPst [5], the theory from which BPA"! originates.

#### Signature of BPA"!

The signature of BPA"! consists of the urgent action constants \( \tilde{a} : \to P' \) (for each \( a \in A \)), the urgent deadlock constant \( \tilde{\delta} : \to P' \), the immediate deadlock constant \( \delta : \to P' \), the alternative composition operator \( + : P' \times P' \to P' \), the sequential composition operator \( \cdot : P' \times P' \to P' \), the relative delay operator \( \sigma_{rel} : \mathbb{R}_{\geq 0} \times P' \to P' \), the relative time-out operator \( \nu_{rel} : \mathbb{R}_{\geq 0} \times P' \to P' \), and the relative initialization operator \( \nu_{rel} : \mathbb{R}_{\geq 0} \times P' \to P' \).
Axioms of BPA\textsuperscript{sr} The axiom system of BPA\textsuperscript{sr} consists of the equations given in Tables 1 and 15.

\[
\begin{align*}
\sigma^0_{\text{rel}}(x) &= x & \text{SRT1} \\
\sigma^p_{\text{rel}}(\sigma^q_{\text{rel}}(x)) &= \sigma^p_{\text{rel}}(x) & \text{SRT2} \\
\sigma^p_{\text{rel}}(x) + \sigma^p_{\text{rel}}(y) &= \sigma^p_{\text{rel}}(x + y) & \text{SRT3} \\
\sigma^p_{\text{rel}}(x) \cdot y &= \sigma^p_{\text{rel}}(x \cdot y) & \text{SRT4} \\
\alpha + \delta &= \alpha & \text{A6SRa} \\
\sigma^p_{\text{rel}}(x) + \delta &= \sigma^p_{\text{rel}}(x) & \text{A6SRb} \\
\delta \cdot x &= \delta & \text{A7SR} \\
v^p_{\text{rel}}(\delta) &= \delta & \text{SRT00} \\
v^0_{\text{rel}}(\delta) &= \delta & \text{SRT01} \\
v^0_{\text{rel}}(x) &= \delta & \text{SRT02} \\
v^p_{\text{rel}}(\sigma^q_{\text{rel}}(x)) &= \sigma^p_{\text{rel}}(v^q_{\text{rel}}(x)) & \text{SRT03} \\
v^p_{\text{rel}}(x + y) &= v^p_{\text{rel}}(x) + v^p_{\text{rel}}(y) & \text{SRT04} \\
v^p_{\text{rel}}(x \cdot y) &= v^p_{\text{rel}}(x) \cdot y & \text{SRT05} \\
v^0_{\text{rel}}(\delta) &= \sigma^0_{\text{rel}}(\delta) & \text{SRI0} \\
v^0_{\text{rel}}(x) &= x & \text{SRI1} \\
v^0_{\text{rel}}(\delta) &= \sigma^0_{\text{rel}}(\delta) & \text{SRI2} \\
v^p_{\text{rel}}(\sigma^q_{\text{rel}}(x)) &= \sigma^p_{\text{rel}}(v^q_{\text{rel}}(x)) & \text{SRI3} \\
v^p_{\text{rel}}(x + y) &= v^p_{\text{rel}}(x) + v^p_{\text{rel}}(y) & \text{SRI4} \\
v^p_{\text{rel}}(x \cdot y) &= v^p_{\text{rel}}(x) \cdot y & \text{SRI5} \\
\end{align*}
\]

Table 15: Additional axioms for BPA\textsuperscript{sr} \((a \in A, p, q \geq 0, r \geq 0)\)

The axioms of BPA\textsuperscript{sr} are to a large extent simple reformulations of the axioms of BPA\textsuperscript{sat}. That is, constants \(\bar{a} (a \in A)\) have been replaced by constants \(\bar{a}\), and the operators \(\sigma_{\text{abs}}, v_{\text{abs}}\) and \(\overline{v}_{\text{abs}}\) have been replaced by \(\sigma_{\text{rel}}, v_{\text{rel}}\) and \(\overline{v}_{\text{rel}}\), respectively. Striking is the replacement of the axioms SAT4, SAT5 and SAT6 by the simple axiom SRT4. This axiom reflects that timing is relative to the most recent execution of an action. Axioms SRI0-SRI5 are reformulations, in the above-mentioned way, of alternative axioms for axioms SA10-SA15 – which, unlike axioms SA10-SA15, do not accommodate the addition of initial abstraction (see also Section 2.1).

Similar to the case of BPA\textsuperscript{sat}, we can prove that the auxiliary operators \(v_{\text{rel}}\) and \(\overline{v}_{\text{rel}}\), as well as sequential compositions in which the form of the first operand is not \(\bar{a}\) \((a \in A)\) and alternative compositions in which the form of the first operand is \(\sigma^p_{\text{rel}}(t)\), can be eliminated in closed terms of BPA\textsuperscript{sr}. The terms that remain after exhaustive elimination are called the basic terms over BPA\textsuperscript{sr}. Because of this elimination result, we are permitted to use induction on the structure of basic terms over BPA\textsuperscript{sr} to prove statements for all closed terms of BPA\textsuperscript{sr}.

\textbf{Examples} We give some examples of a closed term of BPA\textsuperscript{sr} and the corresponding basic term:

\[
\begin{align*}
\sigma^p_{\text{rel}}(\overline{a} \cdot \sigma^q_{\text{rel}}(\overline{b})) &= \sigma^p_{\text{rel}}(\overline{a} \cdot \sigma^q_{\text{rel}}(\overline{b})) \\
\sigma^p_{\text{rel}}(\overline{a} \cdot (\sigma^q_{\text{rel}}(\overline{b}) + \sigma^q_{\text{rel}}(\overline{c}))) &= \sigma^q_{\text{rel}}(\overline{a} \cdot \sigma^q_{\text{rel}}(\overline{b} + \sigma^q_{\text{rel}}(\overline{c}))) \\
v^p_{\text{rel}}(\sigma^q_{\text{rel}}(\overline{a}) + \sigma^q_{\text{rel}}(\overline{b})) &= \sigma^q_{\text{rel}}(\overline{a} + \sigma^q_{\text{rel}}(\overline{b})) \\
v^p_{\text{rel}}(\sigma^q_{\text{rel}}(\overline{a}) + \sigma^q_{\text{rel}}(\overline{b})) &= \sigma^q_{\text{rel}}(\overline{a}) \\
\end{align*}
\]
Semantics of $\text{BPA}^{\text{rst}}$ In case of relative timing, we can use a simple kind of real time transition system. A real time transition system with relative timing over $A$, consists of a set of states $S$, a root state $\rho \in S$ and four kinds of relations on states:

- A binary relation $\_ \xrightarrow{a} \_ \text{ for each } a \in A$,
- A unary relation $\_ \xrightarrow{\cd} \_ \text{ for each } a \in A$,
- A binary relation $\_ \xrightarrow{r} \_ \text{ for each } r \in \mathbb{R}_{>0}$,
- A unary relation $\text{ID}(\_)$;

satisfying

1. if $s \xrightarrow{r+r'} s'$, $r, r' > 0$, then there is a $s''$ such that $s \xrightarrow{r} s''$ and $s'' \xrightarrow{r'} s'$;
2. if $s \xrightarrow{r} s''$ and $s'' \xrightarrow{r'} s'$, then $s \xrightarrow{r+r'} s'$.

We write $\text{RTTS}'(A)$ for the set of all real time transition systems with relative timing over $A$.

We shall associate a transition system $\text{TS}'(t)$ in $\text{RTTS}'(A)$ with a closed term $t$ of $\text{BPA}^{\text{rst}}$ like before in the case of absolute timing. In case of relative timing, the action step, action termination, time step and immediate deadlock relations can be explained as follows:

- $t \xrightarrow{a} t'$: process $t$ is capable of first performing action $a$ without the least delay and then proceeding as process $t'$;
- $t \xrightarrow{\cd}$: process $t$ is capable of first performing action $a$ without the least delay and then terminating successfully;
- $t \xrightarrow{r} t'$: process $t$ is capable of first idling for a time period $r$ and then proceeding as process $t'$;
- $\text{ID}(t)$: process $t$ is not capable of reaching the present time.

A structured operational semantics of $\text{BPA}^{\text{rst}}$ is described by the rules given in Table 16. In one of the rules for the alternative composition operator, a negative formula of the form $t \xrightarrow{r}$ is used as a premise. A negative formula $t \xrightarrow{r}$ means that for all closed terms $t'$ of $\text{BPA}^{\text{rst}}$ not $t \xrightarrow{r} t'$. Hence, $t \xrightarrow{r}$ is to be read as "process $t$ is not capable of idling for a time period $r"$.

Clearly, changing from absolute timing to relative timing leads to a significant simplification of the operational semantics. However, note that there are two rules now for the alternative composition operator concerning time related capabilities of a process $x + y$. These rules have complementary premises. Together they enforce that the choice between two idling processes is postponed till at least one of the processes cannot idle any longer.

Also the notion of bisimulation becomes simpler in case of relative timing. A bisimulation on $\text{RTTS}'(A)$ is a symmetric binary relation $R$ on the set of states $S$ such that:

1. if $R(s, t)$ and $s \xrightarrow{a} s'$, then there is a $t'$ such that $t \xrightarrow{a} t'$ and $R(s', t')$;
2. if $R(s, t)$, then $s \xrightarrow{\cd}$ iff $t \xrightarrow{\cd}$;
3. if $R(s, t)$ and $s \xrightarrow{r} s'$, then there is a $t'$ such that $t \xrightarrow{r} t'$ and $R(s', t')$;
4. if $R(s, t)$, then $\text{ID}(s)$ iff $\text{ID}(t)$.

As in the case of absolute timing, we obtain a model for $\text{BPA}^{\text{rst}}$ by identifying bisimilar processes.
Table 16: Rules for operational semantics of $\text{BPA}^\text{sr}$ ($a \in A, r > 0, p \geq 0$)

4.2 Algebra of communicating processes

In $\text{ACP}^\text{sr}$, we have a relative counterpart of the absolute urgent initialization operator: $\nu_{\text{rel}}$ (relative urgent initialization). The process $\nu_{\text{rel}}(x)$ is the part of process $x$ that starts to perform actions without any delay. Like before in the case of absolute timing, we use the relative urgent initialization operator to axiomatize the parallel composition operator.
The signature of ACP\textsuperscript{r}t is the signature of BPA\textsuperscript{r}t extended with the parallel composition operator \( \parallel: P' \times P' \to P' \), the left merge operator \( \llcorner: P' \times P' \to P' \), the communication merge operator \( \lceil: P' \times P' \to P' \), the encapsulation operators \( \partial_H: P' \to P' \) (for each \( H \subseteq A \)), and the relative urgent initialization operator \( \nu_{rel}: P' \to P' \).

The axiom system of ACP\textsuperscript{r}t consists of the axioms of BPA\textsuperscript{r}t and the equations given in Table 17.

The additional axioms of ACP\textsuperscript{r}t are just simple reformulations of the additional axioms of ACP\textsuperscript{r}. That is, constants \( \bar{a} (a \in A_\delta) \) have been replaced by constants \( \bar{a} \), and the operators \( \sigma_{abs}, \nu_{abs} \) and \( \nu_{abs} \) have been replaced by \( \sigma_{rel}, \nu_{rel} \) and \( \nu_{rel} \), respectively.

### Table 17: Additional axioms for ACP\textsuperscript{r}t \((a, b \in A_\delta, c \in A, p \geq 0, r > 0)\)

<table>
<thead>
<tr>
<th>Axiom</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \bar{a} \circ \bar{b} = \bar{c} ) if ( \gamma(a, b) = c )</td>
<td>CF1SR</td>
</tr>
<tr>
<td>( \bar{a} \circ \bar{b} = \bar{b} ) if ( \gamma(a, b) ) undefined</td>
<td>CF2SR</td>
</tr>
<tr>
<td>( a \parallel y = (x \parallel y + y \parallel x) + x \parallel y )</td>
<td>CM1</td>
</tr>
<tr>
<td>( \delta \parallel x = \bar{\delta} )</td>
<td>CMID1</td>
</tr>
<tr>
<td>( a \parallel (x + \bar{\delta}) = \bar{a} \cdot (x + \bar{\delta}) )</td>
<td>CM2SRID</td>
</tr>
<tr>
<td>( \bar{a} \cdot x \parallel (y + \bar{\delta}) = \bar{a} \cdot (y \parallel (y + \bar{\delta})) )</td>
<td>CM3SRID</td>
</tr>
<tr>
<td>( \sigma_{rel}(x) \parallel (\nu_{rel}(y) + \bar{\delta}) = \bar{\delta} )</td>
<td>SRCM1ID</td>
</tr>
<tr>
<td>( \sigma_{rel}(x) \parallel (\nu_{rel}(y) + \bar{\delta}) = \sigma_{rel}(x \parallel z) )</td>
<td>SRCM2ID</td>
</tr>
<tr>
<td>( (x + y) \parallel z = x \parallel z + y \parallel z )</td>
<td>CM4</td>
</tr>
<tr>
<td>( \delta \parallel x = \bar{\delta} )</td>
<td>CMID3</td>
</tr>
<tr>
<td>( a \parallel \bar{\delta} = \bar{\delta} )</td>
<td>CMID4</td>
</tr>
<tr>
<td>( \bar{a} \cdot x \parallel \bar{b} = (\bar{a} \cdot \bar{b}) \cdot x )</td>
<td>CM5SR</td>
</tr>
<tr>
<td>( \bar{a} \cdot \bar{b} \cdot x = (\bar{a} \cdot \bar{b}) \cdot x )</td>
<td>CM6SR</td>
</tr>
<tr>
<td>( \bar{a} \cdot x \parallel \bar{b} \cdot y = (\bar{a} \cdot \bar{b}) \cdot (x \parallel y) )</td>
<td>CM7SR</td>
</tr>
<tr>
<td>( (\nu_{rel}(x) + \bar{\delta}) \parallel \sigma_{rel}(y) = \bar{\delta} )</td>
<td>SRCM3ID</td>
</tr>
<tr>
<td>( \sigma_{rel}(x) \parallel (\nu_{rel}(y) + \bar{\delta}) = \bar{\delta} )</td>
<td>SRCM4ID</td>
</tr>
<tr>
<td>( \sigma_{rel}(x) \parallel \sigma_{rel}(y) = \sigma_{rel}(x \parallel y) )</td>
<td>SRCM5</td>
</tr>
<tr>
<td>( (x + y) \parallel z = x \parallel z + y \parallel z )</td>
<td>CM8</td>
</tr>
<tr>
<td>( x \parallel (y + z) = x \parallel y + x \parallel z )</td>
<td>CM9</td>
</tr>
<tr>
<td>( \partial_H(\bar{\delta}) = \bar{\delta} )</td>
<td>D0</td>
</tr>
<tr>
<td>( \partial_H(\bar{a}) = \bar{a} ) if ( a \notin H )</td>
<td>D1SR</td>
</tr>
<tr>
<td>( \partial_H(\bar{a}) = \bar{\delta} ) if ( a \in H )</td>
<td>D2SR</td>
</tr>
<tr>
<td>( \partial_H(\sigma_{rel}(x)) = \sigma_{rel}(\partial_H(x)) )</td>
<td>SRD</td>
</tr>
<tr>
<td>( \partial_H(x + y) = \partial_H(x) + \partial_H(y) )</td>
<td>D3</td>
</tr>
<tr>
<td>( \partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y) )</td>
<td>D4</td>
</tr>
<tr>
<td>( \nu_{rel}(\bar{\delta}) = \bar{\delta} )</td>
<td>SRU0</td>
</tr>
<tr>
<td>( \nu_{rel}(\bar{a}) = \bar{a} )</td>
<td>SRU1</td>
</tr>
<tr>
<td>( \nu_{rel}(\sigma_{rel}(x)) = \bar{\delta} )</td>
<td>SRU2</td>
</tr>
<tr>
<td>( \nu_{rel}(x + y) = \nu_{rel}(x) + \nu_{rel}(y) )</td>
<td>SRU3</td>
</tr>
<tr>
<td>( \nu_{rel}(x \cdot y) = \nu_{rel}(x) \cdot \nu_{rel}(y) )</td>
<td>SRU4</td>
</tr>
</tbody>
</table>
Similar to the case of ACP\textsuperscript{sat}, we can prove that the operators $\|$, $\ll$ and $\partial_H$ and $\nu_{\text{rel}}$ can be eliminated in closed terms of ACP\textsuperscript{sat}. Because of the elimination result for BPA\textsuperscript{sat}, we are permitted to use induction on the structure of basic terms over BPA\textsuperscript{sat} to prove statements for all closed terms of ACP\textsuperscript{sat}.

