Real time process algebra with time-dependent conditions

J.C.M. Baeten [a], C.A. Middelburg [a,b,*]

[a] *Computing Science Department, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, Netherlands*

[b] *Department of Philosophy, Utrecht University, P.O. Box 80126, 3508 TC Utrecht, Netherlands*

Abstract

We add conditionals with time-dependent conditions to the real time process algebra with parametric timing from the framework of process algebras with timing presented by Baeten and Middelburg (Handbook of Process Algebra, Elsevier, 2001, Ch. 10). This extension facilitates flexible dependence of process behaviour on initialization time. We show that the conditions concerned generalize the conditions introduced earlier in a discrete time setting by Baeten and Bergstra (Formal Aspects of Computing 8 (1996) 188-208).

Key words: process algebra, real time, discrete time, absolute timing, relative timing, parametric timing, initialization, conditionals, time-dependent conditions.

1 Introduction

Algebraic concurrency theories such as ACP [1–3], CCS [4,5] and CSP [6,7] have been extended to deal with time-dependent behaviour in various ways. In Ref. [8], we presented results of a systematic study of some of the most important issues relevant to dealing with time-dependent behaviour of processes – viz. absolute versus relative timing, continuous versus discrete time scale, and separation versus combination of execution of actions and passage of time – in the setting of ACP. We presented real time and discrete time versions of ACP with both absolute timing and relative timing, starting with a new real time

* Corresponding author. Telephone: +31 40 247 5157, fax: +31 40 247 5361.
Email addresses: josb@win.tue.nl (J.C.M. Baeten), keesm@win.tue.nl (C.A. Middelburg).

version of ACP with absolute timing called ACP^{sat}. We demonstrated that ACP^{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. Integration provides for alternative composition over a continuum of alternatives; and initial abstraction, being reminiscent of λ-abstraction but specific to the case where the parameter is process initialization time, provides a way of forming processes with parametric timing. The extension with integration enables embedding of discrete time process algebras and the extension with initial abstraction enables embedding of process algebras with relative timing. We focussed on versions of ACP with timing where execution of actions and passage of time are separated, but explained how versions with time stamping of actions can be obtained.

The real time versions of ACP presented in Ref. [8], unlike those presented in Refs. [9,10], do not exclude the possibility of two or more actions to be performed consecutively at the same point in time. That is, they include urgent actions, similar to ATP [11] and the different versions of CCS with timing [12–14]. 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 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 Refs. [9,10], 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.

In this paper we extend ACP^{sat} extended with integration and initial abstraction further with conditionals in which the condition depends on time. The conditions concerned generalize the conditions introduced earlier in Ref. [15] to extend discrete time versions of ACP with conditionals in which the condition depends on time. The extension allows an interesting expansion property of processes with parametric timing, called time spectrum expansion, to be expressed. It is practically useful as well, because it facilitates flexible dependence of process behaviour on initialization time. We also extend the discrete time counterpart of ACP^{sat} presented in Ref. [8] with conditionals in which the condition depends on time. In this case, the conditions are essentially the same as the conditions introduced earlier in Ref. [15]. For all that, the emphasis of this paper is on a real time version with parametric timing that essentially encompasses all real time and discrete time versions of ACP with absolute timing and relative timing presented in Ref. [8].

In Ref. [8], our aim was to present a coherent collection of algebraic concurrency theories generalizing ACP that deal with time-dependent behaviour in different ways. In this paper, we extend the main real time and discrete time versions of ACP presented in Ref. [8] with conditionals in which the condition depends on time. By showing that the discrete time version with conditionals can be embedded in the real time version with conditionals, we demonstrate that the extensions with conditionals do not destroy the coherence.

We also give an example of the use of the presented version of $\mathrm{ACP}^{\mathrm{sat}}$ with conditionals. The example concerns the description of the behaviour that is relevant to railroad crossing control. We do not go into detail about the analysis of the described railroad crossing system, but we do mention some of the properties that can be checked using generalizations of the standard process algebraic techniques of linearization and expansion. Various standard process algebraic techniques for a detailed analysis of systems described using ACP-style process algebras, including linearization and expansion, can be generalized to the presented version of $\mathrm{ACP}^{\mathrm{sat}}$ with conditionals. However, a treatment of these techniques in the setting of this real time version of ACP is considered to go beyond the scope of this paper.

Various constants and operators of real time versions of ACP have counterparts in discrete time versions of ACP; and various constants and operators of versions of ACP with absolute timing have counterparts in versions of ACP with relative timing. A notational distinction is made between a constant or operator of one version and its counterparts in another version, by means of different decorations of a common symbol, if they should not be identified in case the versions are integrated. The embeddings of discrete time versions in real time versions (with integration) and the embeddings of versions with relative timing in versions with absolute timing (with initial abstraction) permit discrete time versions and real time versions to be integrated and versions with relative timing and versions with absolute timing to be integrated, respectively. We can, for example, describe a process as the parallel composition of a process described in a real time version with relative timing and a process described in a discrete time version with absolute timing. Of course, so long as one uses a single version, one can safely omit the above-mentioned decorations.

The structure of this paper is as follows. First, we review $\mathrm{ACP}^{\mathrm{sat}}$ and its extension with integration and initial abstraction in Sections 2. Then, in Section 3, we add conditionals in which the condition depends on time to this real time version of ACP. After that, in Section 4, we first briefly review the discrete time counterpart of $\mathrm{ACP}^{\mathrm{sat}}$ and then add conditionals in which the condition depends on time to this discrete time version of ACP. In Section 5, we show that the discrete time version with conditionals can be embedded in the real time version with conditionals.

2 Real time process algebra with absolute timing

In this section, we review $\mathrm{ACP}^{\mathrm{sat}}$, the real time process algebra with absolute timing introduced in Ref. [8], and its extension with integration and initial abstraction. A detailed account of this real time version of ACP and these extensions is given in Ref. [8]. The axioms and operational semantics rules – extracted from Ref. [8] – are given in Appendix A.

In case of $\mathrm{ACP}^{\mathrm{sat}}$, it is assumed that a theory of the non-negative real numbers has been given. Its signature has to include the constant $0 \colon \to \mathbb{R}_{\geq 0}$, the operator $+ \colon \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$, and the predicates $\leq \colon \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}$ and $= \colon \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}$. 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 $\mathrm{ACP}^{\mathrm{sat}}$, 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 $\gamma : \mathsf{A} \times \mathsf{A} \to \mathsf{A}$, has been given. The function γ is regarded to give the result of the synchronous execution of any two actions for which this is possible, and to be undefined otherwise. In $\mathrm{ACP}^{\mathrm{sat}}$, as in the other versions of ACP with timing presented in this paper, the term communication is used in the sense of synchronous communication: communication is considered to take place only when actions are performed synchronously. The weak restrictions on γ allow many kinds of communication between parallel processes to be modeled.

First, in Section 2.1, we treat $\mathrm{BPA}^{\mathrm{sat}}$, basic standard real time process algebra with absolute timing, in which parallelism and communication are not considered. After that, in Section 2.2, $\mathrm{BPA}^{\mathrm{sat}}$ is extended to $\mathrm{ACP}^{\mathrm{sat}}$ to deal with parallelism and communication as well. In Section 2.3, integration and initial abstraction are added to $\mathrm{ACP}^{\mathrm{sat}}$. Finally, some useful additional axioms, derivable for closed terms, and elimination results are given in Section 2.4.

2.1 Basic process algebra

In $\mathrm{BPA}^{\mathrm{sat}}$, we have the sort P of processes, the *urgent action* constants $\tilde{a} \colon \to \mathsf{P}$ (one for each $a \in \mathsf{A}$), the *urgent deadlock* constant $\tilde{\delta} \colon \to \mathsf{P}$, the *deadlocked process* constant $\dot{\delta} \colon \to \mathsf{P}$, the *alternative composition* operator $+ : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$, the *sequential composition* operator $\cdot : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$, the *absolute delay* operator $\sigma_{\mathsf{abs}} : \mathbb{R}_{\geq 0} \times \mathsf{P} \to \mathsf{P}$, the *absolute time-out* operator $\upsilon_{\mathsf{abs}} : \mathbb{R}_{\geq 0} \times \mathsf{P} \to \mathsf{P}$, and the *absolute initialization* operator $\overline{\upsilon}_{\mathsf{abs}} : \mathbb{R}_{\geq 0} \times \mathsf{P} \to \mathsf{P}$.

The process $\tilde{a}$ is only capable of performing action a, immediately followed by successful termination, at time 0. The process $\tilde{\delta}$, although existing at time 0, is incapable of doing anything. The process $\dot{\delta}$ stands a process that exhibits inconsistent timing at time 0. This process, which is called immediate deadlock in Refs. [8,15], can be viewed as a process that has already deadlocked at time 0. The process $\sigma^{p}_{\mathsf{abs}}(x)$ is the process x shifted in time by p. Thus, the process $\sigma^{p}_{\mathsf{abs}}(\tilde{\delta})$ is capable of idling from time 0 upto and including time p – and at time p it gets incapable of doing anything – whereas the process $\sigma^{p}_{\mathsf{abs}}(\dot{\delta})$ is only capable of idling from time 0 upto, but not including, 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 two auxiliary operators: υ_{abs} and $\overline{\upsilon}_{\mathsf{abs}}$. The process $\upsilon^{p}_{\mathsf{abs}}(x)$ is the part of x that starts to perform actions before time p. The process $\overline{\upsilon}^{p}_{\mathsf{abs}}(x)$ is the part of x that starts to perform actions at time p or later. The operator υ_{abs} makes it easy to capture the interaction of absolute delay with sequential composition in the axioms of BPA$^{\mathrm{sat}}$. The operator $\overline{\upsilon}_{\mathsf{abs}}$ is used to anticipate in the axioms of BPA$^{\mathrm{sat}}$ the addition of initial abstraction, by which a process cannot only to be started up at time 0 but also at other time points.

We assume that an infinite set of variables of sort P has been given. Given the signature of BPA$^{\mathrm{sat}}$, terms of BPA$^{\mathrm{sat}}$ are constructed in the usual way. We will in general 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, (ii) the operator $+$ has the lowest precedence, and (iii) all other 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$^{\mathrm{sat}}$ 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 $\sum_{i \in \mathcal{I}} t_i$ stands for $\dot{\delta}$ if $\mathcal{I} = \emptyset$.

We denote variables by $x, x', y, y', \ldots$. We use $a, a', b, b', \ldots$ to denote elements of $\mathsf{A} \cup \{\delta\}$ 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_δ for $\mathsf{A} \cup \{\delta\}$.

Axiom system

The axiom system of BPA$^{\mathrm{sat}}$ consists of the equations given in Table 1. For a discussion of the axioms of BPA$^{\mathrm{sat}}$, see Ref. [8]. The axioms concerning the interaction of absolute delay with sequential composition become easier to understand by realizing that for all closed BPA$^{\mathrm{sat}}$-terms t and for all $p > 0$

Table 1
Axioms of $\mathrm{BPA}^{\mathrm{sat}}$ ($a \in \mathsf{A}_\delta$, $p, q \geq 0$, $r > 0$)

$x + y = y + x$	A1	$\upsilon^p_{\mathsf{abs}}(\dot{\delta}) = \dot{\delta}$	SATO0
$(x + y) + z = x + (y + z)$	A2	$\upsilon^0_{\mathsf{abs}}(x) = \dot{\delta}$	SATO1
$x + x = x$	A3	$\upsilon^r_{\mathsf{abs}}(\tilde{a}) = \tilde{a}$	SATO2
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$\upsilon^{p+q}_{\mathsf{abs}}(\sigma^p_{\mathsf{abs}}(x)) = \sigma^p_{\mathsf{abs}}(\upsilon^q_{\mathsf{abs}}(x))$	SATO3
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\upsilon^p_{\mathsf{abs}}(x + y) = \upsilon^p_{\mathsf{abs}}(x) + \upsilon^p_{\mathsf{abs}}(y)$	SATO4
$x + \dot{\delta} = x$	A6ID	$\upsilon^p_{\mathsf{abs}}(x \cdot y) = \upsilon^p_{\mathsf{abs}}(x) \cdot y$	SATO5
$\dot{\delta} \cdot x = \dot{\delta}$	A7ID		
		$\overline{\upsilon}^0_{\mathsf{abs}}(\dot{\delta}) = \dot{\delta}$	SAI0a
$\sigma^0_{\mathsf{abs}}(x) = \overline{\upsilon}^0_{\mathsf{abs}}(x)$	SAT1	$\overline{\upsilon}^r_{\mathsf{abs}}(\dot{\delta}) = \sigma^r_{\mathsf{abs}}(\dot{\delta})$	SAI0b
$\sigma^p_{\mathsf{abs}}(\sigma^q_{\mathsf{abs}}(x)) = \sigma^{p+q}_{\mathsf{abs}}(x)$	SAT2	$\overline{\upsilon}^0_{\mathsf{abs}}(\tilde{a}) = \tilde{a}$	SAI1
$\sigma^p_{\mathsf{abs}}(x) + \sigma^p_{\mathsf{abs}}(y) = \sigma^p_{\mathsf{abs}}(x + y)$	SAT3	$\overline{\upsilon}^r_{\mathsf{abs}}(\tilde{a}) = \sigma^r_{\mathsf{abs}}(\dot{\delta})$	SAI2
$\sigma^p_{\mathsf{abs}}(x) \cdot \upsilon^p_{\mathsf{abs}}(y) = \sigma^p_{\mathsf{abs}}(x \cdot \dot{\delta})$	SAT4	$\overline{\upsilon}^{p+q}_{\mathsf{abs}}(\sigma^p_{\mathsf{abs}}(x)) = \sigma^p_{\mathsf{abs}}(\overline{\upsilon}^q_{\mathsf{abs}}(\overline{\upsilon}^0_{\mathsf{abs}}(x)))$	SAI3
$\sigma^p_{\mathsf{abs}}(x) \cdot (\upsilon^p_{\mathsf{abs}}(y) + \sigma^p_{\mathsf{abs}}(z)) = \sigma^p_{\mathsf{abs}}(x \cdot \overline{\upsilon}^0_{\mathsf{abs}}(z))$	SAT5	$\overline{\upsilon}^p_{\mathsf{abs}}(x + y) = \overline{\upsilon}^p_{\mathsf{abs}}(x) + \overline{\upsilon}^p_{\mathsf{abs}}(y)$	SAI4
$\sigma^p_{\mathsf{abs}}(\dot{\delta}) \cdot x = \sigma^p_{\mathsf{abs}}(\dot{\delta})$	SAT6	$\overline{\upsilon}^p_{\mathsf{abs}}(x \cdot y) = \overline{\upsilon}^p_{\mathsf{abs}}(x) \cdot y$	SAI5
$\tilde{a} + \tilde{\delta} = \tilde{a}$	A6SAa		
$\sigma^r_{\mathsf{abs}}(x) + \tilde{\delta} = \sigma^r_{\mathsf{abs}}(x)$	A6SAb		
$\tilde{\delta} \cdot x = \tilde{\delta}$	A7SA		

either $t = \upsilon^p_{\mathsf{abs}}(t)$ is derivable or there exists a closed term t' such that $t = \upsilon^p_{\mathsf{abs}}(t) + \sigma^p_{\mathsf{abs}}(t')$ is derivable. Besides, $\overline{\upsilon}^0_{\mathsf{abs}}(t) = t$ is derivable for all closed $\mathrm{BPA}^{\mathrm{sat}}$-terms t. The above-mentioned representation result for closed $\mathrm{BPA}^{\mathrm{sat}}$-terms is a corollary of the following two lemmas from Ref. [8], which are used there to shorten the calculations in the proof of an embedding theorem.

Lemma 1 *In* $\mathrm{BPA}^{\mathrm{sat}}$ *and* $\mathrm{ACP}^{\mathrm{sat}}$*, as well as in the further extensions with restricted integration and initial abstraction:*

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

Lemma 2 *In* $\mathrm{BPA}^{\mathrm{sat}}$ *and* $\mathrm{ACP}^{\mathrm{sat}}$*, as well as in the further extensions with restricted integration and initial abstraction, for each $p \in \mathbb{R}_{\geq 0}$ and each closed term t, there exists a closed term t' such that $\overline{\upsilon}^p_{\mathsf{abs}}(t) = \sigma^p_{\mathsf{abs}}(t')$ and $t' = \overline{\upsilon}^0_{\mathsf{abs}}(t')$.*

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 2 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.

Example 3 *We take* A *such that* $a, b, c \in \mathsf{A}$*. From the axioms of* $\mathrm{BPA}^{\mathrm{sat}}$*, we* *can, for example, derive the equations:*

$$\sigma^{5.1}_{\mathsf{abs}}(\tilde{a}) \cdot (\sigma^{5.4}_{\mathsf{abs}}(\tilde{b}) + \sigma^{4.9}_{\mathsf{abs}}(\tilde{c})) = \sigma^{5.1}_{\mathsf{abs}}(\tilde{a} \cdot \sigma^{0.3}_{\mathsf{abs}}(\tilde{b}))$$

$$\sigma^{5.1}_{\mathsf{abs}}(\tilde{a}) \cdot (\sigma^{5.4}_{\mathsf{abs}}(\tilde{b}) \cdot \sigma^{4.9}_{\mathsf{abs}}(\tilde{c})) = \sigma^{5.1}_{\mathsf{abs}}(\tilde{a} \cdot \sigma^{0.3}_{\mathsf{abs}}(\tilde{b} \cdot \dot{\delta}))$$

$$\upsilon^{5.3}_{\mathsf{abs}}(\sigma^{5.1}_{\mathsf{abs}}(\tilde{a}) + \sigma^{5.4}_{\mathsf{abs}}(\tilde{b})) = \sigma^{5.1}_{\mathsf{abs}}(\tilde{a}) + \sigma^{5.3}_{\mathsf{abs}}(\dot{\delta})$$

$$\overline{\upsilon}^{5.3}_{\mathsf{abs}}(\sigma^{5.1}_{\mathsf{abs}}(\tilde{a}) + \sigma^{5.4}_{\mathsf{abs}}(\tilde{b})) = \sigma^{5.4}_{\mathsf{abs}}(\tilde{b})$$

$$\overline{\upsilon}^{5.7}_{\mathsf{abs}}(\sigma^{5.1}_{\mathsf{abs}}(\tilde{a}) + \sigma^{5.4}_{\mathsf{abs}}(\tilde{b})) = \sigma^{5.7}_{\mathsf{abs}}(\dot{\delta})$$

Semantics

A *real time transition system* over A consists of a set of *states* S, a *root state* $\rho \in \mathsf{S}$ and four kinds of relations on states:

- a binary relation $\langle \_, p\rangle \xrightarrow{a} \langle \_, p\rangle$ for each $a \in \mathsf{A}$, $p \in \mathbb{R}_{\geq 0}$,
- a unary relation $\langle \_, p\rangle \xrightarrow{a} \langle \surd, p\rangle$ for each $a \in \mathsf{A}$, $p \in \mathbb{R}_{\geq 0}$,
- a binary relation $\langle \_, p\rangle \overset{r}{\longmapsto} \langle \_, q\rangle$ for each $r \in \mathbb{R}_{>0}$, $p, q \in \mathbb{R}_{\geq 0}$ where $q = p+r$,
- a unary relation $\langle \_, p\rangle\!\uparrow$ for each $p \in \mathbb{R}_{\geq 0}$;

satisfying

- if $\langle s, p\rangle \overset{r+r'}{\longmapsto} \langle s', q\rangle$, $r, r' > 0$, then there is a s'' such that $\langle s, p\rangle \overset{r}{\longmapsto} \langle s'', p+r\rangle$ and $\langle s'', p+r\rangle \overset{r'}{\longmapsto} \langle s', q\rangle$;
- if $\langle s, p\rangle \overset{r}{\longmapsto} \langle s'', p+r\rangle$ and $\langle s'', p+r\rangle \overset{r'}{\longmapsto} \langle s', q\rangle$, then $\langle s, p\rangle \overset{r+r'}{\longmapsto} \langle s', q\rangle$.