**Examples** We give some examples of a closed term of ACP\textsuperscript{sat} and the corresponding basic term (in case $\gamma(a, c)$ is undefined):

$$\sigma_{\text{rel}}^5(\tilde{a}) \| \sigma_{\text{rel}}^0(\tilde{b}) \cdot \sigma_{\text{rel}}^0(\tilde{c}) = \sigma_{\text{rel}}^5(\tilde{a} \cdot \sigma_{\text{rel}}^0(\tilde{b} \cdot \sigma_{\text{rel}}^0(\tilde{c})))$$

$$\sigma_{\text{rel}}^5(\tilde{a}) \| \sigma_{\text{rel}}^0(\tilde{b}) \cdot \sigma_{\text{rel}}^0(\tilde{c}) = \sigma_{\text{rel}}^5(\tilde{b} \cdot \sigma_{\text{rel}}^0(\tilde{a} \cdot \sigma_{\text{rel}}^0(\tilde{c})))$$

$$\sigma_{\text{rel}}^5(\tilde{a}) \| \sigma_{\text{rel}}^0(\tilde{b}) \cdot \sigma_{\text{rel}}^0(\tilde{c}) = \sigma_{\text{rel}}^5(\tilde{a} \cdot \sigma_{\text{rel}}^0(\tilde{b} \cdot \sigma_{\text{rel}}^0(\tilde{c})))$$

**Semantics of ACP\textsuperscript{sat** The structured operational semantics of ACP\textsuperscript{sat} is described by the rules for BPA\textsuperscript{sat} and the rules given in Table 18.

### Table 18: Additional rules for ACP\textsuperscript{sat} $(a, b, c \in A, r > 0)$

Changing from absolute timing to relative timing also leads to a simplification of the additional rules for parallel composition, left merge, etc. As in the previous cases, we obtain a model for ACP\textsuperscript{sat} by identifying bisimilar processes.

### 4.3 Embedding ACP\textsuperscript{sat} in ACP\textsuperscript{sat}$\sqrt{\!}$

Consider two theories $T$ and $T'$. An embedding of $T$ in $T'$ is a term structure preserving injective mapping $\epsilon$ from the terms of $T$ to the terms of $T'$ such that for all closed terms
s, t of T, s = t is derivable in T implies ε(s) = ε(t) is derivable in T'. If there exists an embedding of T in T', we say that T can be embedded in T'. It roughly means that what is expressible in T remains expressible in T' and what is derivable in T remains derivable in T'. The requirement that ε is term structure preserving means that, for all terms t of T with free variables among x₁, ..., xₙ and all closed terms t₁, ..., tₙ of T of appropriate sorts, ε(t|t₁/ x₁, ..., xₙ] = ε(t)[ε(t₁), ..., ε(tₙ)/ x₁, ..., xₙ].

Let f be an operator that is not in the signature of theory T. An explicit definition of f in T is an equation f(x₁, ..., xₙ) = t where t is a term of T that does not contain other free variables than x₁, ..., xₙ. An extension of theory T with constants and operators defined by explicit definitions in T is called a definitional extension of T.

Consider again two theories T and T'. Suppose that the constants and operators in the signature of T that are not in the signature of T' can be defined in T' by explicit definitions. Let T'' be the resulting definitional extension of T'. Suppose further that the axioms of T are derivable for closed terms in T''. Then T can be embedded in T'.

The explicit definitions induce the following embedding:

\[
\begin{align*}
ε(x) &= x \\
ε(f(t₁, ..., tₙ)) &= f(ε(t₁), ..., ε(tₙ)) & \text{if } f \text{ in the signature of } T' \\
\epsilon(f(t₁, ..., tₙ)) &= t|t₁/ x₁, ..., xₙ] & \text{if the explicit definition of } f = f(t₁, ..., xₙ) = t
\end{align*}
\]

In this paper, we will show the existence of embeddings in the way outlined above.

The explicit definitions needed to show that ACPₚᵣᵗ can be embedded in ACPₛᵣᵗ √ are given in Table 19. The following lemma presents an interesting property of processes with relative timing.

**Lemma 4** For each closed term t of ACPₛᵣᵗ √ generated by the embedded constants and operators of ACPₚᵣᵗ, \( \overline{ν}_{abs}(t) = σ_{abs}(t) \).

**Proof.** It is straightforward to prove this by induction on the structure of t. We present only the case that t is of the form \( \overline{a} \cdot t' \). The other cases are similar, but simpler.

\[
\begin{align*}
\overline{a} &= \sqrt{v} \cdot σ_{abs}(\overline{a}) \\
σ_{abs}(x) &= \sqrt{v} \cdot ν_{abs}^{v+}(x) \\
ν_{abs}(x) &= \sqrt{v} \cdot ν_{abs}^{v+}(\overline{ν}_{abs}(x)) \\
ν_{abs}(x) &= \sqrt{v} \cdot σ_{abs}(\overline{ν}_{abs}(x))
\end{align*}
\]

Table 19: Definitions of relative time operators (a ∈ Aₙ)

\[\begin{array}{ll}
\overline{a} &= \sqrt{v} \cdot σ_{abs}(\overline{a}) \\
σ_{abs}(x) &= \sqrt{v} \cdot ν_{abs}^{v+}(x) \\
ν_{abs}(x) &= \sqrt{v} \cdot ν_{abs}^{v+}(\overline{ν}_{abs}(x)) \\
ν_{abs}(x) &= \sqrt{v} \cdot σ_{abs}(\overline{ν}_{abs}(x))
\end{array}\]

Lemma 4 expresses that, for a process with relative timing, absolute initialization of the process at time \( p \) is the same as shifting the process in time from time 0 to time \( p \).
– which implies preceding absolute initialization at time 0. It follows from Lemma 4 and axiom SAI5 that for each pair of closed terms \( t, t' \) of ACP\textsuperscript{sat} generated by the embedded constants and operators of ACP\textsuperscript{sat}, \( \sigma_{\text{abs}}^p(t \cdot t') = \sigma_{\text{abs}}^p(t) \cdot t' \). The condition that \( t \) is a term generated by the embedded constants and operators of ACP\textsuperscript{sat} can not be dropped here. However, Lemma 5 points out that this condition is not a necessary one, since \( \nu_{\text{abs}}(t) \) is not equal to a term generated by the embedded constants and operators of ACP\textsuperscript{sat} – unless \( t = \delta \). Lemma 5 is needed in the proof of Theorem 6.

**Lemma 5** For each pair of closed terms \( t, t' \) of ACP\textsuperscript{sat} generated by the embedded constants and operators of ACP\textsuperscript{sat}, \( \sigma_{\text{abs}}^p(\nu_{\text{abs}}(t) \cdot t') = \sigma_{\text{abs}}^p(\nu_{\text{abs}}(t)) \cdot t' \).

**Proof.** It is straightforward to prove this by induction on the structure of \( t \). The proof is extremely long. We present only the case that \( t \) is of the form \( a_{\rel}(t_{\rel}(t'_{\rel})) \). The other cases are similar, but simpler, and do not require case distinction.

Case \( q = 0 \):
\[
\sigma_{\text{abs}}(\nu_{\text{abs}}(a_{\rel}(t'_{\rel}))) = \sigma_{\text{abs}}(\nu_{\text{abs}}(a_{\rel}(t'_{\rel})))
\]

Case \( q > 0 \):
\[
\sigma_{\text{abs}}(\nu_{\text{abs}}(a_{\rel}(t'_{\rel}))) = \sigma_{\text{abs}}(\nu_{\text{abs}}(a_{\rel}(t'_{\rel})))
\]

The existence of an embedding of ACP\textsuperscript{sat} in ACP\textsuperscript{sat} is established by proving the following theorem.

**Theorem 6 (Embedding ACP\textsuperscript{sat} in ACP\textsuperscript{sat})** For closed terms, the axioms of ACP\textsuperscript{sat} are derivable from the axioms of ACP\textsuperscript{sat} and the explicit definitions of the constants and operators \( a, a_{\rel}, v_{\rel}, v_{\rel}' \) and \( v_{\rel} \) in Table 19.

**Proof.** The proof of this theorem is given in Appendix A.1. The proof is a matter of straightforward calculations. Equations SIA1 (page 19) and DISTR\( \Box \) (page 20), the standard initialization axioms (Table 14, page 21), and Lemmas 4 and 5 (page 27-28) are very useful in the proof.

## 5 Discrete time process algebra

In this section, we present ACP\textsuperscript{dat} and ACP\textsuperscript{drt}, discrete time process algebras with absolute timing and relative timing, respectively. ACP\textsuperscript{dat} and ACP\textsuperscript{drt} are conservative
extensions of $\text{ACP}_{\text{dat}}$ and $\text{ACP}_{\text{drt}}$ [6], respectively. First, in Section 5.1, we present $\text{ACP}_{\text{dat}}$ and $\text{ACP}_{\text{dat}}^\vee$, the extension of $\text{ACP}_{\text{dat}}$ with initial abstraction. After that, in Section 5.2, we present $\text{ACP}_{\text{dat}}^\prime$. Finally, we show in Section 5.3 how $\text{ACP}_{\text{dat}}^\vee$ can be embedded in $\text{ACP}_{\text{dat}}^\prime$.

5.1 Discrete time process algebra: absolute timing

In this subsection, we give the signature, axioms and term model of $\text{ACP}_{\text{dat}}$, a discrete time process algebra with absolute timing. $\text{ACP}_{\text{dat}}$ is a conservative extension of the theory $\text{ACP}_{\text{dat}}$, presented in [6]. Like $\text{ACP}_{\text{drt}}$, it separates execution of actions and passage of time. In $\text{ACP}_{\text{dat}}$, time is measured on a discrete time scale. The discrete time points divide time into time slices and timing of actions is done with respect to the time slices in which they are performed – “in time slice $n+1$” means “at some time point $p$ such that $n \leq p < n+1$”.

First, we treat $\text{BPA}_{\text{dat}}$, basic discrete time process algebra with absolute timing, in which parallelism and communication are not considered. After that, $\text{BPA}_{\text{dat}}$ is extended to $\text{ACP}_{\text{dat}}$ to deal with parallelism and communication as well. Finally, initial abstraction is added.

Basic process algebra

In $\text{BPA}_{\text{dat}}$, we have the constants $\underline{a}$ and $\underline{\delta}$ instead of $\bar{a}$ and $\bar{\delta}$. The constants $\underline{a}$ and $\underline{\delta}$ stand for $a$ in time slice 1 and a deadlock in time slice 1, respectively. The operators $\sigma_{\text{abs}}, \upsilon_{\text{abs}}$ and $\overline{\upsilon}_{\text{abs}}$ have a natural number instead of a positive real number as their first argument. The process $\sigma_{\text{abs}}^n(x)$ is the process $x$ shifted in time by $n$ on the discrete time scale. The process $\upsilon_{\text{abs}}^n(x)$ is the part of $x$ that starts to perform actions before time slice $n+1$. The process $\overline{\upsilon}_{\text{abs}}^n(x)$ is the part of $x$ that starts to perform actions in time slice $n+1$ or a later time slice. Recall that time point $n$ is the starting-point of time slice $n+1$.

In $\text{ACP}_{\text{dat}}$ [6], the notation $\text{fts}(a)$ was used for actions in the first time slice. A discrete time version of $\text{ACP}$ with absolute timing where the notation $\underline{a}$ was used earlier for actions in a time slice is $\text{ACP}_{\text{drt}}$ [2], but there it always carries a time-stamp.

Signature of $\text{BPA}_{\text{dat}}$ The signature of $\text{BPA}_{\text{dat}}$ consists of the undelayable action constants $\underline{a} : \rightarrow P$ (for each $a \in A$), the undelayable deadlock constant $\underline{\delta} : \rightarrow P$, the immediate deadlock constant $\underline{\delta} : \rightarrow P$, the alternative composition operator $\underline{+} : P \times P \rightarrow P$, the sequential composition operator $\underline{\cdot} : P \times P \rightarrow P$, the absolute delay operator $\sigma_{\text{abs}} : \mathbb{N} \times P \rightarrow P$, the absolute time-out operator $\upsilon_{\text{abs}} : \mathbb{N} \times P \rightarrow P$, and the absolute initialization operator $\overline{\upsilon}_{\text{abs}} : \mathbb{N} \times P \rightarrow P$.