The four kinds of relations are called *action step*, *action termination*, *time step* and *deadlocked* relations, respectively. We write $\mathsf{RTTS}(\mathsf{A})$ for the set of all real time transition systems over A.

We shall associate a transition system in $\mathsf{RTTS}(\mathsf{A})$ with a closed term t of $\mathrm{BPA}^{\mathrm{sat}}$ by taking the set of closed terms of $\mathrm{BPA}^{\mathrm{sat}}$ as set of states and the closed term t as root state, and defining the action step, action termination, time step and deadlocked relations using rules in the style of Plotkin [16]. A semantics given in this way is called a structural operational semantics.

Notice that, by taking closed terms as states, the relations can be explained as follows:

- $\langle t, p\rangle \xrightarrow{a} \langle t', p\rangle$: process t is capable of first performing action a at time p and then proceeding as process t';
- $\langle t, p\rangle \xrightarrow{a} \langle \surd, p\rangle$: process t is capable of first performing action a at time p and then terminating successfully;
- $\langle t, p\rangle \overset{r}{\longmapsto} \langle t', q\rangle$: process t is capable of first idling from time p to time q and

then proceeding as process t';

- $\langle t, p\rangle \uparrow$: process t has already deadlocked at time p.

The rules for the operational semantics have the form $\frac{h_1,\ldots,h_m,s}{c_1,\ldots,c_n}$, where s is optional. They are to be read as "if h_1 and ... and h_m then c_1 and ... and c_n, provided s". The conclusions $c_1, \ldots, c_n$ are positive formulas of the form $\langle t, p\rangle \xrightarrow{a} \langle t', p\rangle$, $\langle t, p\rangle \xrightarrow{a} \langle \surd, p\rangle$, $\langle t, p\rangle \overset{r}{\mapsto} \langle t', q\rangle$ or $\langle t, p\rangle \uparrow$, where t and t' are open terms of BPA$^{\mathrm{sat}}$. The premises $h_1, \ldots, h_m$ are positive formulas of the above forms or negative formulas of the form $\neg(\langle t, p\rangle \uparrow)$. 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 structural operational semantics of BPA$^{\mathrm{sat}}$ is described by the rules given in Table A.1. For a discussion of some of the rules for the operational semantics of BPA$^{\mathrm{sat}}$, see Ref. [8]. On the basis of the rules for the operational semantics of BPA$^{\mathrm{sat}}$, the operators of BPA$^{\mathrm{sat}}$ can be directly defined on the set of real time transition systems in a straightforward way.

By identifying bisimilar processes we obtain our preferred model of BPA$^{\mathrm{sat}}$. 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 $\mathsf{RTTS}(\mathsf{A})$ is a symmetric binary relation R on the set of states S such that:

- 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 $R(s', t')$;
- if $R(s, t)$, then $\langle s, p\rangle \xrightarrow{a} \langle \surd, p\rangle$ iff $\langle t, p\rangle \xrightarrow{a} \langle \surd, p\rangle$;
- if $R(s, t)$ and $\langle s, p\rangle \overset{r}{\mapsto} \langle s', q\rangle$, then there is a t' such that $\langle t, p\rangle \overset{r}{\mapsto} \langle t', q\rangle$ and $R(s', t')$;
- if $R(s, t)$, then $\langle s, p\rangle \uparrow$ iff $\langle t, p\rangle \uparrow$.

We say that two states s and t are *bisimilar*, written $s \leftrightarrow t$, if there exists a bisimulation R such that $R(s, t)$.

Bisimulation equivalence is a congruence for the operators of BPA$^{\mathrm{sat}}$. For this reason, the operators of BPA$^{\mathrm{sat}}$ can be defined on the set of bisimulation equivalence classes. We can prove that this results in a model for BPA$^{\mathrm{sat}}$, i.e. all equations derivable in BPA$^{\mathrm{sat}}$ hold. In other words, the axioms of BPA$^{\mathrm{sat}}$ form a sound axiomatization for the model based on bisimulation equivalence classes. As in the case of the other axiomatizations presented in this paper, we leave it as an open problem whether the axioms of BPA$^{\mathrm{sat}}$ form a complete axiomatization for this model.

In ACP$^{\mathrm{sat}}$, we have, in addition to the constants and operators of BPA$^{\mathrm{sat}}$, the *parallel composition* operator $\| : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$, the *left merge* operator $\mathbin{\|\!\!\lfloor} : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$, the *communication merge* operator $| : \mathsf{P} \times \mathsf{P} \to \mathsf{P}$, the *encapsulation* operators $\partial_H : \mathsf{P} \to \mathsf{P}$ (for each $H \subseteq \mathsf{A}$), and the *absolute urgent initialization* operator $\nu_{\mathsf{abs}} : \mathsf{P} \to \mathsf{P}$.

The process $x \| y$ is the process that proceeds with the processes x and y in parallel. It may start to perform actions by (i) performing an action of x if x can do so before or at the ultimate time for y to start performing actions or to deadlock, (ii) performing an action of y if y can do so before or at the ultimate time for x to start performing actions or to deadlock or (iii) performing an action of x and an action of y synchronously if x and y can do so at the same time. Furthermore, we have the encapsulation operators ∂_H (one for each $H \subseteq \mathsf{A}$) which turns all urgent actions $\tilde{a}$, where $a \in H$, into $\tilde{\delta}$. As in ACP, we also have the auxiliary operators $\mathbin{\|\!\!\lfloor}$ and $|$ to get a finite axiomatization of the parallel composition operator. The processes $x \mathbin{\|\!\!\lfloor} y$ and $x \| y$ are the same except that $x \mathbin{\|\!\!\lfloor} y$ must start to perform actions by performing an action of x. The processes $x \mid y$ and $x \| y$ are the same except that $x \mid y$ must start to perform actions by performing an action of x and an action of y synchronously. In case of ACP$^{\mathrm{sat}}$, one additional auxiliary operator is used: ν_{abs}. The process $\nu_{\mathsf{abs}}(x)$ is the part of process x that starts to perform actions at time 0. The operator ν_{abs} makes it easy to capture the interaction of absolute delay with left merge and communication merge in the axioms of ACP$^{\mathrm{sat}}$. Notice that the process $\overline{\upsilon}^{p}_{\mathsf{abs}}(\upsilon^{q}_{\mathsf{abs}}(x))$ $(p \leq q)$ is the part of process x that starts to perform actions in the time interval $[p, q)$. Because the interval is always right open, the operator ν_{abs} cannot be defined in terms of the operators υ_{abs} and $\overline{\upsilon}_{\mathsf{abs}}$. Changing the operator υ_{abs} such that the interval becomes right closed, would make the operator useless to capture the interaction of absolute delay with sequential composition in the axioms of ACP$^{\mathrm{sat}}$.

Axiom system

The axiom system of ACP$^{\mathrm{sat}}$ consists of the axioms of BPA$^{\mathrm{sat}}$ and the equations given in Table 2. For a discussion of the axioms of ACP$^{\mathrm{sat}}$, see Ref. [8]. The axioms concerning the interaction of absolute delay with left merge and communication merge become easier to understand by realizing that for all closed ACP$^{\mathrm{sat}}$-terms t either $t = \dot{\delta}$ is derivable or $t = \nu_{\mathsf{abs}}(t) + \tilde{\delta}$ is derivable or there exists a $p > 0$ and a closed term t' such that $t = \upsilon^{p}_{\mathsf{abs}}(t) + \sigma^{p}_{\mathsf{abs}}(t')$ is derivable.

Example 4 *We take* A *such that* $a, b, c, d \in \mathsf{A}$ *and* γ *such that* $\gamma(b, c) =$

Table 2
Additional axioms for $\mathrm{ACP^{sat}}$ $(a, b \in \mathsf{A}_\delta,\ c \in \mathsf{A},\ p \geq 0,\ r > 0)$

$x \parallel y = x \mathbin{⌊\!⌊} y + y \mathbin{⌊\!⌊} x + x \mid y$	CM1	$\tilde{a} \mid \tilde{b} = \tilde{c}$ if $\gamma(a,b) = c$	CF1SA
$\dot{\delta} \mathbin{⌊\!⌊} x = \dot{\delta}$	CMID1	$\tilde{a} \mid \tilde{b} = \tilde{\delta}$ if $\gamma(a,b)$ undefined	CF2SA
$x \mathbin{⌊\!⌊} \dot{\delta} = \dot{\delta}$	CMID2		
$\tilde{a} \mathbin{⌊\!⌊} (x + \tilde{\delta}) = \tilde{a} \cdot (x + \tilde{\delta})$	CM2SA	$\partial_H(\dot{\delta}) = \dot{\delta}$	D0
$\tilde{a} \cdot x \mathbin{⌊\!⌊} (y + \tilde{\delta}) = \tilde{a} \cdot (x \parallel (y + \tilde{\delta}))$	CM3SA	$\partial_H(\tilde{a}) = \tilde{a}$ if $a \notin H$	D1SA
$\sigma^r_{\mathsf{abs}}(x) \mathbin{⌊\!⌊} (\nu_{\mathsf{abs}}(y) + \tilde{\delta}) = \tilde{\delta}$	SACM1	$\partial_H(\tilde{a}) = \tilde{\delta}$ if $a \in H$	D2SA
$\sigma^p_{\mathsf{abs}}(x) \mathbin{⌊\!⌊} (v^p_{\mathsf{abs}}(y) + \sigma^p_{\mathsf{abs}}(z)) = \sigma^p_{\mathsf{abs}}(x \mathbin{⌊\!⌊} z)$	SACM2	$\partial_H(\sigma^p_{\mathsf{abs}}(x)) = \sigma^p_{\mathsf{abs}}(\partial_H(x))$	SAD
$(x + y) \mathbin{⌊\!⌊} z = x \mathbin{⌊\!⌊} z + y \mathbin{⌊\!⌊} z$	CM4	$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3
$\dot{\delta} \mid x = \dot{\delta}$	CMID3	$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4
$x \mid \dot{\delta} = \dot{\delta}$	CMID4		
$\tilde{a} \cdot x \mid \tilde{b} = (\tilde{a} \mid \tilde{b}) \cdot x$	CM5SA	$\nu_{\mathsf{abs}}(\dot{\delta}) = \dot{\delta}$	SAU0
$\tilde{a} \mid \tilde{b} \cdot x = (\tilde{a} \mid \tilde{b}) \cdot x$	CM6SA	$\nu_{\mathsf{abs}}(\tilde{a}) = \tilde{a}$	SAU1
$\tilde{a} \cdot x \mid \tilde{b} \cdot y = (\tilde{a} \mid \tilde{b}) \cdot (x \parallel y)$	CM7SA	$\nu_{\mathsf{abs}}(\sigma^r_{\mathsf{abs}}(x)) = \tilde{\delta}$	SAU2
$(\nu_{\mathsf{abs}}(x) + \tilde{\delta}) \mid \sigma^r_{\mathsf{abs}}(y) = \tilde{\delta}$	SACM3	$\nu_{\mathsf{abs}}(x + y) = \nu_{\mathsf{abs}}(x) + \nu_{\mathsf{abs}}(y)$	SAU3
$\sigma^r_{\mathsf{abs}}(x) \mid (\nu_{\mathsf{abs}}(y) + \tilde{\delta}) = \tilde{\delta}$	SACM4	$\nu_{\mathsf{abs}}(x \cdot y) = \nu_{\mathsf{abs}}(x) \cdot y$	SAU4
$\sigma^p_{\mathsf{abs}}(x) \mid \sigma^p_{\mathsf{abs}}(y) = \sigma^p_{\mathsf{abs}}(x \mid y)$	SACM5		
$(x + y) \mid z = x \mid z + y \mid z$	CM8		
$x \mid (y + z) = x \mid y + x \mid z$	CM9		

$\gamma(c, b) = d$ and γ is undefined otherwise. From the axioms of $\mathrm{ACP^{sat}}$*, we can, for example, derive the equations:*

$$(\tilde{a} + \sigma^{5.4}_{\mathsf{abs}}(\tilde{b})) \parallel \sigma^{4.9}_{\mathsf{abs}}(\tilde{c}) = \tilde{a} \cdot \sigma^{4.9}_{\mathsf{abs}}(\tilde{c}) + \sigma^{4.9}_{\mathsf{abs}}(\tilde{c} \cdot \sigma^{0.5}_{\mathsf{abs}}(\tilde{b}))$$
$$(\tilde{a} + \sigma^{4.9}_{\mathsf{abs}}(\tilde{b})) \parallel \sigma^{5.4}_{\mathsf{abs}}(\tilde{c}) = \tilde{a} \cdot \sigma^{5.4}_{\mathsf{abs}}(\tilde{c}) + \sigma^{4.9}_{\mathsf{abs}}(\tilde{b} \cdot \sigma^{0.5}_{\mathsf{abs}}(\tilde{c}))$$
$$\tilde{a} \cdot \sigma^{4.9}_{\mathsf{abs}}(\tilde{b}) \parallel \sigma^{4.9}_{\mathsf{abs}}(\tilde{c}) = \tilde{a} \cdot \sigma^{4.9}_{\mathsf{abs}}(\tilde{b} \cdot \tilde{c} + \tilde{c} \cdot \tilde{b} + \tilde{d})$$
$$\sigma^{4.9}_{\mathsf{abs}}(\tilde{a}) \cdot \tilde{b} \parallel \sigma^{4.9}_{\mathsf{abs}}(\tilde{c}) = \sigma^{4.9}_{\mathsf{abs}}(\tilde{a} \cdot \dot{\delta} + \tilde{c} \cdot \tilde{a} \cdot \dot{\delta})$$

Semantics

The structural operational semantics of $\mathrm{ACP^{sat}}$ is described by the rules for $\mathrm{BPA^{sat}}$ and the rules given in Table A.2. For a discussion of some of the additional rules for $\mathrm{ACP^{sat}}$, see Ref. [8]. Bisimulation equivalence is also a congruence for the additional operators of $\mathrm{ACP^{sat}}$. Therefore, these operators can be defined on the set of bisimulation equivalence classes as well. As in the case of $\mathrm{BPA^{sat}}$, we can prove that this results in a model for $\mathrm{ACP^{sat}}$.

2.3 Integration and initial abstraction

In this subsection, we review the extension of $\mathrm{ACP^{sat}}$ with integration and initial abstraction. The extension with integration enables embedding of discrete time process algebras and the extension with initial abstraction enables embedding of process algebras with relative timing, as exemplified in Ref. [8]. For embedding of discrete time process algebras only a restricted form of integration, known as prefix integration (see Ref. [17]), is needed. The usefulness of integration in practical applications of real time process algebra has been demonstrated in various case studies, see e.g. Refs. [18,19], but the usefulness of initial abstraction in practical applications has not been demonstrated yet.

Integration and initial abstraction are both variable binding operators. Following e.g. Refs. [20,21], we will introduce *variable binding operators* by a declaration of the form $f : S_{11}, \ldots, S_{1k_1} . S_1 \times \ldots \times S_{n1}, \ldots, S_{nk_n} . S_n \to S$. Hereby is indicated that f combines an operator $f^* : ((S_{11} \times \ldots \times S_{1k_1}) \to S_1) \times \ldots \times ((S_{n1} \times \ldots \times S_{nk_n}) \to S_n) \to S$ with λ-calculus-like functional abstraction, binding k_i variables ranging over $S_{i1}, \ldots, S_{ik_i}$ in the ith argument $(0 \leq i \leq n)$. Applications of f have the following form: $f(x_{11}, \ldots, x_{1k_1} . t_1, \ldots, x_{n1}, \ldots, x_{nk_n} . t_n)$, 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, $\mathrm{ACP}^{\mathrm{sat}}$ is extended with integration. After that, initial abstraction is added.

Integration

In $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}$, we have, in addition to the constants and operators of $\mathrm{ACP}^{\mathrm{sat}}$, the *integration* (variable-binding) operator $\textstyle\int : \mathcal{P}(\mathbb{R}_{\geq 0}) \times \mathbb{R}_{\geq 0} . \mathsf{P} \to \mathsf{P}$. The integration operator $\textstyle\int$ provides for alternative composition over a continuum of alternatives. That is, $\textstyle\int_{v \in V} P$, 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 $p \in V$. We use the notation $P[p/v]$ for the term P with all free occurrences of variable v replaced by p. Obviously, we could first have added integration to $\mathrm{BPA}^{\mathrm{sat}}$, resulting in $\mathrm{BPA}^{\mathrm{sat}}\mathrm{I}$, and then have extended $\mathrm{BPA}^{\mathrm{sat}}\mathrm{I}$ to deal with parallelism and communication.

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 $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}$ by $P, Q, \ldots$. We will use the following notational convention. We write $\textstyle\int_{v \in V} P$ for $\textstyle\int(V, v . P)$.

Table 3
Axioms for integration ($p \geq 0$, v not free in R)

$\int_{w \in V} R = \int_{v \in V} R[v/w]$	INT1	$\int_{v \in V} \sigma^p_{\mathsf{abs}}(P) = \sigma^p_{\mathsf{abs}}(\int_{v \in V} P)$	INT10
$\int_{v \in \emptyset} P = \dot{\delta}$	INT2	$\int_{v \in V}(P + Q) = \int_{v \in V} P + \int_{v \in V} Q$	INT11
$\int_{v \in \{p\}} P = P[p/v]$	INT3	$\int_{v \in V}(P \cdot R) = (\int_{v \in V} P) \cdot R$	INT12
$\int_{v \in V \cup W} P = \int_{v \in V} P + \int_{v \in W} P$	INT4	$\int_{v \in V}(P \mathbin{\|\!\lfloor} R) = (\int_{v \in V} P) \mathbin{\|\!\lfloor} R$	INT13
$V \neq \emptyset \Rightarrow \int_{v \in V} R = R$	INT5	$\int_{v \in V}(P \mid R) = (\int_{v \in V} P) \mid R$	INT14
$(\forall p \in V \bullet P[p/v] = Q[p/v]) \Rightarrow \int_{v \in V} P = \int_{v \in V} Q$	INT6	$\int_{v \in V}(R \mid P) = R \mid (\int_{v \in V} P)$	INT15
$V \neq \emptyset \Rightarrow \int_{v \in V} \sigma^v_{\mathsf{abs}}(\dot{\delta}) = \sigma^{\sup V}_{\mathsf{abs}}(\dot{\delta})$	INT7	$\int_{v \in V} \partial_H(P) = \partial_H(\int_{v \in V} P)$	INT16
$V \neq \emptyset, \sup V \notin V \Rightarrow \int_{v \in V} \sigma^v_{\mathsf{abs}}(\tilde{\delta}) = \sigma^{\sup V}_{\mathsf{abs}}(\dot{\delta})$	INT8	$\upsilon^p_{\mathsf{abs}}(\int_{v \in V} P) = \int_{v \in V} \upsilon^p_{\mathsf{abs}}(P)$	SATO6
$\sup V \in V \Rightarrow \int_{v \in V} \sigma^v_{\mathsf{abs}}(\tilde{\delta}) = \sigma^{\sup V}_{\mathsf{abs}}(\tilde{\delta})$	INT9	$\overline{\upsilon}^p_{\mathsf{abs}}(\int_{v \in V} P) = \int_{v \in V} \overline{\upsilon}^p_{\mathsf{abs}}(P)$	SAI6
		$\nu_{\mathsf{abs}}(\int_{v \in V} P) = \int_{v \in V} \nu_{\mathsf{abs}}(P)$	SAU5

Axiom system The axiom system of $\mathrm{ACP^{sat}I}$ consists of the axioms of $\mathrm{ACP^{sat}}$ and the equations given in Table 3. Axioms INT1-INT6 are the crucial axioms of integration. They reflect the informal explanation given above.