We denote elements of $\mathbb{N}$ by $m, m', n, n'$.

Axioms of $\text{BPA}_{\text{dat}}$ The axiom system of $\text{BPA}_{\text{dat}}$ consists of the equations given in Tables 1 and 20.

The axioms of $\text{BPA}_{\text{dat}}$ are to a large extent simple reformulations of the axioms of $\text{BPA}_{\text{dat}}$. That is, constants $\bar{a}$ ($a \in A$) have been replaced by constants $\underline{a}$, and the first argument of the operators $\sigma_{\text{abs}}, \upsilon_{\text{abs}}$ and $\overline{\upsilon}_{\text{abs}}$ has been restricted to elements of...
\[a_0^{abs}(x) = v_0^{abs}(x)\]
\[\sigma_{abs}^{m+n}(x) = \sigma_{abs}^{m+n}(x)\]
\[\sigma_{abs}^{n}(x) + \sigma_{abs}^{n}(y) = \sigma_{abs}^{n}(x + y)\]
\[\sigma_{abs}^{n}(x) \cdot v_{abs}^{n}(y) = \sigma_{abs}^{n}(x \cdot \delta)\]
\[\sigma_{abs}^{n}(x) \cdot (v_{abs}^{n}(y) + \sigma_{abs}^{n}(z)) = \sigma_{abs}^{n}(x \cdot v_{abs}^{n}(z))\]
\[\sigma_{abs}^{n}(\delta) \cdot x = \sigma_{abs}^{n}(\delta)\]
\[\sigma_{abs}^{1}(\delta) = \delta\]
\[\delta + \delta = \delta\]
\[u_{abs}^{n}(\delta) = \delta\]
\[u_{abs}^{n+1}(a) = a\]
\[u_{abs}^{0}(x) = \delta\]
\[u_{abs}^{m+n}(\sigma_{abs}^{n}(x)) = \sigma_{abs}^{n}(u_{abs}^{n}(x))\]
\[u_{abs}^{n}(x + y) = u_{abs}^{n}(x) + u_{abs}^{n}(y)\]
\[u_{abs}^{n}(x \cdot y) = u_{abs}^{n}(x) \cdot y\]
\[\sigma_{abs}^{0}(\delta) = \delta\]
\[v_{abs}^{n+1}(\delta) = \sigma_{abs}^{n+1}(\delta)\]
\[v_{abs}^{0}(\delta) = \rho\]
\[v_{abs}^{n+1}(a) = \sigma_{abs}^{n+1}(\delta)\]
\[v_{abs}^{m+n}(\sigma_{abs}^{m}(x)) = \sigma_{abs}^{m}(u_{abs}^{m}(v_{abs}^{n}(x)))\]
\[v_{abs}^{n}(x + y) = v_{abs}^{n}(x) + v_{abs}^{n}(y)\]
\[v_{abs}^{n}(x \cdot y) = v_{abs}^{n}(x) \cdot y\]

Table 20: Additional axioms for BPA\textsuperscript{dat} (\(a \in A_\delta\))

N. Striking is the new axiom DAT7. This axiom makes the reformulations of axioms A6SAb and A7SA, i.e. \(\sigma_{abs}^{n+1}(x) + \delta = \sigma_{abs}^{n+1}(x)\) and \(\delta \cdot x = \delta\), derivable. Axiom DAT7 expresses that an immediate deadlock shifted in time by 1 is identified with an undelayable deadlock in the first time slice.

Like in the case of BPA\textsuperscript{ant}, we can prove that the auxiliary operators \(u_{abs}\) and \(v_{abs}\), as well as sequential compositions in which the form of the first operand is not \(a\) \((a \in A)\) and alternative compositions in which the form of the first operand is \(\sigma_{abs}(t)\), can be eliminated in closed terms of BPA\textsuperscript{dat}. The terms that remain after exhaustive elimination are called the basic terms over BPA\textsuperscript{dat}. Because of this elimination result, we are permitted to use induction on the structure of basic terms over BPA\textsuperscript{dat} to prove statements for all closed terms of BPA\textsuperscript{dat}.

**Examples** We give some examples of a closed term of BPA\textsuperscript{dat} and the corresponding basic term:

\[\sigma_{abs}^{1}(a) \cdot b = \sigma_{abs}^{1}(a \cdot \delta)\]
\[v_{abs}^{3}(\sigma_{abs}^{2}(a) + \sigma_{abs}^{5}(b)) = \sigma_{abs}^{2}(a)\]
\[v_{abs}^{3}(\sigma_{abs}^{2}(a) + \sigma_{abs}^{5}(b)) = \sigma_{abs}^{5}(b)\]

**Semantics of BPA\textsuperscript{dat}** In case a discrete time scale is used, we use a variant of real time transition systems. A discrete time transition system over \(A\), consists of a set of states \(S\), a root state \(\rho \in S\) and four kinds of relations on states:
a binary relation \( \langle -, n \rangle \xrightarrow{a} \langle -, n \rangle \) for each \( a \in A, n \in \mathbb{N} \),
a unary relation \( \langle -, n \rangle \xrightarrow{\sigma} \langle \sqrt[n]{}, n \rangle \) for each \( a \in A, n \in \mathbb{N} \),
a binary relation \( \langle -, n \rangle \xrightarrow{m} \langle -, n' \rangle \) for each \( m \in \mathbb{N}_{>0}, n, n' \in \mathbb{N} \) where \( n' = n + m \),
a unary relation \( \text{ID}(\langle -, n \rangle) \) for each \( n \in \mathbb{N} \); satisfying

1. if \( \langle s, n \rangle \xrightarrow{m+m'} \langle s', n' \rangle \), \( m, m' > 0 \), then there is a \( s'' \) such that \( \langle s, n \rangle \xrightarrow{m} \langle s'', n + m \rangle \) and \( \langle s'', n + m \rangle \xrightarrow{m'} \langle s', n' \rangle \);
2. if \( \langle s, n \rangle \xrightarrow{m} \langle s'', n + m \rangle \) and \( \langle s'', n + m \rangle \xrightarrow{m'} \langle s', n' \rangle \), then \( \langle s, n \rangle \xrightarrow{m+m'} \langle s', n' \rangle \).

We write \( \text{DTTS}(A) \) for the set of all discrete time transition systems. Associating a transition system in \( \text{DTTS}(A) \) with a closed term \( t \) of \( \text{BPA}_{\text{dat}} \) proceeds in essentially the same way as associating a transition system in \( \text{RTTS}(A) \) with a closed term \( t \) of \( \text{BPA}_{\text{sat}} \). The only difference is that in the rules for the operational semantics of \( \text{BPA}_{\text{dat}} \) all numbers involved are restricted to \( \mathbb{N} \). Therefore, we refrain from giving the rules.

Bisimulation on \( \text{DTTS}(A) \) is defined as on \( \text{RTTS}(A) \). As in the real time cases, we obtain a model for \( \text{BPA}_{\text{dat}} \) by identifying bisimilar processes.

**Algebra of communicating processes**

In \( \text{ACP}_{\text{dat}} \), we do not have a discrete time counterpart of \( \nu_{\text{abs}} \). Unlike before in the case of real time, we can use \( v_{\text{abs}}^1 \) instead.

**Signature of \( \text{ACP}_{\text{dat}} \)**
The signature of \( \text{ACP}_{\text{dat}} \) is the signature of \( \text{BPA}_{\text{dat}} \) extended with the **parallel composition** operator \( \parallel: \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P} \), the **left merge** operator \( |: \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P} \), the **communication merge** operator \( \vdash: \mathcal{P} \times \mathcal{P} \rightarrow \mathcal{P} \), and the **encapsulation** operators \( \partial_H: \mathcal{P} \rightarrow \mathcal{P} \) (for each \( H \subseteq A \)).

**Axioms of \( \text{ACP}_{\text{dat}} \)** The axiom system of \( \text{ACP}_{\text{dat}} \) consists of the axioms of \( \text{BPA}_{\text{dat}} \) and the equations given in Table 21.

The additional axioms of \( \text{ACP}_{\text{dat}} \) are to a large extent simple reformulations of the additional axioms of \( \text{ACP}_{\text{sat}} \). That is, constants \( \bar{a} (a \in A_0) \) have been replaced by constants \( a \), the first argument of the operators \( \sigma_{\text{abs}} \) and \( v_{\text{abs}} \) has been restricted to \( \mathbb{N} \), and the operator \( \nu_{\text{abs}} \) has been replaced by \( v_{\text{abs}}^1 \). A counterpart of \( \text{SACM1} \) is missing. However, axiom \( \text{DAT7} \) makes the simple reformulation of axiom \( \text{SACM1} \), i.e. \( \sigma_{\text{abs}}^n(x) \parallel (v_{\text{abs}}^1(y) + \delta) = \delta \), derivable.

As in the case of \( \text{ACP}_{\text{sat}} \), we can prove that the operators \( \parallel, |, \vdash \) and \( \partial_H \) can be eliminated in closed terms of \( \text{ACP}_{\text{dat}} \). Because of the elimination result for \( \text{BPA}_{\text{dat}} \), we are permitted to use induction on the structure of basic terms over \( \text{BPA}_{\text{dat}} \) to prove statements for all closed terms of \( \text{ACP}_{\text{dat}} \).

**Examples** We give some examples of a closed term of \( \text{ACP}_{\text{dat}} \) and the corresponding basic term (in case \( \gamma(a, b) \) is undefined):

\[
\begin{align*}
\sigma_{\text{abs}}^3(a) \cdot (\sigma_{\text{abs}}^3(b) \parallel \sigma_{\text{abs}}^3(c)) &= \sigma_{\text{abs}}^3(a \cdot \delta) \\
\sigma_{\text{abs}}^3(a) \cdot \sigma_{\text{abs}}^3(b) \parallel \sigma_{\text{abs}}^2(c) &= \sigma_{\text{abs}}^3(c \cdot \sigma_{\text{abs}}^3(a \cdot \sigma_{\text{abs}}^2(b))) \\
(a \cdot \sigma_{\text{abs}}^3(b)) \parallel (b \cdot \sigma_{\text{abs}}^2(c)) &= a \cdot (b \cdot \sigma_{\text{abs}}^2(a \cdot b \cdot \delta \cdot a)) + b \cdot (a \cdot \sigma_{\text{abs}}^3(a \cdot b \cdot b \cdot a))
\end{align*}
\]
Table 21: Additional axioms for ACP_{\text{dat}} (a, b \in A_\delta, c \in A)

\begin{tabular}{l|l}
\hline
$a \mid b = c$ & CF1DA \\
$a \mid b = \delta$ & CF2DA \\
$x \parallel y = (x \parallel y + y \parallel x) + x \mid y$ & CM1 \\
$\delta \parallel x = \delta$ & CMID1 \\
$x \parallel \delta = \delta$ & CMID2 \\
$a \parallel (x + \delta) = a \cdot (x + \delta)$ & CM2DA \\
$a \cdot x \parallel (y + \delta) = a \cdot (x \parallel (y + \delta))$ & CM3DA \\
$\sigma_{\text{abs}}^1(x) \mid (\sigma_{\text{abs}}^1(y) + \sigma_{\text{abs}}^1(z)) = \sigma_{\text{abs}}^1(x \parallel z)$ & D ACM2 \\
$(x + y) \parallel z = x \parallel z + y \parallel z$ & CM4 \\
$\delta \mid x = \delta$ & CMID3 \\
$x \mid \delta = \delta$ & CMID4 \\
$a \cdot x \mid b = (a \mid b) \cdot x$ & CM5DA \\
$a \mid b \cdot x = (a \mid b) \cdot x$ & CM6DA \\
$a \cdot x \mid b \cdot y = (a \mid b) \cdot (x \parallel y)$ & CM7DA \\
$(v_{\text{abs}}^1(x) + \delta) \mid \sigma_{\text{abs}}^{n+1}(y) = \delta$ & DACM3 \\
$\sigma_{\text{abs}}^{n+1}(x) \mid (v_{\text{abs}}^1(y) + \delta) = \delta$ & DACM4 \\
$\sigma_{\text{abs}}^n(x) \mid \sigma_{\text{abs}}^n(y) = \sigma_{\text{abs}}^n(x \parallel y)$ & DACM5 \\
$(x + y) \mid z = x \parallel z + y \parallel z$ & CM8 \\
$x \mid (y + z) = x \parallel y + x \parallel z$ & CM9 \\
$\partial_H(\delta) = \delta$ & D0 \\
$\partial_H(a) = a$ if $a \notin H$ & D1DA \\
$\partial_H(a) = \delta$ if $a \in H$ & D2DA \\
$\partial_H(\sigma_{\text{abs}}^n(x)) = \sigma_{\text{abs}}^n(\partial_H(x))$ & DAD \\
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$ & D3 \\
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$ & D4 \\
\hline
\end{tabular}

Semantics of ACP_{\text{dat}} Like for the rules for the operational semantics of BPA_{\text{dat}}, the additional rules for ACP_{\text{dat}} differ from the corresponding rules for the real time case only in that all numbers involved are restricted to \( \mathbb{N} \). Therefore, we refrain again from giving the rules.

Again, we obtain a model for ACP_{\text{dat}} by identifying bisimilar processes.

Initial abstraction

We add the initial abstraction operator \( \sqrt{\delta} \) to ACP_{\text{dat}}. This operator is the discrete counterpart of \( \sqrt{\delta} \). This means that \( \sqrt{\delta} i.F \), where \( i \) is a variable ranging over \( \mathbb{N} \) and \( F \) is a term that may contain free variables, denotes a function \( f : \mathbb{N} \rightarrow P \) that satisfies \( f(n) = \overline{\sigma_{\text{abs}}^n(f(n))} \) for all \( n \in \mathbb{N} \). In the resulting theory, called ACP_{\text{dat}}\sqrt{\delta}, the sort \( P \) of processes is replaced by the sort \( P^* \) of parametric time processes.

Signature of ACP_{\text{dat}}\sqrt{\delta} The signature of ACP_{\text{dat}}\sqrt{\delta} is the signature of ACP_{\text{dat}} extended with the initial abstraction (variable-binding) operator \( \sqrt{\delta} : \mathbb{N} \cdot P^* \rightarrow P^* \).