Example 5 *We take* A *such that* $a, b \in \mathsf{A}$. *From the axioms of* $\mathrm{ACP^{sat}I}$, *we can, for example, derive the equations:*

$$\int_{v \in [4.9,5.1)} \sigma^v_{\mathsf{abs}}(\tilde{a}) + \int_{v \in [4.9,5.4)} \sigma^v_{\mathsf{abs}}(\tilde{b}) = \int_{v \in [4.9,5.1)} \sigma^v_{\mathsf{abs}}(\tilde{a} + \int_{w \in [0,0.3)} \sigma^w_{\mathsf{abs}}(\tilde{b}))$$
$$(\int_{v \in [4.9,5.1)} \sigma^v_{\mathsf{abs}}(\tilde{a})) \cdot \int_{v \in [4.9,5.4)} \sigma^v_{\mathsf{abs}}(\tilde{b}) = \int_{v \in [4.9,5.1)} \sigma^v_{\mathsf{abs}}(\tilde{a} \cdot \int_{w \in [0,5.4-v)} \sigma^w_{\mathsf{abs}}(\tilde{b}))$$
$$(\int_{v \in [4.9,5.4)} \sigma^v_{\mathsf{abs}}(\tilde{a})) \cdot \int_{v \in [4.9,5.1)} \sigma^v_{\mathsf{abs}}(\tilde{b}) = \int_{v \in [4.9,5.1)} \sigma^v_{\mathsf{abs}}(\tilde{a} \cdot \int_{w \in [0,5.1-v)} \sigma^w_{\mathsf{abs}}(\tilde{b})) + \int_{v \in [5.1,5.4)} \sigma^v_{\mathsf{abs}}(\tilde{a} \cdot \dot{\delta})$$

Semantics The structural operational semantics of $\mathrm{ACP^{sat}I}$ is described by the rules for $\mathrm{ACP^{sat}}$ and the rules given in Table A.3. The rules for integration are simple generalizations of the rules for alternative composition to the infinite case. 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 $\mathrm{BPA^{sat}}$ and $\mathrm{ACP^{sat}}$, we can prove that this results in a model for $\mathrm{ACP^{sat}I}$. We will call this model $\mathsf{M_A}$.

For a formal treatment of structural operational semantics in the presence of variable binding operators, the reader is referred to Ref. [21].

Initial abstraction

In $\mathrm{ACP^{sat}I}\surd$, we have, in addition to the constants and operators of $\mathrm{ACP^{sat}I}$, the *initial abstraction* (variable-binding) operator $\surd_{\mathsf{s}} : \mathbb{R}_{\geq 0} \,.\, \mathsf{P}^* \to \mathsf{P}^*$. The sort P of processes with absolute timing is replaced in $\mathrm{ACP^{sat}I}\surd$ by the sort P^* of processes with parametric timing. The initial abstraction operator $\surd_{\mathsf{s}}$

Table 4
Axioms for standard initial abstraction ($p \geq 0$, v not free in G)

$\surd_{\mathsf{s}} w \,.\, G = \surd_{\mathsf{s}} v \,.\, G[v/w]$	SIA1	$v^{p}_{\mathsf{abs}}(\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, v^{p}_{\mathsf{abs}}(F)$	SIA10
$\overline{v}^{p}_{\mathsf{abs}}(\surd_{\mathsf{s}} v \,.\, F) = \overline{v}^{p}_{\mathsf{abs}}(F[p/v])$	SIA2	$(\surd_{\mathsf{s}} v \,.\, F) \lfloor\!\lfloor G = \surd_{\mathsf{s}} v \,.\, (F \lfloor\!\lfloor \overline{v}^{v}_{\mathsf{abs}}(G))$	SIA11
$\surd_{\mathsf{s}} v \,.\, (\surd_{\mathsf{s}} w \,.\, F) = \surd_{\mathsf{s}} v \,.\, F[v/w]$	SIA3	$G \lfloor\!\lfloor (\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, (\overline{v}^{v}_{\mathsf{abs}}(G) \lfloor\!\lfloor F)$	SIA12
$G = \surd_{\mathsf{s}} v \,.\, G$	SIA4	$(\surd_{\mathsf{s}} v \,.\, F) \mid G = \surd_{\mathsf{s}} v \,.\, (F \mid \overline{v}^{v}_{\mathsf{abs}}(G))$	SIA13
$(\forall p \in \mathbb{R}_{\geq 0} \bullet \overline{v}^{p}_{\mathsf{abs}}(x) = \overline{v}^{p}_{\mathsf{abs}}(y)) \Rightarrow x = y$	SIA5	$G \mid (\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, (\overline{v}^{v}_{\mathsf{abs}}(G) \mid F)$	SIA14
$\sigma^{p}_{\mathsf{abs}}(\tilde{a}) \cdot x = \sigma^{p}_{\mathsf{abs}}(\tilde{a}) \cdot \overline{v}^{p}_{\mathsf{abs}}(x)$	SIA6	$\partial_H(\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, \partial_H(F)$	SIA15
$\sigma^{p}_{\mathsf{abs}}(\surd_{\mathsf{s}} v \,.\, F) = \sigma^{p}_{\mathsf{abs}}(F[0/v])$	SIA7	$\nu_{\mathsf{abs}}(\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, \nu_{\mathsf{abs}}(F)$	SIA16
$(\surd_{\mathsf{s}} v \,.\, F) + G = \surd_{\mathsf{s}} v \,.\, (F + \overline{v}^{v}_{\mathsf{abs}}(G))$	SIA8	$\int_{v \in V}(\surd_{\mathsf{s}} w \,.\, F) = \surd_{\mathsf{s}} w \,.\, (\int_{v \in V} F)$ if $v \neq w$	SIA17
$(\surd_{\mathsf{s}} v \,.\, F) \cdot G = \surd_{\mathsf{s}} v \,.\, (F \cdot G)$	SIA9		

provides the primary way of forming processes with parametric timing. The operators of $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}$ can simply be lifted to processes with parametric timing. The behaviour of processes with parametric timing depends on the time of initialization. They can be perceived as functions from non-negative real numbers to processes with absolute timing that map each non-negative real number p to a process with absolute timing that is initialized at time p. Initial abstraction is an abstraction mechanism to form such functions. It is reminiscent of λ-abstraction, but specific to the case where the parameter is process initialization time. That is, $\surd_{\mathsf{s}} v \,.\, F$, 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 $p \in \mathbb{R}_{\geq 0}$. Of course, it is also possible to add the initial abstraction operator to $\mathrm{ACP}^{\mathrm{sat}}$, resulting in a theory $\mathrm{ACP}^{\mathrm{sat}}\surd$.

We now use $x, y, \ldots$ to denote variables of sort P^*. Terms of $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}\surd$ are denoted by $F, G, \ldots$. We will use the following notational convention. We write $\surd_{\mathsf{s}} v \,.\, F$ for $\surd_{\mathsf{s}}(v \,.\, F)$.

Axiom system The axiom system of $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}\surd$ consists of the axioms of $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}$ and the equations given in Table 4. Axioms SIA1-SIA6 are the crucial axioms of initial abstraction. Axioms SIA1 and SIA2 are similar to the α- and β-conversion rules of λ-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. The remaining axioms concern the lifting of the operators of $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}$ to processes with parametric timing.

Example 6 *We take* A *such that* $a \in \mathsf{A}$. *From the axioms of* $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}\surd$, *we* *can, for example, derive the equations:*

$$\surd_{\mathsf{s}} v \,.\, \upsilon_{\mathsf{abs}}^{v+2.3}(\surd_{\mathsf{s}} w \,.\, \sigma_{\mathsf{abs}}^{w+1.2}(\tilde{a})) = \surd_{\mathsf{s}} v \,.\, \sigma_{\mathsf{abs}}^{v+1.2}(\tilde{a})$$
$$\surd_{\mathsf{s}} v \,.\, \overline{\upsilon}_{\mathsf{abs}}^{v+2.3}(\surd_{\mathsf{s}} w \,.\, \sigma_{\mathsf{abs}}^{w+1.2}(\tilde{a})) = \surd_{\mathsf{s}} v \,.\, \sigma_{\mathsf{abs}}^{v+3.5}(\tilde{a})$$
$$\overline{\upsilon}_{\mathsf{abs}}^{3.9}(\surd_{\mathsf{s}} v \,.\, \overline{\upsilon}_{\mathsf{abs}}^{v+2.3}(\textstyle\int_{w\in[4.8,4.9)}\sigma_{\mathsf{abs}}^{w+1.2}(\tilde{a}))) = \surd_{\mathsf{s}} v \,.\, \sigma_{\mathsf{abs}}^{6.2}(\dot{\delta})$$
$$\overline{\upsilon}_{\mathsf{abs}}^{3.6}(\surd_{\mathsf{s}} v \,.\, \overline{\upsilon}_{\mathsf{abs}}^{v+2.3}(\textstyle\int_{w\in[4.8,4.9)}\sigma_{\mathsf{abs}}^{w+1.2}(\tilde{a}))) = \surd_{\mathsf{s}} v \,.\, \textstyle\int_{w\in[6,6.1)}\sigma_{\mathsf{abs}}^{w}(\tilde{a})$$

Because of axiom SIA4, the right-hand sides of the third and fourth equation can be simplified further to $\sigma_{\mathsf{abs}}^{6.2}(\dot{\delta})$ *and* $\int_{w\in[6,6.1)}\sigma_{\mathsf{abs}}^{w}(\tilde{a})$*, respectively.*

Semantics On the basis of the rules for the operational semantics of $\mathrm{ACP^{sat}I}$, all operators of $\mathrm{ACP^{sat}I}$ can be directly defined on real time transition systems in a straightforward way. We will now describe a model of $\mathrm{ACP^{sat}I}\surd$ in terms of these operators.

We have to extend $\mathsf{RTTS}(\mathsf{A})$ to the function space

$$\mathsf{RTTS}^*(\mathsf{A}) = \{f : \mathbb{R}_{\geq 0} \to \mathsf{RTTS}(\mathsf{A}) \mid \forall p \in \mathbb{R}_{\geq 0} \bullet f(p) = \overline{\upsilon}_{\mathsf{abs}}^{p}(f(p))\}$$

of *real time transition systems with parametric timing*. In Table A.4, the constants and operators of $\mathrm{ACP^{sat}I}\surd$ are defined on $\mathsf{RTTS}^*(\mathsf{A})$.

We say that $f, g \in \mathsf{RTTS}^*(\mathsf{A})$ are bisimilar if for all $p \in \mathbb{R}_{\geq 0}$, the root states of $f(p)$ and $g(p)$ are bisimilar.

We obtain a model of $\mathrm{ACP^{sat}I}\surd$ by defining all operators on the set of bisimulation equivalence classes. We will call this model $\mathsf{M}_{\mathsf{A}}^*$. Notice that $f \in \mathsf{RTTS}^*(\mathsf{A})$ corresponds to a process that can be written with only the constants and operators of $\mathrm{ACP^{sat}I}$ iff $\overline{\upsilon}_{\mathsf{abs}}^{0}(f) = f$. In fact, M_{A} is isomorphic to a subalgebra of the reduct of $\mathsf{M}_{\mathsf{A}}^*$ that leaves out initial abstraction.

2.4 Miscellaneous

Standard initialization axioms

In Table 5, some equations concerning initialization and time-out are given that hold in the model $\mathsf{M}_{\mathsf{A}}^*$, and that are derivable for closed terms of $\mathrm{ACP^{sat}I}\surd$. We will use these axioms in proofs. Using the standard initialization axioms, the following can easily be derived for all terms F and F':

$$(\surd_{\mathsf{s}} v \,.\, F) \,\Box\, (\surd_{\mathsf{s}} v \,.\, F') = \surd_{\mathsf{s}} v \,.\, (F \,\Box\, F') \quad \mathrm{DISTR}\Box$$

Table 5
Standard initialization axioms ($p, q, q' \geq 0$, $r > 0$)

$\overline{\upsilon}^{p}_{\mathsf{abs}}(\upsilon^{p+r}_{\mathsf{abs}}(x)) = \upsilon^{p+r}_{\mathsf{abs}}(\overline{\upsilon}^{p}_{\mathsf{abs}}(x))$	SI1	$\overline{\upsilon}^{p}_{\mathsf{abs}}(x \mathbin{⌊\!⌊} y) = \overline{\upsilon}^{p}_{\mathsf{abs}}(x) \mathbin{⌊\!⌊} \overline{\upsilon}^{p}_{\mathsf{abs}}(y)$	SI10
$\overline{\upsilon}^{p}_{\mathsf{abs}}(\overline{\upsilon}^{p+q}_{\mathsf{abs}}(x)) = \overline{\upsilon}^{p+q}_{\mathsf{abs}}(x)$	SI2	$\overline{\upsilon}^{p}_{\mathsf{abs}}(x \mid y) = \overline{\upsilon}^{p}_{\mathsf{abs}}(x) \mid \overline{\upsilon}^{p}_{\mathsf{abs}}(y)$	SI11
$\overline{\upsilon}^{p+q}_{\mathsf{abs}}(\upsilon^{p}_{\mathsf{abs}}(x)) = \sigma^{p+q}_{\mathsf{abs}}(\dot{\delta})$	SI3	$\overline{\upsilon}^{p}_{\mathsf{abs}}(\partial_H(x)) = \partial_H(\overline{\upsilon}^{p}_{\mathsf{abs}}(x))$	SI12
$\upsilon^{p}_{\mathsf{abs}}(\overline{\upsilon}^{p+q}_{\mathsf{abs}}(x)) = \sigma^{p}_{\mathsf{abs}}(\dot{\delta})$	SI4	$\overline{\upsilon}^{0}_{\mathsf{abs}}(\nu_{\mathsf{abs}}(x)) = \nu_{\mathsf{abs}}(\overline{\upsilon}^{0}_{\mathsf{abs}}(x))$	SI13
$\sigma^{p}_{\mathsf{abs}}(\dot{\delta}) + \overline{\upsilon}^{p}_{\mathsf{abs}}(x) = \overline{\upsilon}^{p}_{\mathsf{abs}}(x)$	SI5	$\overline{\upsilon}^{r}_{\mathsf{abs}}(\nu_{\mathsf{abs}}(x)) = \sigma^{r}_{\mathsf{abs}}(\dot{\delta})$	SI14
$\sigma^{p}_{\mathsf{abs}}(\tilde{\delta}) + \overline{\upsilon}^{p}_{\mathsf{abs}}(x + \tilde{\delta}) = \overline{\upsilon}^{p}_{\mathsf{abs}}(x + \tilde{\delta})$	SI6	$\nu_{\mathsf{abs}}(\overline{\upsilon}^{r}_{\mathsf{abs}}(x)) = \tilde{\delta}$	SI15
$\overline{\upsilon}^{r}_{\mathsf{abs}}(x) + \tilde{\delta} = \overline{\upsilon}^{r}_{\mathsf{abs}}(x)$	SI7	$\upsilon^{r}_{\mathsf{abs}}(\nu_{\mathsf{abs}}(x)) = \nu_{\mathsf{abs}}(x)$	SI16
$\upsilon^{p}_{\mathsf{abs}}(\upsilon^{q}_{\mathsf{abs}}(x)) = \upsilon^{\min p,q}_{\mathsf{abs}}(x)$	SI8	$\nu_{\mathsf{abs}}(\upsilon^{r}_{\mathsf{abs}}(x)) = \nu_{\mathsf{abs}}(x)$	SI17
$\overline{\upsilon}^{p}_{\mathsf{abs}}(\overline{\upsilon}^{q}_{\mathsf{abs}}(\overline{\upsilon}^{q'}_{\mathsf{abs}}(x))) = \overline{\upsilon}^{\max p,q}_{\mathsf{abs}}(\overline{\upsilon}^{q'}_{\mathsf{abs}}(x))$	SI9		

for $\square = +, \|, ⌊\!⌊, |$. In other words, initial abstraction distributes over $+$, $\|$, $⌊\!⌊$ and $|$. This fact is a useful aid to shorten the calculations needed in proofs.

Elimination results

We can prove that the auxiliary operators υ_{abs} and $\overline{\upsilon}_{\mathsf{abs}}$, as well as sequential compositions in which the form of the first operand is not $\tilde{a}$ ($a \in \mathsf{A}$) and alternative compositions in which the form of the first operand is $\sigma^{p}_{\mathsf{abs}}(t)$, can be eliminated in closed terms of BPA$^{\mathrm{sat}}$I with a restricted form of integration. Basically, this restriction means that in terms of the form $\int_{v \in V} P$, V is an interval of which the bounds are given by linear expressions over time variables and P is of the form $\sigma^{v}_{\mathsf{abs}}(\tilde{a})$ or $\sigma^{v}_{\mathsf{abs}}(\tilde{a}) \cdot t$ ($a \in \mathsf{A}_{\delta}$). This restricted form of integration is essentially the same as prefix integration from Ref. [17]. The terms that remain after exhaustive elimination are called the basic terms over BPA$^{\mathrm{sat}}$ with restricted integration. We can also prove that the operators $\|$, $⌊\!⌊$, $|$, ∂_H and ν_{abs} can be eliminated in closed terms of ACP$^{\mathrm{sat}}$ with restricted integration. Because of these elimination results, we are permitted to use induction on the structure of basic terms over BPA$^{\mathrm{sat}}$ with restricted integration to prove statements for all closed terms of ACP$^{\mathrm{sat}}$ with restricted integration. The right-hand sides of the equations in Examples 3, 4 and 5 are all basic terms over BPA$^{\mathrm{sat}}$ with restricted integration.

The elimination results for ACP$^{\mathrm{sat}}\surd$ with restricted integration are essentially the same as the ones for ACP$^{\mathrm{sat}}$ with restricted integration. Besides, all closed terms of ACP$^{\mathrm{sat}}\surd$ with restricted integration can be written in the form $\surd_{\mathsf{s}} v \,.\, F$ where F is a basic term over BPA$^{\mathrm{sat}}$ with restricted integration. The right-hand sides of the equations in Example 6 are all of this form.

3 Conditionals with time-dependent conditions

In this section, we add a conditional operator with time-dependent conditions to $\mathrm{ACP^{sat}I}\surd$. This operator facilitates flexible dependence of process behaviour on initialization time. The time-dependent conditions introduced here generalize the time-dependent conditions introduced in a discrete time setting in Ref. [15]. First, in Section 3.1, $\mathrm{ACP^{sat}I}\surd$ is extended with time-dependent conditions and conditionals. After that, in Section 3.2, we describe a similar extension of $\mathrm{ACP^{sat}I}$ and explain how it is related to the extension of $\mathrm{ACP^{sat}I}\surd$. We treat the extension of $\mathrm{ACP^{sat}I}\surd$ first because it is semantically simpler to add a conditional operator with time-dependent conditions to $\mathrm{ACP^{sat}I}\surd$. In Section 3.4, we give an example of the use of conditionals with time-dependent conditions. In Section 3.3, we describe the addition of recursion in outline to make understanding of the specifications given in Section 3.4 easier.

3.1 Parametric timing

We first introduce time-dependent conditions. We have the sort $\mathbb{B}^*$ of time-dependent conditions, the *at time point* operator $\mathsf{pt}{:}\mathbb{R} \to \mathbb{B}^*$, the *at time point greater than* operator $\mathsf{pt}_{>}{:}\mathbb{R} \to \mathbb{B}^*$ (for technical reasons, it is convenient to use $\mathbb{R}$ instead of $\mathbb{R}_{\geq 0}$ as the domain of these functions), the *logical* constants and operators $\mathsf{t}{:} \to \mathbb{B}^*$, $\mathsf{f}{:} \to \mathbb{B}^*$, $\neg{:}\mathbb{B}^* \to \mathbb{B}^*$, $\vee{:}\mathbb{B}^* \times \mathbb{B}^* \to \mathbb{B}^*$ and $\wedge{:}\mathbb{B}^* \times \mathbb{B}^* \to \mathbb{B}^*$, the *initialization* operator $\overline{\upsilon}_{\mathsf{abs}} : \mathbb{R}_{\geq 0} \times \mathbb{B}^* \to \mathbb{B}^*$, and the *initial abstraction* operator $\surd_{\mathsf{s}} : \mathbb{R}_{\geq 0} \,.\, \mathbb{B}^* \to \mathbb{B}^*$.