We assume that an infinite set of variables ranging over \( \mathbb{N} \) has been given, and denote them by \( i, j, \ldots \). We denote terms of ACP_{\text{dat}}\sqrt{\delta} by \( F, G, \ldots \).
Axiom system of \( \text{ACP}^{\text{dat}} \) The axiom system of \( \text{ACP}^{\text{dat}} \) consists of the axioms of \( \text{ACP}^{\text{dat}} \) and the equations given in Table 22.

\[
\begin{align*}
\sqrt{\sigma} j \cdot G &= \sqrt{\sigma} i \cdot G[i/j] & \text{DIA1} \\
\overline{\sigma}_{\text{abs}}(\sqrt{\sigma} i \cdot F) &= \overline{\sigma}_{\text{abs}}(F[n/i]) & \text{DIA2} \\
\sqrt{\sigma} i \cdot (\sqrt{\sigma} j \cdot F) &= \sqrt{\sigma} i \cdot F[i/j] & \text{DIA3} \\
G &= \sqrt{\sigma} i \cdot G & \text{DIA4} \\
(\forall n \in \mathbb{N} \bullet \overline{\sigma}_{\text{abs}}(x) = \overline{\sigma}_{\text{abs}}(y)) & \Rightarrow x = y & \text{DIA5} \\
\sigma_{\text{abs}}(a) \cdot x &= \sigma_{\text{abs}}(a) \cdot \overline{\sigma}_{\text{abs}}(x) & \text{DIA6} \\
\sigma_{\text{abs}}(\sqrt{\sigma} i \cdot F) &= \sigma_{\text{abs}}(F[0/i]) & \text{DIA7} \\
(\sqrt{\sigma} i \cdot F) + G &= \sqrt{\sigma} i \cdot (F + \overline{\sigma}_{\text{abs}}(G)) & \text{DIA8} \\
(\sqrt{\sigma} i \cdot F) \cdot G &= \sqrt{\sigma} i \cdot (F \cdot G) & \text{DIA9} \\
\overline{\sigma}_{\text{abs}}(\sqrt{\sigma} i \cdot F) &= \sqrt{\sigma} i \cdot \overline{\sigma}_{\text{abs}}(F) & \text{if } n \neq i & \text{DIA10} \\
(\sqrt{\sigma} i \cdot F) \parallel G &= \sqrt{\sigma} i \cdot (F \parallel \overline{\sigma}_{\text{abs}}(G)) & \text{DIA11} \\
G \parallel (\sqrt{\sigma} i \cdot F) &= \sqrt{\sigma} i \cdot (\overline{\sigma}_{\text{abs}}(G) \parallel F) & \text{DIA12} \\
(\sqrt{\sigma} i \cdot F) \mid G &= \sqrt{\sigma} i \cdot (F \mid \overline{\sigma}_{\text{abs}}(G)) & \text{DIA13} \\
G \mid (\sqrt{\sigma} i \cdot F) &= \sqrt{\sigma} i \cdot (\overline{\sigma}_{\text{abs}}(G) \mid F) & \text{DIA14} \\
\partial_H(\sqrt{\sigma} i \cdot F) &= \sqrt{\sigma} i \cdot \partial_H(F) & \text{DIA15}
\end{align*}
\]

Table 22: Axioms for discrete initial abstraction (\( i \) not free in \( G \))

The axioms for discrete initial abstraction are simple reformulations of the axioms for standard initial abstraction. That is, the operator \( \sqrt{s} \) has been replaced by \( \sqrt{d} \), and the variables ranging over \( \mathbb{R}_{\geq 0} \) have been replaced by variables ranging over \( \mathbb{N} \).

As in the case of \( \text{ACP}^{\text{dat}} \), all closed terms of \( \text{ACP}^{\text{dat}} \) can be written in the form \( \sqrt{d} i \cdot F \) where \( F \) is a basic term over \( \text{BPA}^{\text{dat}} \).

**Examples** We give some examples of a closed term of \( \text{ACP}^{\text{dat}} \), the corresponding term of the form \( \sqrt{d} i \cdot F \) where \( F \) is a basic term and, if possible, the corresponding basic term without initial abstraction:

\[
\begin{align*}
\sigma_{\text{abs}}^1(\sqrt{d} i \cdot \sigma_{\text{abs}}^1(a)) &= \sqrt{d} i \cdot \sigma_{\text{abs}}^1(a) \\
\overline{\sigma}_{\text{abs}}(\sqrt{d} i \cdot \sigma_{\text{abs}}^1(a)) &= \sqrt{d} i \cdot \sigma_{\text{abs}}^1(a) \\
\sqrt{d} i \cdot (\sqrt{d} j \cdot \sigma_{\text{abs}}^1(a)) &= \sqrt{d} i \cdot \sigma_{\text{abs}}^1(a)
\end{align*}
\]

**Semantics of \( \text{ACP}^{\text{dat}} \)** We have to extend \( \text{DTTS}(A) \) to the function space

\[
\text{DTTS}^*(A) = \{ f : \mathbb{N} \rightarrow \text{DTTS}(A) \mid \forall n \in \mathbb{N} \bullet f(n) = \overline{\sigma}_{\text{abs}}(f(n)) \}
\]

The constants and operators of \( \text{ACP}^{\text{dat}} \) can be defined on \( \text{DTTS}^*(A) \) in the same way as for the real time case.

We say that \( f, g \in \text{DTTS}^*(A) \) are bisimilar if for all \( n \in \mathbb{N} \), there exists a bisimulation \( R \) such that \( R(f(n), g(n)) \). Like before, we obtain a model of \( \text{ACP}^{\text{dat}} \) by defining all operators on the set of bisimulation equivalence classes.
5.2 Discrete time process algebra: relative timing

In this subsection, we give the signature, axioms and term model of $ACP^drt$, a discrete time process algebra with relative timing. $ACP^drt$ is a conservative extension of the theory $ACP_drt$, presented in [6]. Like $ACP_drt$, it separates execution of actions and passage of time.

First, we treat $BPA^{drt}$, basic discrete time process algebra with relative timing, in which parallelism and communication are not considered. After that, $BPA^{drt}$ is extended to $ACP^{drt}$ to deal with parallelism and communication as well.

Basic process algebra

In $BPA^{drt}$, we have the constants $a$ and $\hat{a}$ instead of $a$ and $\hat{a}$, and the operator $\sigma_{rel}$ instead of $\sigma_{abs}$. The constants $a$ and $\hat{a}$ stand for $a$ in the current time slice and a deadlock in the current time slice, respectively. The process $\sigma_{rel}^n(x)$ is the process $x$ delayed for a period of time $n$ on the discrete time scale, i.e. till the $n$-th next time slice. We have relative counterparts of the absolute time-out and initialization operators as well: $v_{rel}$ and $\overline{v}_{rel}$. The process $v_{rel}^n(x)$ is the part of $x$ that starts to perform actions before the $n$-th next time slice. The process $\overline{v}_{rel}^n(x)$ is the part of $x$ that starts to perform actions in the $n$-th next time slice or a later time slice. As in Section 4, we use $P'$ for the sort of relative time processes.

In some presentations of $ACP^{drt}$, including [6], the notation $cts(a)$ was used instead of $a$. The notation $a$ for actions in the current time slice was first used in $ACP^{drt}$ [2].

Signature of $BPA^{drt}$ The signature of $BPA^{drt}$ consists of the undelayable action constants $a : \to P'$ (for each $a \in A$), the undelayable deadlock constant $\hat{a} : \to P'$, the immediate deadlock constant $\hat{\sigma} : \to P'$, the alternative composition operator $+: P' \times P' \to P'$, the sequential composition operator $\cdot: P' \times P' \to P'$, the relative delay operator $\sigma_{rel}: N \times P' \to P'$, the relative time-out operator $v_{rel}: N \times P' \to P'$, and the relative initialization operator $\overline{v}_{rel}: N \times P' \to P'$.

Axioms of $BPA^{drt}$ The axiom system of $BPA^{drt}$ consists of the equations given in Tables 1 and 23.

The axioms of $BPA^{drt}$ are to a large extent simple reformulations of the axioms of $BPA^{dat}$. That is, constants $a$ ($a \in A_d$) have been replaced by constants $a$, and the operators $\sigma_{abs}$, $v_{abs}$ and $\overline{v}_{abs}$ have been replaced by $\sigma_{rel}$, $v_{rel}$ and $\overline{v}_{rel}$, respectively. The replacement of the axioms DAT4, DAT5 and DAT6 by the simple axiom DRT4 as well as the replacement of the axioms DA10-DA15 by the axioms DRI0-DRI5 are strongly reminiscent of the real time case.

Similar to the case of $BPA^{dat}$, we can prove that the auxiliary operators $v_{rel}$ and $\overline{v}_{rel}$, as well as sequential compositions in which the form of the first operand is not $a$ ($a \in A$) and alternative compositions in which the form of the first operand is $\sigma_{rel}^n(t)$, can be eliminated in closed terms of $BPA^{drt}$. The terms that remain after exhaustive elimination are called the basic terms over $BPA^{drt}$.

Semantics of $BPA^{drt}$ In case of relative timing, we can use a simple kind of discrete time transition system. A discrete time transition system with relative timing over $A$, consists of a set of states $S$, a root state $\rho \in S$ and four kinds of relations on states:
\[
\begin{align*}
\sigma^0_{\text{rel}}(x) &= x & \text{DRT1} \\
\sigma^m_{\text{rel}}(\sigma^n_{\text{rel}}(x)) &= \sigma^{m+n}_{\text{rel}}(x) & \text{DRT2} \\
\sigma^n_{\text{rel}}(x) + \sigma^n_{\text{rel}}(y) &= \sigma^n_{\text{rel}}(x + y) & \text{DRT3} \\
\sigma^n_{\text{rel}}(x) \cdot y &= \sigma^n_{\text{rel}}(x \cdot y) & \text{DRT4} \\
\sigma^1_{\text{rel}}(\delta) &= \delta & \text{DRT7} \\
\alpha + \delta &= \alpha & \text{A6DRa} \\
\nu^0_{\text{rel}}(\delta) &= \delta & \text{DRTO0} \\
\nu^{n+1}_{\text{rel}}(\alpha) &= \alpha & \text{DRTO1} \\
\nu^0_{\text{rel}}(x) &= \delta & \text{DRTO2} \\
\nu^{m+n}_{\text{rel}}(\sigma^m_{\text{rel}}(x)) &= \sigma_{\text{rel}}^m(\nu^m_{\text{rel}}(x)) & \text{DRT03} \\
\nu^m_{\text{rel}}(x + y) &= \nu^m_{\text{rel}}(x) + \nu^m_{\text{rel}}(y) & \text{DRT04} \\
\nu^m_{\text{rel}}(x \cdot y) &= \nu^m_{\text{rel}}(x) \cdot y & \text{DRT05} \\
\overline{\nu}^0_{\text{rel}}(\delta) &= \sigma^0_{\text{rel}}(\delta) & \text{DRI0} \\
\overline{\nu}^0_{\text{rel}}(x) &= x & \text{DRI1} \\
\overline{\nu}^{n+1}_{\text{rel}}(\alpha) &= \sigma^n_{\text{rel}}(\delta) & \text{DRI2} \\
\overline{\nu}^{m+n}_{\text{rel}}(\sigma^m_{\text{rel}}(x)) &= \sigma_{\text{rel}}^m(\overline{\nu}^m_{\text{rel}}(x)) & \text{DRI3} \\
\overline{\nu}^m_{\text{rel}}(x + y) &= \nu^m_{\text{rel}}(x) + \nu^m_{\text{rel}}(y) & \text{DRI4} \\
\overline{\nu}^m_{\text{rel}}(x \cdot y) &= \overline{\nu}^m_{\text{rel}}(x) \cdot y & \text{DRI5} \\
\end{align*}
\]

Table 23: Additional axioms for BPA^{drt} (a ∈ A_δ)

<table>
<thead>
<tr>
<th>Condition</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>a binary relation ( \overset{a}{\rightarrow} ) for each ( a \in A ),</td>
<td></td>
</tr>
<tr>
<td>a unary relation ( \overset{a}{\rightarrow} \sqrt{\delta} ) for each ( a \in A ),</td>
<td></td>
</tr>
<tr>
<td>a binary relation ( \overset{1}{\rightarrow} \overset{n}{\rightarrow} ) for each ( n \in \mathbb{N}_{&gt;0} ),</td>
<td></td>
</tr>
<tr>
<td>a unary relation ( \overset{1}{\rightarrow} \text{D}(_); )</td>
<td></td>
</tr>
</tbody>
</table>

satisfying

1. if \( s \overset{n+n'}{\rightarrow} s' \), \( n, n' > 0 \), then there is a \( s'' \) such that \( s \overset{n}{\rightarrow} s'' \) and \( s'' \overset{n'}{\rightarrow} s' \);

2. if \( s \overset{n}{\rightarrow} s'' \) and \( s'' \overset{n'}{\rightarrow} s' \), then \( s \overset{n+n'}{\rightarrow} s' \).

We write DTTS'(A) for the set of all discrete time transition systems with relative timing over A. Associating a transition system in DTTS'(A) with a closed term \( t \) of BPA^{drt} proceeds in essentially the same way as associating a transition system in RTTS'(A) with a closed term \( t \) of BPA^{art}. The only difference is that in the rules for the operational semantics of BPA^{drt} all numbers involved are restricted to \( \mathbb{N} \). Therefore, we refrain from giving the rules.

Bisimulation on DTTS'(A) is defined as on RTTS'(A). As in the real time cases, we obtain a model for BPA^{drt} by identifying bisimilar processes.

Algebra of communicating processes

**Signature of ACP^{drt}** The signature of ACP^{drt} is the signature of BPA^{drt} extended with the parallel composition operator \(||\): \( P' \times P' \rightarrow P' \), the left merge operator \(||\): \( P' \times P' \rightarrow P' \), the communication merge operator \(|\): \( P' \times P' \rightarrow P' \), and the encapsulation operators \( \partial_H : P' \rightarrow P' \) (for each \( H \subseteq A \)).
Axioms of ACP<sup>drt</sup> The axiom system of ACP<sup>drt</sup> consists of the axioms of BPA<sup>drt</sup> and the equations given in Table 24.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>CF1DR</td>
<td>[ \mathbf{a} \parallel \mathbf{b} = \mathbf{c} \text{ if } \gamma(a, b) = c ]</td>
</tr>
<tr>
<td>CF2DR</td>
<td>[ \mathbf{a} \parallel \mathbf{b} = \mathbf{\delta} \text{ if } \gamma(a, b) \text{ undefined} ]</td>
</tr>
<tr>
<td>CM1</td>
<td>[ x \parallel y = (\mathbf{a} \parallel y + y \parallel x) + x \parallel y ]</td>
</tr>
<tr>
<td>CMID1</td>
<td>[ \mathbf{\delta} \parallel x = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>CM2</td>
<td>[ x \parallel \mathbf{\delta} = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>CM3</td>
<td>[ \mathbf{a} \parallel (x + \mathbf{\delta}) = \mathbf{a} \cdot (x + \mathbf{\delta}) ]</td>
</tr>
<tr>
<td>CM4</td>
<td>[ \mathbf{a} \cdot x \parallel (y + \mathbf{\delta}) = \mathbf{a} \cdot (x \parallel (y + \mathbf{\delta})) ]</td>
</tr>
<tr>
<td>DRCM2</td>
<td>[ \sigma^{\mathbf{n}+1}<em>{\mathbf{rel}}(x) \parallel (v^{\mathbf{n}+1}</em>{\mathbf{rel}}(y) + \sigma^{\mathbf{n}+1}<em>{\mathbf{rel}}(z)) = \sigma^{\mathbf{n}+1}</em>{\mathbf{rel}}(x \parallel z) ]</td>
</tr>
<tr>
<td>CM5</td>
<td>[ (x + y) \parallel z = x \parallel z + y \parallel z ]</td>
</tr>
<tr>
<td>CM6</td>
<td>[ \mathbf{\delta} \parallel x = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>CM7</td>
<td>[ x \parallel \mathbf{\delta} = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>CM8</td>
<td>[ \mathbf{a} \cdot x \parallel \mathbf{b} = (\mathbf{a} \parallel \mathbf{b}) \cdot x ]</td>
</tr>
<tr>
<td>CM9</td>
<td>[ \mathbf{a} \parallel \mathbf{b} \cdot x = (\mathbf{a} \parallel \mathbf{b}) \cdot x ]</td>
</tr>
<tr>
<td>CM10</td>
<td>[ (v^{\mathbf{n}+1}<em>{\mathbf{rel}}(y) + \mathbf{\delta}) \parallel (v^{\mathbf{n}+1}</em>{\mathbf{rel}}(y) + \mathbf{\delta}) = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>DRCM3</td>
<td>[ \sigma^{\mathbf{n}+1}<em>{\mathbf{rel}}(x) \parallel (v^{\mathbf{n}+1}</em>{\mathbf{rel}}(y) + \mathbf{\delta}) = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>DRCM4</td>
<td>[ \sigma^{\mathbf{n}+1}<em>{\mathbf{rel}}(x) \parallel (v^{\mathbf{n}+1}</em>{\mathbf{rel}}(y) + \mathbf{\delta}) = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>DRCM5</td>
<td>[ (x + y) \parallel z = x \parallel z + y \parallel z ]</td>
</tr>
<tr>
<td>D0</td>
<td>[ \partial_{\mathbf{H}}(\mathbf{\delta}) = \mathbf{\delta} ]</td>
</tr>
<tr>
<td>D1DR</td>
<td>[ \partial_{\mathbf{H}}(\mathbf{a}) = \mathbf{a} \text{ if } a \notin \mathbf{H} ]</td>
</tr>
<tr>
<td>D2DR</td>
<td>[ \partial_{\mathbf{H}}(\mathbf{a}) = \mathbf{\delta} \text{ if } a \in \mathbf{H} ]</td>
</tr>
<tr>
<td>DRD</td>
<td>[ \partial_{\mathbf{H}}(\sigma^{\mathbf{n}}<em>{\mathbf{rel}}(x)) = \sigma^{\mathbf{n}}</em>{\mathbf{rel}}(\partial_{\mathbf{H}}(x)) ]</td>
</tr>
<tr>
<td>D3</td>
<td>[ \partial_{\mathbf{H}}(x + y) = \partial_{\mathbf{H}}(x) + \partial_{\mathbf{H}}(y) ]</td>
</tr>
<tr>
<td>D4</td>
<td>[ \partial_{\mathbf{H}}(x \cdot y) = \partial_{\mathbf{H}}(x) \cdot \partial_{\mathbf{H}}(y) ]</td>
</tr>
</tbody>
</table>

Table 24: Additional axioms for ACP<sup>drt</sup> \((a, b \in A_{\mathbf{\delta}}, c \in A)\)

The additional axioms of ACP<sup>drt</sup> are just simple reformulations of the additional axioms of ACP<sup>dlat</sup>. That is, constants \(\mathbf{a} (a \in A_{\mathbf{\delta}})\) have been replaced by constants \(\mathbf{a}\), and the operators \(\sigma_{\mathbf{rel}}\) and \(v_{\mathbf{rel}}\) have been replaced by \(\sigma_{\mathbf{rel}}\) and \(v_{\mathbf{rel}}\), respectively.

As in the case of ACP<sup>dlat</sup>, we can prove that the operators \(\parallel, \parallel, \mid\), and \(\partial_{\mathbf{H}}\) can be eliminated in closed terms of ACP<sup>drt</sup>.

Semantics of ACP<sup>drt</sup> Like for the rules for the operational semantics of BPA<sup>drt</sup>, the additional rules for ACP<sup>drt</sup> differ from the corresponding rules for the real time case only in that all numbers involved are restricted to \(\mathbb{N}\). Therefore, we refrain again from giving the rules.

Again, we obtain a model for ACP<sup>drt</sup> by identifying bisimilar processes.

5.3 Embedding ACP<sup>dlat</sup> in ACP<sup>sat</sup>\(\backslash V\)\(\backslash V\)

In this subsection, we will show that ACP<sup>dlat</sup> can be embedded in ACP<sup>sat</sup>\(\backslash V\). We will do so in the way outlined in Section 4.3. The explicit definitions needed are given in
Table 25. Notice that the operators $\sigma_{\text{abs}}$, $v_{\text{abs}}$ and $\overline{v}_{\text{abs}}$ of $\text{ACP}^\text{dat\vee}$ are simply defined

\[
\begin{align*}
\delta &= \int_{t \in [0,1)} \sigma^\text{abs}_{\text{abs}}(\delta) \\
\sigma^\text{abs}_{\text{abs}}(x) &= \sigma^\text{abs}_{\text{abs}}(x) \\
v^\text{abs}_{\text{abs}}(x) &= v^\text{abs}_{\text{abs}}(x) \\
\overline{v}^\text{abs}_{\text{abs}}(x) &= \overline{v}^\text{abs}_{\text{abs}}(x) \\
\forall i \cdot F &= \forall i \cdot F[\nu / i]
\end{align*}
\]

Table 25: Definitions of discrete time operators ($a \in A_\delta$)

as the operators $\sigma_{\text{abs}}$, $v_{\text{abs}}$ and $\overline{v}_{\text{abs}}$ of $\text{ACP}^\text{satI\vee}$ restricted in their first argument to $N$. We will establish the existence of an embedding by proving that for closed terms the axioms of $\text{ACP}^\text{dat\vee}$ are derivable from the axioms of $\text{ACP}^\text{satI\vee}$ and the explicit definitions given in Table 25. However, we first take another look at the connection between $\text{ACP}^\text{satI\vee}$ and $\text{ACP}^\text{dat\vee}$ by introducing the notions of a discretized real time process and a discretely initialized real time process.

In Section 3.2, we have introduced the model $M^*_A$ of $\text{ACP}^\text{satI\vee}$. The model of $\text{ACP}^\text{dat\vee}$ outlined in Section 5.1 is isomorphic to the subalgebra of $M^*_A$ generated by the embedded constants and operators of $\text{ACP}^\text{dat\vee}$. The domain of this subalgebra consists of those real time processes, i.e. elements of the domain of $M^*_A$, that are discretized. We define the notion of a discretized real time process in terms of the auxiliary discretization operator $\mathcal{D} : P^* \to P^*$ of which the defining axioms are given in Table 26. A real time process $x$ is a discretized real time process, written

\[
\begin{align*}
\mathcal{D}(\delta) &= \delta \\
\mathcal{D}(\overline{\delta}) &= \delta \\
\mathcal{D}(\sigma^\text{abs}_{\text{abs}}(x)) &= \sigma^\text{abs}_{\text{abs}}(\mathcal{D}(x)) \\
\mathcal{D}(x + y) &= \mathcal{D}(x) + \mathcal{D}(y) \\
\mathcal{D}(x \cdot y) &= \mathcal{D}(x) \cdot \mathcal{D}(y) \\
\mathcal{D}(\int_{\nu \in V} F) &= \int_{\nu \in V} \mathcal{D}(F) \\
\mathcal{D}(\forall i \cdot F) &= \forall i \cdot \mathcal{D}(F)
\end{align*}
\]

Table 26: Definition of discretization ($a \in A_\delta$)

$x \in \text{DIS}$, if $x = \mathcal{D}(x)$. The properties given in Table 27 express that the set of all discretized real time processes is closed under the operators of $\text{ACP}^\text{dat\vee}$, integration and discretization.

\[
\begin{align*}
\delta, \overline{\delta} \in \text{DIS} & \Rightarrow x \in \text{DIS} \\
x, y \in \text{DIS} & \Rightarrow x + y, x \cdot y, x | y, x || y, x | y \in \text{DIS} \\
(\forall n \in N : F[n/i] \in \text{DIS}) & \Rightarrow \forall i \cdot F \in \text{DIS} \\
(\forall p \in V : F[p/v] \in \text{DIS}) & \Rightarrow \int_{\nu \in V} F \in \text{DIS} \\
x \in \text{DIS} & \Rightarrow \mathcal{D}(x) \in \text{DIS}
\end{align*}
\]

Table 27: Properties of discretized processes ($a \in A_\delta$)

For elements $f$ of $\text{RTTS}^*(A)$, the discretization of $f$, $\mathcal{D}(f)$, is obtained as follows ($t \in \mathbb{R}_\geq 0$, $q = p + r$ and $q' = p + r')$:

37
1. for each $t$, if $\langle s, p \rangle \xrightarrow{a} \langle s', p \rangle$ in $f(t)$, then $\langle s, p' \rangle \xrightarrow{a} \langle s', p' \rangle$ in $D(f)(t)$ for each $p' \in [p], [p + 1]$;
2. for each $t$, if $\langle s, p \rangle \xrightarrow{a} \langle \sqrt{v}, p \rangle$ in $f(t)$, then $\langle s, p' \rangle \xrightarrow{a} \langle \sqrt{v}, p' \rangle$ in $D(f)(t)$ for each $p' \in [p], [p + 1]$;
3. for each $t$, if $\langle s, p \rangle \xrightarrow{t} \langle s, q \rangle$ in $f(t)$, then $\langle s, p' \rangle \xrightarrow{t'} \langle s, q' \rangle$ in $D(f)(t)$ for each $q' \in [q], [q + 1]$;
4. for each $t$, if $\text{ID}(s, p) \in f(t)$, then $\text{ID}(s, p) \in D(f)(t)$;
5. for each $t$, if neither $\text{ID}(s, p) \in f(t)$ nor either $\langle s, p \rangle \xrightarrow{a} \langle \sqrt{v}, p \rangle$ in $f(t)$ for some $a, s'$ or $\langle s, p \rangle \xrightarrow{t} \langle s, q \rangle$ in $f(t)$ for some $r$, then $\langle s, p \rangle \xrightarrow{t'} \langle s, q' \rangle$ in $D(f)(t)$ for each $q' \in (p, [p + 1])$.

Hence, for real time processes corresponding to discrete time processes, the following holds: if an action can be performed at some time $p$ such that $n \leq p < n + 1$, it can also be performed at any other time $p'$ such that $n \leq p' < n + 1$.

A real time process $x$ is a \textit{discretely initialized} real time process, written $x \in \text{DIP}$, if $x = \sqrt{v} \cdot \nu_{\text{abs}}(x)$. It follows immediately that $x \in \text{DIP} \iff x = \sqrt{v} \cdot \nu_{\text{abs}}^{(v)}(x)$. It is easy to show by induction on the term structure that all discretized processes are discretely initialized, i.e. $x \in \text{DIS} \Rightarrow x \in \text{DIP}$. Not all discretely initialized processes are discretized, e.g. $\sqrt{v} \cdot \sigma_{\text{abs}}^{(v + 1)}(\delta) \in \text{DIP}$ and $\sqrt{v} \cdot \sigma_{\text{abs}}^{(v + 1)}(\delta) \notin \text{DIS}$. This means that for real time processes corresponding to discrete time processes, initialization takes always place at discrete points in time; and that there are real time processes not corresponding to discrete time processes for which initialization takes always place at discrete points in time.

**Lemma 7** For each closed term $t$ of $\text{ACP^{dat\lor}}$ generated by the embedded constants and operators of $\text{ACP^{dat\lor}}$, $t = \sqrt{v} \cdot \nu_{\text{abs}}^{(v)}(t)$.

**Proof.** From the properties given in Table 27, we know that each process $x$ generated by the embedded constants and operators of $\text{ACP^{dat\lor}}$ is discretized, i.e. $x \in \text{DIS}$. Because $x \in \text{DIS} \Rightarrow x \in \text{DIP}$ and $x \in \text{DIP} \iff x = \sqrt{v} \cdot \nu_{\text{abs}}^{(v)}(x)$, the result immediately follows.

The following lemmas present other useful properties of discrete time processes.

**Lemma 8** For each closed term $t$ of $\text{ACP^{dat\lor}}$ generated by the embedded constants and operators of $\text{ACP^{dat\lor}}$, there exists a term $t'$ containing no other free variable than $v$ such that for each $p \in \mathbb{R}_{\geq 0}$: $\nu_{\text{abs}}^{(p)}(t) = \sigma_{\text{abs}}^{(p)}(t'[p/v])$, $t'[p/v] = \nu_{\text{abs}}^{(p)}(t'[p/v])$, and if $p \in [0, 1)$ and $t \neq \delta$, $t'[p/v] = t'[p/v] + \sigma_{\text{abs}}^{(p)}(\delta)$ and $\nu_{\text{abs}}^{(p)}(t + \delta) = \nu_{\text{abs}}^{(p)}(t'[p/v] + \delta)$.

In subsequent proofs, we write $t[v]$ for a fixed but arbitrary term $t'$ that fulfills these conditions.