For a time-dependent condition b, $\overline{\upsilon}^{p}_{\mathsf{abs}}(b)$ is either t or f, determined by whether b holds at time point p or not. For $p \in \mathbb{R}_{\geq 0}$, the condition $\mathsf{pt}(p)$ holds only at time point p and the condition $\mathsf{pt}_{>}(p)$ holds at all time points greater than p. For $r \in \mathbb{R}_{>0}$, the condition $\mathsf{pt}(-r)$ never holds and the condition $\mathsf{pt}_{>}(-r)$ always holds – recall that all time points are in $\mathbb{R}_{\geq 0}$. The logical operators $\neg$, $\vee$ and $\wedge$ are defined on $\mathbb{B}^*$ pointwise. Initial abstraction for conditions is similar to initial abstraction for processes.

We join time-dependent conditions with parametric time processes by means of the conditional operator $::\!\to$. In $\mathrm{ACP^{sat}I}\surd\mathrm{C}$, we have, in addition to the above-mentioned constants and operators on $\mathbb{B}^*$, the constants and operators of $\mathrm{ACP^{sat}I}\surd$ and the *conditional* operator $::\!\to : \mathbb{B}^* \times \mathsf{P}^* \to \mathsf{P}^*$.

Initialized at a time point p where the condition b holds, the process $b ::\!\to x$ proceeds as the process x initialized at time point p; and initialized at a time point p where the condition b does not hold, it proceeds as the process $\dot{\delta}$ initialized at time point p.

Table 6
Axioms for logical operators

$\neg\mathsf{t} = \mathsf{f}$	BOOL1	$\mathsf{t} \vee b = \mathsf{t}$	BOOL4
$\neg\mathsf{f} = \mathsf{t}$	BOOL2	$\mathsf{f} \vee b = b$	BOOL5
$\neg\neg b = b$	BOOL3	$b \vee b' = b' \vee b$	BOOL6
		$b \wedge b' = \neg(\neg b \vee \neg b')$	BOOL7

Table 7
Axioms for conditions ($p, q \geq 0$, $r > 0$, v not free in D)

$\overline{v}^p_{\mathsf{abs}}(\mathsf{t}) = \mathsf{t}$	CSAI1	$\surd_{\mathsf{s}} w \,.\, D = \surd_{\mathsf{s}} v \,.\, D[v/w]$	CSIA1
$\overline{v}^p_{\mathsf{abs}}(\mathsf{f}) = \mathsf{f}$	CSAI2	$\overline{v}^p_{\mathsf{abs}}(\surd_{\mathsf{s}} v \,.\, C) = \overline{v}^p_{\mathsf{abs}}(C[p/v])$	CSIA2
$\overline{v}^p_{\mathsf{abs}}(\mathsf{pt}(p)) = \mathsf{t}$	CSAI3	$\surd_{\mathsf{s}} v \,.\, (\surd_{\mathsf{s}} w \,.\, C) = \surd_{\mathsf{s}} v \,.\, C[v/w]$	CSIA3
$\overline{v}^p_{\mathsf{abs}}(\mathsf{pt}(p - r)) = \mathsf{f}$	CSAI4	$D = \surd_{\mathsf{s}} v \,.\, D$	CSIA4
$\overline{v}^p_{\mathsf{abs}}(\mathsf{pt}(p + r)) = \mathsf{f}$	CSAI5	$(\forall p \in \mathbb{R}_{\geq 0} \bullet \overline{v}^p_{\mathsf{abs}}(b) = \overline{v}^p_{\mathsf{abs}}(b')) \Rightarrow b = b'$	CSIA5
$\overline{v}^p_{\mathsf{abs}}(\mathsf{pt}_{>}(p - r)) = \mathsf{t}$	CSAI6	$\neg(\surd_{\mathsf{s}} v \,.\, C) = \surd_{\mathsf{s}} v \,.\, \neg C$	CSIA6
$\overline{v}^p_{\mathsf{abs}}(\mathsf{pt}_{>}(p + q)) = \mathsf{f}$	CSAI7	$(\surd_{\mathsf{s}} v \,.\, C) \wedge D = \surd_{\mathsf{s}} v \,.\, (C \wedge \overline{v}^v_{\mathsf{abs}}(D))$	CSIA7
$\overline{v}^p_{\mathsf{abs}}(\neg b) = \neg\overline{v}^p_{\mathsf{abs}}(b)$	CSAI8	$(\surd_{\mathsf{s}} v \,.\, C) \vee D = \surd_{\mathsf{s}} v \,.\, (C \vee \overline{v}^v_{\mathsf{abs}}(D))$	CSIA8
$\overline{v}^p_{\mathsf{abs}}(b \wedge b') = \overline{v}^p_{\mathsf{abs}}(b) \wedge \overline{v}^p_{\mathsf{abs}}(b')$	CSAI9		
$\overline{v}^p_{\mathsf{abs}}(b \vee b') = \overline{v}^p_{\mathsf{abs}}(b) \vee \overline{v}^p_{\mathsf{abs}}(b')$	CSAI10		

We write $b, b', \ldots$ to denote variables of sort $\mathbb{B}^*$. Terms of sort $\mathbb{B}^*$ are denoted by $C, D, \ldots$. We will use the following abbreviations. We write $\mathsf{pt}_{\geq}(p)$ for $\mathsf{pt}_{>}(p) \vee \mathsf{pt}(p)$, $\mathsf{pt}_{\leq}(p)$ for $\neg\mathsf{pt}_{>}(p)$ and $\mathsf{pt}_{<}(p)$ for $\neg\mathsf{pt}_{\geq}(p)$. We further write $\surd_{\mathsf{s}} v \,.\, C$ for $\surd_{\mathsf{s}}(v \,.\, C)$.

Axiom system

The axiom system of ACP$^{\mathrm{sat}}$I$\surd$C consists of the axioms of ACP$^{\mathrm{sat}}$I$\surd$ and the equations given in Tables 6, 7 and 8. Axioms CSAI1-CSAI10 (Table 7) reflect the intended meaning of the initialization operator on conditions, viz. evaluation at initialization time, clearly. Axioms CSIA1-CSIA8 (Table 7) closely resemble the axioms for initial abstraction of processes. Axioms SCG1, SCG2ID, SASGC1 and SASGC2 from Table 8 are the crucial axioms of conditionals. Axioms SCG1, SCG2ID and SASGC1 reflect the informal explanation of the conditional operator given above. Axiom SASGC2, also called the *time spectrum expansion* axiom, indicates that a parametric time process can be regarded as including a separate alternative for each initialization time. These alternatives are expressed by terms of the form $\mathsf{pt}(v) :\!\!:\!\!\rightarrow \overline{v}^v_{\mathsf{abs}}(x)$. The important point here is that $\overline{v}^v_{\mathsf{abs}}(x)$ is a process with absolute timing, i.e. it can be written with the constants and operators of ACP$^{\mathrm{sat}}$I only. Notice further that axiom SASGC2 could not be expressed in an extension of ACP$^{\mathrm{sat}}$ without integration. Axiom SASGC3 shows that checking whether a condition holds at initialization time can safely be postponed till after an initial delay provided that it does not matter that, if the condition does not hold at initialization

Table 8
Axioms for conditionals ($p \geq 0$, v not free in D and G)

$\mathsf{t} :\to x = x$	SGC1
$\mathsf{f} :\to x = \dot{\delta}$	SGC2ID
$\overline{v}^p_{\mathsf{abs}}(b :\to x) = \overline{v}^p_{\mathsf{abs}}(b) :\to \overline{v}^p_{\mathsf{abs}}(x) + \sigma^p_{\mathsf{abs}}(\dot{\delta})$	SASGC1
$x = \int_{v \in [0,p]} (\mathsf{pt}(v) :\to \overline{v}^v_{\mathsf{abs}}(x)) + \mathsf{pt}_{>}(p) :\to x$	SASGC2
$b :\to \dot{\delta} = \dot{\delta}$	SGC3ID
$b :\to \sigma^p_{\mathsf{abs}}(x) + \sigma^p_{\mathsf{abs}}(\dot{\delta}) = \surd_{\mathsf{s}} v \,.\, \sigma^p_{\mathsf{abs}}(\overline{v}^v_{\mathsf{abs}}(b) :\to x)$	SASGC3
$b :\to (x + y) = b :\to x + b :\to y$	SGC4
$b :\to x \cdot y = (b :\to x) \cdot y$	SGC5
$(b \vee b') :\to x = b :\to x + b' :\to x$	SGC6
$b :\to (b' :\to x) = (b \wedge b') :\to x$	SGC7
$b :\to v^p_{\mathsf{abs}}(x) = v^p_{\mathsf{abs}}(b :\to x)$	SASGC4
$b :\to (x \mathbin{\|\!\!\lfloor} y) = (b :\to x) \mathbin{\|\!\!\lfloor} (b :\to y)$	SASGC5
$b :\to (x \mid y) = (b :\to x) \mid (b :\to y)$	SASGC6
$b :\to \partial_H(x) = \partial_H(b :\to x)$	SASGC7
$b :\to \nu_{\mathsf{abs}}(x) = \nu_{\mathsf{abs}}(b :\to x)$	SASGC8
$D :\to (\int_{v \in V} P) = \int_{v \in V} (D :\to P)$	SASGC9
$D :\to (\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, (\overline{v}^v_{\mathsf{abs}}(D) :\to F)$	SASGC10
$(\surd_{\mathsf{s}} v \,.\, C) :\to G = \surd_{\mathsf{s}} v \,.\, (C :\to \overline{v}^v_{\mathsf{abs}}(G))$	SASGC11

time, deadlock will have occurred after the initial delay.