**Proof.** Observe that, if $p \in [0, 1)$ and $t \neq \delta$, $\nu_{\text{abs}}^{(p)}(t + \delta) = \sigma_{\text{abs}}^{(p)}(t'[p/v] + \delta)$ follows directly from $\nu_{\text{abs}}^{(p)}(t) = \sigma_{\text{abs}}^{(p)}(t'[p/v])$ and $t'[p/v] = t'[p/v] + \sigma_{\text{abs}}^{(p)}(\delta)$. Observe further that, if $t'$ is a term that fulfills all above-mentioned conditions but $t'[p/v] = \nu_{\text{abs}}^{(p)}(t'[p/v])$, $\nu_{\text{abs}}^{(p)}(t')$ is a term that fulfills all conditions. Consequently, it suffices to prove that there exists a $t'$ such that $\nu_{\text{abs}}^{(p)}(t) = \sigma_{\text{abs}}^{(p)}(t'[p/v])$ and, if $p \in [0, 1)$ and $t \neq \delta$, $t'[p/v] = t'[p/v] + \sigma_{\text{abs}}^{(p)}(\delta)$. It is straightforward to prove this by induction on the structure of $t$. We present only the case.
that \( t \) is of the form \( a \). The other cases are simpler or similar to corresponding cases in the proof of Lemma 3.

\[
\begin{align*}
\overline{\nu}^p_{\text{abs}}(\delta) &= \overline{\nu}^p_{\text{abs}}(\int_{u \in [0,1]} \sigma^w_{\text{abs}}(\tilde{a}\cdot)) \overset{\text{SAT1,5}}{=}= \int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) \\
\overline{\nu}^p_{\text{abs}}(\delta) &= \overline{\nu}^p_{\text{abs}}(\int_{u \in [0,1]} \sigma^w_{\text{abs}}(\tilde{a}\cdot)) \\
\int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) &= \int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) \\
\int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) &= \int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) \\
\int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) &= \int_{u \in [0,1]} \overline{\nu}^p_{\text{abs}}(\sigma^w_{\text{abs}}(\tilde{a}\cdot)) \\
\end{align*}
\]

**Lemma 9** For each \( p \in \mathbb{R}_{\geq 0} \) and closed term \( t \) of \( \text{ACP}^{\text{sat1}/v} \) generated by the embedded constants and operators of \( \text{ACP}^{\text{dat}/v} \), there exists a closed term \( t' \) such that \( \overline{\nu}^p_{\text{abs}}(t) = \sigma^p_{\text{abs}}(t') \), \( t = \overline{\nu}^p_{\text{abs}}(t') \), and if \( p \in [0,1] \) and \( t \neq \bar{\delta} \), \( t' = t + \sigma^p_{\text{abs}}(\bar{\delta}) \) and \( \overline{\nu}^p_{\text{abs}}(t + \bar{\delta}) = \sigma^p_{\text{abs}}(t' + \bar{\delta}) \). In subsequent proofs, we write \( t_{[v]} \) for a fixed but arbitrary closed term \( t' \) that fulfills these conditions – like in case of applications of Lemma 3.

**Proof.** This follows immediately from Lemma 8.

**Lemma 10** For each closed term \( t \) of \( \text{ACP}^{\text{sat1}/v} \) generated by the embedded constants and operators of \( \text{ACP}^{\text{dat}/v} \), there exists a closed term \( t' \) such that \( v^p_{\text{abs}}(t + \bar{\delta}) = \int_{v \in [0,1]} \sigma^w_{\text{abs}}(\nu^p_{\text{abs}}(t')) \). In subsequent proofs we write \( t^v \) for a fixed but arbitrary closed term \( t' \) that fulfills this condition.

**Proof.** It is straightforward to prove this by induction on the structure of \( t \). We present only the case that \( t \) is of the form \( a \cdot t'' \). The other cases are simpler.

\[
\begin{align*}
v^1_{\text{abs}}(a \cdot t'' + \bar{\delta}) &= v^1_{\text{abs}}((\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot)) \cdot t'' + \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\bar{\delta})) \\
v^1_{\text{abs}}(\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot)) \cdot t'' + v^1_{\text{abs}}(\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\bar{\delta})) &= \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot t'' + \bar{\delta} \\
(\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot)) \cdot t'' + \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\bar{\delta}) &= \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot t'' + \bar{\delta} \\
\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot t'' &= \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot v^0_{\text{abs}}(t'') \\
\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot v^0_{\text{abs}}(t'') &= \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot v^0_{\text{abs}}(t'') \\
\int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot v^0_{\text{abs}}(t'') &= \int_{v \in [0,1]} \sigma^v_{\text{abs}}(\tilde{a}\cdot) \cdot v^0_{\text{abs}}(t'') \\
\end{align*}
\]

Lemmas 7-10 are used to shorten the calculations in the proof of Theorem 12. The following lemma is also used in the proof of that theorem.

**Lemma 11** For \( p \in [0,1] \), the equation \( \overline{\nu}^p_{\text{abs}}(\bar{\delta}) = \sigma^1_{\text{abs}}(\bar{\delta}) \) is derivable from the axioms of \( \text{ACP}^{\text{sat1}/v} \) and the explicit definition of the constants and operators in Table 25.

**Proof.**
The existence of an embedding of $\text{ACP}^{\text{dat}}$ in $\text{ACP}^{\text{sat}I\forall}$ is now established by proving the following theorem.

**Theorem 12 (Embedding $\text{ACP}^{\text{dat}}$ in $\text{ACP}^{\text{sat}I\forall}$)** For closed terms, the axioms of $\text{ACP}^{\text{dat}}$ are derivable from the axioms of $\text{ACP}^{\text{sat}I\forall}$ and the explicit definitions of the constants and operators $\alpha$, $\sigma_{\text{abs}}$, $v_{\text{abs}}$, $\overline{v}_{\text{abs}}$ and $\sqrt{d}$ in Table 25.

**Proof.** The proof of this theorem is given in Appendix A.2. The proof is a matter of straightforward calculations. Lemmas 1 and 3 (pages 7-8) and Lemmas 7-11 (page 38-39) are very useful in the proof. □

6 Concluding remarks

We presented real time and discrete time versions of ACP with both absolute timing and relative timing, starting with a new real time version of ACP with absolute timing called $\text{ACP}^{\text{sat}}$. We demonstrated that $\text{ACP}^{\text{sat}}$ extended with integration and initial abstraction generalizes the presented real time version with relative timing and the presented discrete time version with absolute timing. We focussed on versions of ACP with timing where execution of actions and passage of time are separated, but explained how they can be combined in these versions. The material resulted from a systematic study of some of the most important issues relevant to dealing with time-dependent behaviour of processes – viz. absolute vs relative timing, continuous vs discrete time scale, and separation vs combination of execution of actions and passage of time – in the setting of ACP.

All real time and discrete time versions of ACP presented in this paper include the immediate deadlock constant $\delta$. This constant enables us to distinguish timing inconsistencies from incapabilities of performing actions as well as idling. This is certainly relevant to versions with absolute timing because timing inconsistencies readily arise. The usefulness of the immediate deadlock constant in practice is not yet clear for versions with relative timing. Minor adaptations of the versions of ACP with relative timing presented in this paper are needed to obtain versions without the immediate deadlock constant.

The discrete time versions of ACP presented in this paper are conservative extensions of the discrete time versions of [6]. The real time versions presented in this paper, unlike the real time versions of [1] and [3], do not exclude the possibility of two or more actions to be performed consecutively at the same point in time. This feature seems to be essential to obtain simple and natural embeddings of discrete time versions as well as useful in practice when describing and analyzing distributed systems where entirely independent actions happen at different locations.
We did not extend the different versions of ACP with timing presented in this paper with recursion, abstraction, and other features that are important to make these versions suitable for being applied. This has been done for the earlier versions of ACP with timing referred to in this paper. Some of those versions have been successfully used for describing and analyzing systems and protocols of various kinds, see e.g. [13], [22], [25], [33], [34] and [36], as well as for defining semantics of programming and specification languages, see e.g. [10], [12] and [14].

We did not give explicit consideration to other algebraic concurrency theories that deal with time-dependent behaviour. In general, they have urgent actions and relative timing. This is, for example, the case with ATP [30], the different versions of CCS with timing [16, 28, 35] and TIC [32] – TIC is rooted in LOTOS [37]. We claim, on the basis of the connections described in [5], that there are indeed close connections between these theories and the versions of ACP with relative timing presented in this paper, i.e. ACP_{rt} and ACP_{drt}. We also claim that there is a close connection between TPL [21] and ACP_{drt}, only TPL is based on testing equivalence instead of bisimulation equivalence. Timed CSP [18], which is based on timed traces and timed failures, has urgent actions and relative timing as well. In [17], the CCS-like process algebras with timing of [21], [28], [29] and [35] are compared.

References


A Proofs of theorems

A.1 Theorem 6

Theorem 6 (Embedding \( \text{ACP}^{\text{art}} \) in \( \text{ACP}^{\text{sat}, \forall} \)) For closed terms, the axioms of \( \text{ACP}^{\text{art}} \) are derivable from the axioms of \( \text{ACP}^{\text{sat}, \forall} \) and the explicit definitions of the constants and operators \( \overline{a}, \sigma_{\text{rel}}, u_{\text{rel}}, \overline{v}_{\text{rel}}, \) and \( v_{\text{rel}} \) in Table 19.

Proof.
To begin with, we show that the axioms of BPA* are derivable for closed terms. Throughout this proof we do not expound the trivial cases.

**SRT1**: \( s^0_{rel}(t) = \sqrt{v} \cdot \bar{u}_{abs}(t) \)  

**SRT2**: \( s^0_{rel}(s^0_{rel}(t)) = \sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(t)) \)  

**SRT3**: \( s^0_{rel}(t) + s^0_{rel}(t') = (\sqrt{v} \cdot \bar{u}_{abs}(t) + \sqrt{v} \cdot \bar{u}_{abs}(t')) \)  

**SRT4**: \( s^0_{rel}(t) \cdot t' = (\sqrt{v} \cdot \bar{u}_{abs}(t)) \cdot t' \)  

**A6SRa**: \( \tilde{\alpha} + \tilde{\delta} = (\sqrt{v} \cdot \sigma_{abs}(\tilde{\alpha})) + (\sqrt{v} \cdot \sigma_{abs}(\tilde{\delta})) \)  

**A6SRb**: \( \sigma_{rel}(t) + \tilde{\delta} = (\sqrt{v} \cdot \bar{u}_{abs}(t)) + (\sqrt{v} \cdot \sigma_{abs}(\tilde{\delta})) \)  

**A7SR**: \( \tilde{\delta} \cdot t = (\sqrt{v} \cdot \sigma_{abs}(\tilde{\delta})) \cdot t \)  

**SRT0**: \( \bar{u}^p_{rel}(t) = \sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(t)) \)  

**SRT1**: \( \bar{u}^p_{rel}(\tilde{\alpha}) = \sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(\tilde{\alpha})) \)  

**SRT2**: \( \bar{u}^p_{rel}(t) = \sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(t)) \)  

**SRT3**: \( \bar{u}^p_{rel}(p) = \sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(p)) \)  

**SRT4**: \( \bar{u}^p_{rel}(t + t') = \sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(t + \bar{u}_{abs}(t'))) \)  

**SRT5**: \( \bar{u}^p_{rel}(t + t') = (\sqrt{v} \cdot \bar{u}_{abs}(\sqrt{v} \cdot \bar{u}_{abs}(t + t'))) \)
SR10: \[
\overline{u}^{p}_{rel}(\delta) = \overline{v} \cdot \overline{u}^{\nu+p}_{abs}(\overline{u}_{abs}(\delta)) = \overline{v} \cdot \overline{u}^{\nu+p}_{abs}(\overline{u}_{abs}(\delta)) \]

SR11: \[
\overline{u}^{0}_{rel}(t) = \overline{v} \cdot \overline{u}^{\nu}_{abs}(\overline{u}_{abs}(t)) = \overline{v} \cdot \overline{u}^{\nu}_{abs}(t) = \overline{v} \cdot t \]

SR12: \[
\overline{u}^{r}_{rel}(\overline{u}_{abs}(\delta)) = \overline{v} \cdot \overline{u}^{\nu+r}_{abs}(\overline{u}_{abs}(\delta)) \]

SR13: \[
\overline{u}^{p+q}_{rel}(\overline{u}_{abs}(t)) = \overline{v} \cdot \overline{u}^{\nu+p+q}_{abs}(\overline{u}_{abs}(t)) \]

SR14: \[
\overline{u}^{p+q}_{rel}(t+t') = \overline{v} \cdot \overline{u}^{\nu+p+q}_{abs}(\overline{u}_{abs}(t) \cdot t') \]

SR15: \[
\overline{u}^{p}_{rel}(t \cdot t') = \overline{v} \cdot \overline{u}^{\nu+p}_{abs}(\overline{u}_{abs}(t) \cdot t') \]

Next, we show that the additional axioms for ACP\textsuperscript{act} are derivable for closed terms.

CF1SR: \[
\overline{a} \parallel \overline{b} = (\overline{v} \cdot \overline{u}^{\nu}_{abs}(\overline{u}_{abs}(\delta)) = \overline{a} \parallel \overline{b} \]

CF2SR: The proof is similar to the proof of axiom CF1SR - axiom CF2SA is used instead of axiom CF1SA.

CM2SRID: \[
\overline{a} \parallel (t + \overline{\delta}) = (\overline{v} \cdot \overline{u}^{\nu}_{abs}(\overline{u}_{abs}(t + \overline{\delta})) \]

CM3SRID: The proof is similar to the proof of axiom CM2SRID - axiom CM3SA is used instead of axiom CM2SA.
SRCM1ID:
\[ \sigma^r_{rel}(t) \parallel (\nu_{rel}(t') + \delta) = \]
\[ (\forall v. \nu_{abs}(v'(t'))) \parallel ((\forall v \cdot \sigma^v_{abs}(\nu_{abs}(t'))) + (\forall v \cdot \sigma^v_{abs}(\delta))) \text{ DISTR} \]
\[ \forall v. (\nu_{abs}(v'(t')) \parallel (\sigma^v_{abs}(\nu_{abs}(t'))) + \sigma^v_{abs}(\delta)) \text{ Lemma 4, SAT3} \]
\[ \forall v. (\sigma^v_{abs}(t') \parallel (\sigma^v_{abs}(\nu_{abs}(t') + \delta))) \text{ S ACM4'} \]
\[ \forall v. \sigma^v_{abs}(\delta) = \delta \]

SRCM2ID:
\[ \sigma^p_{rel}(t) \parallel (\nu_{rel}(t') + \sigma^p_{rel}(t'')) = \]
\[ (\forall v. \nu_{abs}(t')) \parallel ((\forall v \cdot \nu_{abs}(\nu_{abs}(t'))) + (\forall v \cdot \nu_{abs}(t''))) \text{ DISTR} \]
\[ \forall v. (\nu_{abs}(t') \parallel (\nu_{abs}(\nu_{abs}(t')) + \nu_{abs}(t''))) \text{ Lemma 4, SAT3} \]
\[ \forall v. (\sigma^v_{abs}(t') \parallel (\nu_{abs}(\nu_{abs}(t')) + \nu_{abs}(t''))) \text{ S ACM2} \]
\[ \forall v. \sigma^v_{abs}(t' \parallel t'') \text{ Lemma 4, SAT3} \]
\[ \forall v. \sigma^v_{abs}(t) = \sigma^v_{rel}(t \parallel t'') \]

CM5SR:
\[ \tilde{a}.t \parallel \tilde{b} = ((\forall v \cdot \sigma^v_{abs}(\tilde{a})) \cdot t) \parallel ((\forall v \cdot \sigma^v_{abs}(\tilde{b})) \text{ SIA6, DISTR} \]
\[ \forall v. ((\sigma^v_{abs}(\tilde{a}) \cdot t) \parallel \sigma^v_{abs}(\tilde{b})) \text{ SIA6, Lemma 4} \]
\[ \forall v. ((\sigma^v_{abs}(\tilde{a}) \cdot \sigma^v_{abs}(t)) \parallel \sigma^v_{abs}(\tilde{b})) \text{ SAT5} \]
\[ \forall v. (\sigma^v_{abs}(\tilde{a} \cdot \tilde{b}) \parallel \sigma^v_{abs}(\tilde{b})) \text{ S ACM5 } \]
\[ \forall v. (\sigma^v_{abs}(\tilde{a} \parallel \tilde{b}) \cdot \sigma^v_{abs}(\tilde{b})) \text{ S ACM5, DISTR} \]
\[ (\forall v \cdot \sigma^v_{abs}(\tilde{a} \parallel \tilde{b})) \cdot t = ((\forall v \cdot \sigma^v_{abs}(\tilde{a})) \cdot t) \text{ SIA6} \]

CM6SR and CM7SR:
The proofs are similar to the proof of axiom CM5SR – axioms CM6SA and CM7SA are used instead of axiom CM5SA.

SRCM3ID:
\[ (\nu_{rel}(t) + \delta) \parallel \sigma^r_{rel}(t') = \]
\[ (\forall v \cdot \sigma^v_{abs}(\nu_{abs}(t'))) + (\forall v \cdot \sigma^v_{abs}(\delta)) \parallel (\forall v \cdot \nu_{abs}(t')) \text{ DISTR} \]
\[ \forall v. (\nu_{abs}(t') \parallel (\sigma^v_{abs}(\nu_{abs}(t')) + \nu_{abs}(\delta)) \text{ Lemma 4, SAT3} \]
\[ \forall v. (\sigma^v_{abs}(t') \parallel (\sigma^v_{abs}(\nu_{abs}(t')) + \sigma^v_{abs}(\delta))) \text{ S ACM3'} \]

SRCM4ID:
The proof is similar to the proof of axiom SRCM3ID – axiom SAC4'M is used instead of axiom SAC3'.

SRCM5:
\[ \sigma^p_{rel}(t) \parallel \sigma^p_{rel}(t') = (\forall v. \nu_{abs}(t')) \parallel (\forall v. \nu_{abs}(t')) \text{ DISTR, SH1} \]
\[ \forall v. \nu_{abs}(t' \parallel t') = \sigma^p_{rel}(t' \parallel t') \]

D1SR:
\[ \delta_H(\tilde{a}) = \delta_H(\forall v \cdot \sigma_{abs}(\tilde{a})) \text{ SIA15, SAD} \]
\[ \forall v \cdot \sigma_{abs}(\delta_H(\tilde{a})) \text{ D ISA} \]
\[ \forall v \cdot \sigma_{abs}(\delta) = \tilde{a} \text{ if } a \notin H \]

D2SR:
The proof is similar to the proof of axiom D1SR – axiom D2SA is used instead of axioms D1SA.

SRD:
\[ \delta_H(\sigma^p_{rel}(t)) = \delta_H(\forall v \cdot \nu_{abs}(t)) \text{ SIA15, Lemma 4, SAD} \]
\[ \forall v \cdot \nu_{abs}(\delta_H(t)) = \sigma^p_{rel}(\delta_H(t)) \]
SRU0: \[ \nu_{\text{rel}}(\delta) = \nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(\delta)) \quad \text{SAU, SAD} \]
\[ = \nu v \cdot \nu_{\text{abs}}^v(\delta) \quad \text{SIA1, SIA4} \]

SRU1: \[ \nu_{\text{rel}}(\delta) = \nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(\nu w \cdot \sigma_{\text{abs}}^w(\delta))) \quad \text{SIA16} \]
\[ = \nu v \cdot \sigma_{\text{abs}}^v(\nu w \cdot \sigma_{\text{abs}}^w(\delta)) \quad \text{SIA16} \]
\[ = \nu v \cdot \sigma_{\text{abs}}^v(\nu w \cdot \sigma_{\text{abs}}^w(\delta)) \quad \text{SAA1, SAD} \]
\[ = \nu v \cdot \sigma_{\text{abs}}^v(\delta) = \delta \]

SRU2: \[ \nu_{\text{rel}}(\sigma_{\text{rel}}(t)) = \nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(\nu w \cdot \nu_{\text{abs}}^w(t))) \quad \text{SIA16} \]
\[ = \nu v \cdot \sigma_{\text{abs}}^v(\nu w \cdot \nu_{\text{abs}}^w(t)) \quad \text{SIA16} \]
\[ = \nu v \cdot \sigma_{\text{abs}}^v(\nu w \cdot \nu_{\text{abs}}^w(t)) \quad \text{SAA1, SAD} \]
\[ = \nu v \cdot \nu_{\text{rel}}(\delta) = \delta \]

SRU3: \[ \nu_{\text{rel}}(t + t') = \nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(t + t')) \quad \text{SAU3, SAT3} \]
\[ = \nu v \cdot (\sigma_{\text{abs}}^v(\nu_{\text{abs}}(t)) + \sigma_{\text{abs}}^v(\nu_{\text{abs}}(t'))) \quad \text{SAA1} \]
\[ = \nu v \cdot (\sigma_{\text{abs}}^v(\nu_{\text{abs}}(t)) + \nu_{\text{abs}}^v(\nu_{\text{abs}}(t'))) \quad \text{SAA2} \]
\[ = \nu v \cdot (\sigma_{\text{abs}}^v(\nu_{\text{abs}}(t)) + \nu_{\text{abs}}^v(\nu_{\text{abs}}(t'))) \quad \text{SAA8} \]
\[ = \nu v \cdot \nu_{\text{rel}}(t) + \nu_{\text{rel}}(t') \]

SRU4: \[ \nu_{\text{rel}}(t \cdot t') = \nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(t \cdot t')) \quad \text{SAU4} \]
\[ = \nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(t) \cdot t') \quad \text{SAA1} \]
\[ = \nu v \cdot (\sigma_{\text{abs}}^v(\nu_{\text{abs}}(t)) \cdot t') \quad \text{SAA1} \]
\[ = (\nu v \cdot \sigma_{\text{abs}}^v(\nu_{\text{abs}}(t))) \cdot t' = \nu_{\text{rel}}(t) \cdot t' \]

\[ \square \]

A.2 Theorem 12

Theorem 12 (Embedding \( \text{ACP}^\text{dat} \) in \( \text{ACP}^\text{satI} \)) For closed terms, the axioms of \( \text{ACP}^\text{dat} \) are derivable from the axioms of \( \text{ACP}^\text{satI} \) and the explicit definitions of the constants and operators \( \sigma, \sigma_{\text{abs}}, \nu_{\text{abs}}, \nu_{\text{abs}}^v \) and \( \nu_{\text{rel}} \) in Table 25.

Proof.

To begin with, we show that the axioms of \( \text{BPA}^\text{dat} \) are derivable for closed terms. Throughout this proof we do not expound the trivial cases.

\[ \text{DAT7:} \quad \sigma_{\text{abs}}^v(\delta) \quad \text{INT7} \quad \int_{v \in [0,1]} \sigma_{\text{abs}}^v(\delta) = \delta \]

\[ \text{A6DA:} \quad \sigma_{\text{abs}}^v(\tilde{\delta}) = \sigma_{\text{abs}}^v(\delta) \quad \text{INT10} \quad \int_{v \in [0,1]} \sigma_{\text{abs}}^v(\tilde{\delta}) = \sigma_{\text{abs}}^v(\delta) \]

\[ \text{DAT01:} \quad \sigma_{\text{abs}}^{n+1}(\tilde{\delta}) = \sigma_{\text{abs}}^{n+1}(\int_{v \in [0,1]} \sigma_{\text{abs}}^v(\tilde{\delta})) \quad \text{SAT06} \quad \int_{v \in [0,1]} \sigma_{\text{abs}}^{n+1}(\tilde{\delta}) = \sigma_{\text{abs}}^{n+1}(\tilde{\delta}) \]

Next, we show that the additional axioms for \( \text{ACP}^\text{dat} \) are derivable for closed terms.
CF1DA: 
\[ a \parallel b = (\forall_{a \in [0,1]} \sigma_{abs}^v(\tilde{a})) \parallel (\forall_{b \in [0,1]} \sigma_{abs}^v(\tilde{b})) \quad \text{INT}^{13,14} \]
\[ \forall_{a \in [0,1]} \forall_{b \in [0,1]} (\sigma_{abs}^v(\tilde{a}) \parallel \sigma_{abs}^v(\tilde{b})) \quad \text{INT}^{13,14} \]
\[ \forall_{a \in [0,1]} \forall_{b \in [0,0]} (\sigma_{abs}^v(\tilde{a}) \parallel \sigma_{abs}^v(\tilde{b})) + \forall_{c \in [0,0]} (\sigma_{abs}(\tilde{a}) \parallel \sigma_{abs}(\tilde{b})) + \]
\[ \forall_{a \in [0,1]} \forall_{b \in [0,1]} (\sigma_{abs}^v(\tilde{a}) \parallel \sigma_{abs}^v(\tilde{b})) \quad \text{see} \quad \text{ILL,III} \]
\[ \forall_{a \in [0,1]} \sigma_{abs}^v(\tilde{a}) + \forall_{b \in [0,1]} \sigma_{abs}^v(\tilde{b}) + \forall_{c \in [0,0]} \sigma_{abs}(\tilde{c}) \quad \text{INT}^{10, SAT3} \]
\[ \forall_{c \in [0,0]} \sigma_{abs}(\tilde{c}) \equiv \sigma_{abs}^v(\tilde{c}) \quad \text{AID, AID} \]
\[ \forall_{a \in [0,1]} (\sigma_{abs}(\tilde{b}) \equiv \sigma_{abs}^v(\tilde{b})) \quad \text{CM}^{2A} \]

I. Suppose \( p \in [0,1], \ q \in [0, p] \).

Then \( \sigma_{abs}^v(\tilde{a}) \equiv \sigma_{abs}^v(\tilde{b}) \quad \text{SAU1, AID} \]
\[ \equiv \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{INT}^{5} \), \( \forall_{x \in [0,p]} (\sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b})) = \forall_{x \in [0,p]} (\sigma_{abs}(\tilde{b})) \equiv \text{INT}^{5} \sigma_{abs}^v(\tilde{b}). \]
By \( \text{INT}^{5} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b})) = \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{b})). \]

II. Suppose \( p \in [0,1], \ q = p \).

Then \( \sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b}) \quad \text{SAU1, AID} \]
By \( \text{INT}^{5} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b})) = \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{b})). \]

III. Suppose \( p \in [0,1], \ q \in [p, 1] \).

Then \( \sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b}) \quad \text{SAU1, AID} \]
By \( \text{INT}^{5} \), \( \forall_{x \in [p,1]} (\sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b})) = \forall_{x \in [p,1]} (\sigma_{abs}(\tilde{b})). \]
By \( \text{INT}^{5} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a}) \equiv \sigma_{abs}(\tilde{b})) = \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{b})). \]

CF2DA: 
The proof is similar to the proof of axiom CF1DA – axiom CF2SA is used instead of axiom CF1SA.

CM2DA: 
\( \forall p \in [0, \infty), \ q \in [0, 1], \)
\[ \forall_{x \in [0,1]} (\sigma_{abs}^v(\tilde{a})) \parallel \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \quad \text{ILL,III} \]
By \( \text{INT}^{5} \), \( \forall_{x \in [0,1]} (\sigma_{abs}^v(\tilde{a})) \parallel \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) = \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})). \]
By \( \text{SAI10, SI15, 6, INT}^{11,12} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \parallel (t + \tilde{a}) = \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a}) \parallel (t + \tilde{a})). \]
By \( \text{SIA5} \), \( a \parallel (t + \tilde{a}) = a \cdot (t + \tilde{a}). \)

I. Suppose \( p \in [0, q], \ q \in [0, 1], \ t \neq \tilde{a} \).

Then \( \forall_{x \in [0,1]} (\sigma_{abs}^v(\tilde{a})) \parallel \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \quad \text{SAU1, CM}^{2A} \]
\[ \sigma_{abs}(\tilde{a}) \parallel \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{LEMMA}^{9} \), \( \forall_{x \in [0,1]} (\sigma_{abs}^v(\tilde{a})) \parallel \sigma_{abs}(\tilde{a}) = \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{LEMMA}^{2} \), \( \forall_{x \in [0,1]} (\sigma_{abs}^v(\tilde{a})) \parallel \sigma_{abs}(\tilde{a}) = \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{LEMMA}^{3} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \parallel (\forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a}))) = \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{LEMMA}^{11} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \parallel (t + \tilde{a}) = \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{SIA5} \), \( \sigma_{abs}(\tilde{a}) \parallel (t + \tilde{a}) = a \cdot (t + \tilde{a}). \]

II. Suppose \( p \in [0, q], \ q \in [0, 1], \ t = \tilde{a} \).

Then \( \forall_{x \in [0,1]} (\sigma_{abs}^v(\tilde{a})) \parallel \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \quad \text{SAU1, AID} \]
\[ \sigma_{abs}(\tilde{a}) \parallel (t + \tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{LEMMA}^{11} \), \( \forall_{x \in [0,1]} (\sigma_{abs}(\tilde{a})) \parallel (t + \tilde{a}) = \sigma_{abs}(\tilde{a}) \quad \text{SAU1, CM}^{2A} \]
By \( \text{SIA5} \), \( \sigma_{abs}(\tilde{a}) \parallel (t + \tilde{a}) = a \cdot (t + \tilde{a}). \]
III. Suppose \( p \in (q, \infty) \), \( q \in [0,1) \).

Then \( \sigma_{\text{abs}}^q(\sigma_{\text{abs}}(\tilde{a})) \parallel \sigma_{\text{abs}}^p(t + \delta) \overset{\text{Lemm.9}}{=} \sigma_{\text{abs}}^q(\sigma_{\text{abs}}(\tilde{a})) \parallel \sigma_{\text{abs}}^p(t + \delta) \).

\( \sigma_{\text{abs}}^q(\sigma_{\text{abs}}(\tilde{a})) \parallel \sigma_{\text{abs}}^p(t + \delta) \overset{\text{SAT2, SADM2, CMID1}}{=} \sigma_{\text{abs}}^p(t + \delta) \).

\( \sigma_{\text{abs}}^p(t + \delta) \overset{\text{SAT9}}{=} \sigma_{\text{abs}}^q(\sigma_{\text{abs}}(\tilde{a})) \cdot (t + \delta) \).

CM3DA: The proof is similar to the proof of axiom CM2DA – axiom CM3SA is used instead of axiom CM2SA.

CM5DA, CM6DA and CM7DA:

The proofs are similar to the proof of axiom CF1DA – axiom SIA6 is used in addition and axioms CM5SA, CM6SA and CM7SA are used instead of axiom CF1SA.

DACM3:

\[
\begin{align*}
\sigma_{\text{abs}}^1(\tilde{a}) + \sigma_{\text{abs}}^1(t') & \overset{\text{INT2,3}}{=} \sigma_{\text{abs}}^{1+1}(\tilde{a} + t') \\
(\sigma_{\text{abs}}^1(\tilde{a}) + \sigma_{\text{abs}}^1(t)) & \overset{\text{SAT0,3,6}}{=} \sigma_{\text{abs}}^{1+1}(\tilde{a} + t') \\
\sigma_{\text{abs}}^1(\tilde{a}) + \sigma_{\text{abs}}^1(t) & \overset{\text{SAT0,3,6}}{=} \sigma_{\text{abs}}^{1+1}(\tilde{a} + t') \\
\sigma_{\text{abs}}^1(\tilde{a}) + \sigma_{\text{abs}}^1(t) & \overset{\text{SAT0,3,6}}{=} \sigma_{\text{abs}}^{1+1}(\tilde{a} + t')
\end{align*}
\]

DACM4:
The proof is similar to the proof of axiom DACM3 – axioms INT14 and SACM4' are used instead of axioms INT13 and SACM3'.

D1DA:

\[
\begin{align*}
\theta_H(\tilde{a}) & = \theta_H(\sigma_{\text{abs}}^1(\tilde{a})) \overset{\text{INT15, SAD}}{=} \int_{t \in [0,1]} \sigma_{\text{abs}}^1(\tilde{a}) \overset{\text{D1SA}}{=} \theta_H(\tilde{a})
\end{align*}
\]

D2DA:
The proof is similar to the proof of axiom D1DA – axiom D2SA is used instead of axioms D1SA.

Finally, we show that the additional axioms for discrete initial abstraction are derivable for closed terms.
DIA1: \( \forall v \cdot G[v/w] = \forall v \cdot G[w/v] \)  
DIA2: \( \forall v \cdot (F[v]/i) = \forall v \cdot (F[w]/j) \)  
DIA3: \( \forall v \cdot (F[v]/i) = \forall v \cdot (F[w]/j) \)  
DIA4: \( G \mapsto \forall v \cdot G[v/w] \)  
DIA5: Suppose \( p \in \mathbb{R}_{\geq 0} \) and \( n \in \mathbb{N} \). \( F' \) is \( F \).  
DIA6: \( \sigma_{abs}(a) \cdot F = \sigma_{abs}^{n} (F) \)  
DIA7: \( \sigma_{abs}(\forall v \cdot F[v/i]) = \sigma_{abs}^{n} (F[v/i]) \)  
DIA8: \( (\forall v \cdot F[v/i]) + G = (\forall v \cdot F[w/i]) + G \)  
DIA9: \( (\forall v \cdot F[v/i]) + G = \forall v \cdot (F[w/i]) + G \)  
DIA10: \( \forall v \cdot F[v/i] = \forall v \cdot (F[w/i]) \)  
DIA11, DIA12, DIA13 and DIA14: \( \)The proofs are similar to the proof of axiom DIA8 – axioms SIA11, SIA12, SIA13 and SIA14 are used instead of axioms SIA8.  
DIA15: The proof is similar to the proof of axiom DIA10 – axiom SIA15 is used instead of axioms SIA10.
<table>
<thead>
<tr>
<th>Number</th>
<th>Authors</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>96/01</td>
<td>M. Voorhoeve and T. Basten</td>
<td>Process Algebra with Autonomous Actions</td>
<td>12</td>
</tr>
<tr>
<td>96/02</td>
<td>P. de Bra and A. Aerts</td>
<td>Multi-User Publishing in the Web: DreSS, A Document Repository Service Station</td>
<td>12</td>
</tr>
<tr>
<td>96/03</td>
<td>W.M.P. van der Aalst</td>
<td>Parallel Computation of Reachable Dead States in a Free-choice Petri Net</td>
<td>26</td>
</tr>
<tr>
<td>96/05</td>
<td>T. Basten and W.M.P. v.d. Aalst</td>
<td>A Process-Algebraic Approach to Life-Cycle Inheritance</td>
<td>15</td>
</tr>
<tr>
<td>96/06</td>
<td>W.M.P. van der Aalst and T. Basten</td>
<td>Life-Cycle Inheritance A Petri-Net-Based Approach</td>
<td>18</td>
</tr>
<tr>
<td>96/07</td>
<td>M. Voorhoeve</td>
<td>Structural Petri Net Equivalence</td>
<td>16</td>
</tr>
<tr>
<td>96/08</td>
<td>A.T.M. Aerts, P.M.E. De Bra, J.T. de Munk</td>
<td>OODB Support for WWW Applications: Disclosing the internal structure of Hyperdocuments</td>
<td>14</td>
</tr>
<tr>
<td>96/09</td>
<td>F. Dignum, H. Weigard, E. Verharen</td>
<td>A Formal Specification of Deadlines using Dynamic Deontic Logic</td>
<td>18</td>
</tr>
<tr>
<td>96/10</td>
<td>R. Bleo, H. Geuvers</td>
<td>Explicit Substitution: on the Edge of Strong Normalisation</td>
<td>13</td>
</tr>
<tr>
<td>96/11</td>
<td>T. Laan</td>
<td>AUTOMATH and Pure Type Systems</td>
<td>30</td>
</tr>
<tr>
<td>96/12</td>
<td>F. Kamareddine and T. Laan</td>
<td>A Correspondence between Nuprl and the Ramified Theory of Types</td>
<td>12</td>
</tr>
<tr>
<td>96/13</td>
<td>T. Borghuis</td>
<td>Prioran Tense Logics in Modal Pure Type Systems</td>
<td>61</td>
</tr>
<tr>
<td>96/14</td>
<td>S.H.J. Bos and M.A. Reniers</td>
<td>The $I^2$ C-bus in Discrete-Time Process Algebra</td>
<td>25</td>
</tr>
<tr>
<td>96/15</td>
<td>M.A. Reniers and J.J. Vereijken</td>
<td>Completeness in Discrete-Time Process Algebra</td>
<td>139</td>
</tr>
<tr>
<td>96/17</td>
<td>E. Boiten and P. Hoogendijk</td>
<td>Nested collections and polytypism</td>
<td>11</td>
</tr>
<tr>
<td>96/18</td>
<td>P.D.V. van der Stok</td>
<td>Real-Time Distributed Concurrency Control Algorithms with mixed time constraints</td>
<td>71</td>
</tr>
<tr>
<td>96/19</td>
<td>M.A. Reniers</td>
<td>Static Semantics of Message Sequence Charts</td>
<td>71</td>
</tr>
<tr>
<td>96/20</td>
<td>L. Feijs</td>
<td>Algebraic Specification and Simulation of Lazy Functional Programs in a concurrent Environment</td>
<td>27</td>
</tr>
<tr>
<td>96/21</td>
<td>L. Bijlsma and R. Nederpelt</td>
<td>Predicate calculus: concepts and misconceptions</td>
<td>25</td>
</tr>
<tr>
<td>96/22</td>
<td>M.C.A. van de Graaf and G.J. Houben</td>
<td>Designing Effective Workflow Management Processes</td>
<td>22</td>
</tr>
<tr>
<td>96/23</td>
<td>W.M.P. van der Aalst</td>
<td>Structural Characterizations of sound workflow nets</td>
<td>22</td>
</tr>
<tr>
<td>96/24</td>
<td>M. Voorhoeve and W. van der Aalst</td>
<td>Conservative Adoption of Workflow</td>
<td>22</td>
</tr>
<tr>
<td>96/25</td>
<td>M. Vaccari and R.C. Backhouse</td>
<td>Deriving a systolic regular language recognizer</td>
<td>28</td>
</tr>
<tr>
<td>97/02</td>
<td>J. Hooman and O. v. Ruijsmalen</td>
<td>A Programming-Language Extension for Distributed Real-Time Systems</td>
<td>50</td>
</tr>
<tr>
<td>97/03</td>
<td>J. Blanco and A. v. Deursen</td>
<td>Basic Conditional Process Algebra</td>
<td>20</td>
</tr>
<tr>
<td>97/04</td>
<td>J.C.M. Baeten and J.A. Bergstra</td>
<td>Discrete Time Process Algebra: Absolute Time, Relative Time and Parametric Time</td>
<td>26</td>
</tr>
<tr>
<td>97/05</td>
<td>J.C.M. Baeten and J.J. Vereijken</td>
<td>Discrete-Time Process Algebra with Empty Process</td>
<td>51</td>
</tr>
<tr>
<td>97/06</td>
<td>M. Fransen</td>
<td>Tools for the Construction of Correct Programs: an Overview</td>
<td>33</td>
</tr>
<tr>
<td>97/07</td>
<td>J.C.M. Baeten and J.A. Bergstra</td>
<td>Bounded Stacks, Bugs and Queues</td>
<td>15</td>
</tr>
<tr>
<td>97/08</td>
<td>P. Hoogendijk and R.C. Backhouse</td>
<td>When do datatypes commute?</td>
<td>35</td>
</tr>
<tr>
<td>Year</td>
<td>Title</td>
<td>Authors</td>
<td>ISBN</td>
</tr>
<tr>
<td>------</td>
<td>----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>97/11</td>
<td>Distributed real-time systems: a survey of applications and a general design model, p. 31.</td>
<td>D. Hauschildt, E. Verbeek and W. van der Aalst</td>
<td></td>
</tr>
<tr>
<td>97/13</td>
<td>Exploring the Process Dimension of Workflow Management, p. 56.</td>
<td>W.M.P. van der Aalst</td>
<td></td>
</tr>
<tr>
<td>97/14</td>
<td>A computer checked algebraic verification of a distributed summation algorithm, p. 28.</td>
<td>E.H.L. Aarts, A. Engels, S. Mauw and M.A. Reniers</td>
<td></td>
</tr>
<tr>
<td>97/15</td>
<td>(\lambda P): A Pure Type System for First Order Logic with Automated Theorem Proving, p. 35.</td>
<td>M. Franssen, D. Hauschildt, E. Verbeek and W. van der Aalst</td>
<td></td>
</tr>
<tr>
<td>97/16</td>
<td>On the verification of Inter-organizational workflows, p. 23.</td>
<td>M. Vaccari and R.C. Backhouse</td>
<td></td>
</tr>
<tr>
<td>98/02</td>
<td>State / Event Net Equivalence, p. 25.</td>
<td>M. Voorhoeve</td>
<td></td>
</tr>
<tr>
<td>98/03</td>
<td>Deadlock Behaviour in Split and ST Bisimulation Semantics, p. 15.</td>
<td>J.C.M. Baaten and J.A. Bergstra</td>
<td></td>
</tr>
<tr>
<td>98/04</td>
<td>Pair Algebras and Galois Connections, p. 14.</td>
<td>R.C. Backhouse</td>
<td></td>
</tr>
<tr>
<td>98/05</td>
<td>Flat Fragments of CTL and CTL*: Separating the Expressive and Distinguishing Powers, P. 22.</td>
<td>D. Dams</td>
<td></td>
</tr>
<tr>
<td>98/10</td>
<td>Process Algebra in PVS</td>
<td>T. Basten and J. Hooman</td>
<td></td>
</tr>
<tr>
<td>98/11</td>
<td>The Proof-assistant Yarrow, p. 15.</td>
<td>J. Zwanenburg</td>
<td></td>
</tr>
<tr>
<td>98/13</td>
<td>Checking verifications of protocols and distributed systems by computer. Extended version of a tutorial at CONCUR’98, p. 27.</td>
<td>T. Verheoef (artikel volgt)</td>
<td></td>
</tr>
<tr>
<td>99/01</td>
<td>Structured Operational Semantics of (\chi), p. 27.</td>
<td>V. Bos and J.J.T. Kleijn</td>
<td></td>
</tr>
<tr>
<td>99/02</td>
<td>Diagnosing Workflow Processes using Woflan, p. 44.</td>
<td>H.M.W. Verbeek, T. Basten and W.M.P. van der Aalst</td>
<td></td>
</tr>
<tr>
<td>99/03</td>
<td>Final Dialgebras: From Categories to Allegories, p. 26.</td>
<td>R.C. Backhouse and P. Hoogendijk</td>
<td></td>
</tr>
<tr>
<td>99/04</td>
<td>Process Algebra with Interleaving Probabilistic Parallel Composition, p. 81.</td>
<td>S. Andoova</td>
<td></td>
</tr>
<tr>
<td>Year</td>
<td>Authors</td>
<td>Title and Details</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>99/06</td>
<td>T. Basten and W. v.d. Aalst</td>
<td>Inheritance of Workflows: An Approach to tackling problems related to change, p. 66</td>
<td></td>
</tr>
<tr>
<td>99/10</td>
<td>T.A.C. Willemse</td>
<td>The Analysis of a Conveyor Belt System, a case study in Hybrid Systems and timed μCRL, p. 44.</td>
<td></td>
</tr>
</tbody>
</table>