Example 7 *We take* A *such that* $a, b \in \mathsf{A}$. *From the axioms of* ACP$^{\text{sat}}$I$\surd$C, *we can, for example, derive the equation:*

$$\begin{aligned}\surd_{\mathsf{s}} v \,.\, (\sigma^{v+1.2}_{\mathsf{abs}}(\tilde{a}) \parallel \sigma^{4}_{\mathsf{abs}}(\tilde{b})) = {} & \int_{v \in [0,2.8]} (\mathsf{pt}(v) :\to \sigma^{v+1.2}_{\mathsf{abs}}(\tilde{a}) \cdot \sigma^{4}_{\mathsf{abs}}(\tilde{b})) + \\ & \int_{v \in [2.8,4]} (\mathsf{pt}(v) :\to \sigma^{4}_{\mathsf{abs}}(\tilde{b}) \cdot \sigma^{v+1.2}_{\mathsf{abs}}(\tilde{a})) + \\ & \mathsf{pt}(2.8) :\to \sigma^{4}_{\mathsf{abs}}(\tilde{a} \mid \tilde{b})\end{aligned}$$

In addition to the axioms needed for the expansion of parallel composition, the time spectrum expansion axiom is important in the derivation of this equation. The second alternative of the right-hand side of that axiom can be eliminated here: it is easy to show, using the extensionality axiom for processes with parametric timing, that this alternative equals $\dot{\delta}$.

It is easy to check that Lemmas 1 and 2 from Section 2.1 go through for the extension with conditionals.

Semantics

First of all, we need the structural operational semantics of ACP$^{\text{sat}}$I$\surd$ extended with a restricted form of conditionals, viz. conditionals where the condition is either t or f. It is described by the rules for ACP$^{\text{sat}}$I$\surd$ and the rules given in

Table 9
Rules for conditionals ($a \in \mathsf{A}$, $r > 0$, $p \geq 0$)

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle}{\langle \mathsf{t} :\!:\!\rightarrow x, p\rangle \xrightarrow{a} \langle x', p\rangle} \quad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle}{\langle \mathsf{t} :\!:\!\rightarrow x, p\rangle \xrightarrow{a} \langle \surd, p\rangle}$$

$$\frac{\langle x, p\rangle \overset{r}{\longmapsto} \langle x, p+r\rangle}{\langle \mathsf{t} :\!:\!\rightarrow x, p\rangle \overset{r}{\longmapsto} \langle \mathsf{t} :\!:\!\rightarrow x, p+r\rangle} \quad \frac{\langle x, p\rangle\uparrow}{\langle \mathsf{t} :\!:\!\rightarrow x, p\rangle\uparrow} \quad \frac{}{\langle \mathsf{f} :\!:\!\rightarrow x, p\rangle\uparrow}$$

Table 10
Definition of conditional operator on RTTS^* ($p \in \mathbb{R}_{\geq 0}$, $s \in \mathbb{R}$)

$c :\!:\!\rightarrow f = \lambda t \,.\, (c(t) :\!:\!\rightarrow f(t))$	$\neg c = \lambda t \,.\, \neg(c(t))$
$\mathsf{t} = \lambda t \,.\, \mathsf{t}$	$c \wedge d = \lambda t \,.\, (c(t) \wedge d(t))$
$\mathsf{f} = \lambda t \,.\, \mathsf{f}$	$c \vee d = \lambda t \,.\, (c(t) \vee d(t))$
$\mathsf{pt}(s) = \lambda t \,.\, (\mathsf{if}\ t = s\ \mathsf{then\ t\ else\ f})$	$\overline{v}^{p}_{\mathsf{abs}}(c) = c(p)$
$\mathsf{pt}_{>}(s) = \lambda t \,.\, (\mathsf{if}\ t > s\ \mathsf{then\ t\ else\ f})$	$\surd_{\mathsf{s}}^{*}(\gamma) = \lambda t \,.\, \overline{v}^{t}_{\mathsf{abs}}(\gamma(t))$

Table 9. On the basis of these rules, the restricted conditional operator can also be directly defined on real time transition systems in a straightforward way. In Table 10, the conditional operator is defined on $\mathsf{RTTS}^*(\mathsf{A})$ in terms of this operator. Additionally, the operators introduced for conditions are defined on $\mathbb{B}^*$. We use f to denote elements of $\mathsf{RTTS}^*(\mathsf{A})$, c and d to denote elements of $\mathbb{B}^*$, and γ to denote elements of $\mathbb{R}_{\geq 0} \rightarrow \mathbb{B}^*$. We use λ-notation for functions, t is a variable ranging over $\mathbb{R}_{\geq 0}$. As in the case of $\mathrm{ACP^{sat}I}\surd$, we obtain a model by defining all operators on bisimulation equivalence classes.

Standard initialization axioms

The following equation concerning initialization of conditions holds in the model described above, and is derivable for closed terms of sort $\mathbb{B}^*$:

$$\overline{v}^{p}_{\mathsf{abs}}(\overline{v}^{q}_{\mathsf{abs}}(b)) = \overline{v}^{q}_{\mathsf{abs}}(b) \quad \mathrm{SI18}$$

We will use this axiom in proofs in subsequent sections.

3.2 Absolute timing

Conditions of the forms $\mathsf{pt}(p)$ and $\mathsf{pt}_{>}(p)$ make it possible to express time-dependent conditions without using initial abstraction. As a result, an extension of $\mathrm{ACP^{sat}I}$ similar to the extension of $\mathrm{ACP^{sat}I}\surd$ described in Section 3.1 is possible. This would not have been the case if we had taken conditions of the forms $v = p$ and $v > p$, where v is a variable ranging over $\mathbb{R}_{\geq 0}$, as basic conditions instead.

The signature and axioms of this extension of $\mathrm{ACP^{sat}I}$, called $\mathrm{ACP^{sat}IC}$, are as follows. The signature of ACP$^{\mathrm{sat}}$IC is simply the signature of ACP$^{\mathrm{sat}}$I$\surd$C without the initial abstraction operators for conditions and processes. The axioms of ACP$^{\mathrm{sat}}$IC consists of the axioms of ACP$^{\mathrm{sat}}$I, the equations given in Tables 7 and 8 except SASGC3, SASGC10 and SASGC11, and the following equation:

$$\overline{\upsilon}^{p}_{\mathsf{abs}}(b :\rightarrow \sigma^{q}_{\mathsf{abs}}(x) + \sigma^{q}_{\mathsf{abs}}(\dot{\delta})) = \overline{\upsilon}^{p}_{\mathsf{abs}}(\sigma^{q}_{\mathsf{abs}}(\overline{\upsilon}^{p}_{\mathsf{abs}}(b) :\rightarrow x)) \quad \text{SASGC3}'$$

Note that axiom SASGC3 can be replaced by axiom SASGC3′ in ACP$^{\mathrm{sat}}$I$\surd$C as well; it follows immediately from axiom SIA5.

We treated ACP$^{\mathrm{sat}}$I$\surd$C first, despite the fact that it is a conservative extension of ACP$^{\mathrm{sat}}$IC. The reason is that semantically the conditionals with time-dependent conditions are simpler to deal with in case of ACP$^{\mathrm{sat}}$I$\surd$C. A model of ACP$^{\mathrm{sat}}$IC can be obtained from the model of ACP$^{\mathrm{sat}}$I$\surd$C presented in Section 3.1 by taking a subalgebra of the reduct that leaves out initial abstraction, viz. the subalgebra of bisimulation equivalence classes of $f \in \mathsf{RTTS}^*(\mathsf{A})$ for which $\overline{\upsilon}^{0}_{\mathsf{abs}}(f) = f$. An isomorphic model can be obtained by using the variant of real time transition systems described below.

A *real time transition system with initialization times* over A consists of a set of *states* S, a *root state* $\rho \in \mathsf{S}$ and four kinds of relations on states:

- a binary relation $\langle \_, p\rangle \xrightarrow{a}_{p'} \langle \_, p\rangle$ for each $a \in \mathsf{A}$, $p, p' \in \mathbb{R}_{\geq 0}$ where $p' \leq p$,
- a unary relation $\langle \_, p\rangle \xrightarrow{a}_{p'} \langle \surd, p\rangle$ for each $a \in \mathsf{A}$, $p, p' \in \mathbb{R}_{\geq 0}$ where $p' \leq p$,
- a binary relation $\langle \_, p\rangle \overset{r}{\longmapsto}_{p'} \langle \_, q\rangle$ for each $r \in \mathbb{R}_{>0}$, $p, p', q \in \mathbb{R}_{\geq 0}$ where $p' \leq p$ and $q = p + r$,
- a unary relation $\langle \_, p\rangle{\uparrow}_{p'}$ for each $p, p' \in \mathbb{R}_{\geq 0}$ where $p' \leq p$;

satisfying

- if $\langle s, p\rangle \overset{r+r'}{\longmapsto}_{p'} \langle s', q\rangle$, $r, r' > 0$, then there is a s'' such that $\langle s, p\rangle \overset{r}{\longmapsto}_{p'} \langle s'', p + r\rangle$ and $\langle s'', p + r\rangle \overset{r'}{\longmapsto}_{p'} \langle s', q\rangle$;
- if $\langle s, p\rangle \overset{r}{\longmapsto}_{p'} \langle s'', p + r\rangle$ and $\langle s'', p + r\rangle \overset{r'}{\longmapsto}_{p'} \langle s', q\rangle$, then $\langle s, p\rangle \overset{r+r'}{\longmapsto}_{p'} \langle s', q\rangle$.

We write $\mathsf{RTTS}^{+}(\mathsf{A})$ for the set of all real time transition systems with initialization times over A.

We can associate a transition system in $\mathsf{RTTS}^{+}(\mathsf{A})$ with a closed term t of ACP$^{\mathrm{sat}}$IC like before. The action step, action termination, time step and deadlocked relations can be explained by adding the proviso "provided t is initialized at time p'" to the explanation given for the case of the original real time transition systems in Section 2.1.

The structural operational semantics of BPA$^{\mathrm{sat}}$C is described by the rules

Table 11

Rules for $\mathrm{BPA^{sat}C}$ ($a \in \mathsf{A}$, $r > 0$, $p, p', q, q', r' \geq 0$, $p' \leq p$, $q' \leq q$, $r' \leq r$)

$$\frac{}{\langle \dot{\delta}, p \rangle \uparrow_{p'}} \qquad \frac{}{\langle \tilde{\delta}, r \rangle \uparrow_{r'}} \qquad \frac{}{\langle \tilde{a}, 0 \rangle \xrightarrow{a}_0 \langle \surd, 0 \rangle} \qquad \frac{}{\langle \tilde{a}, r \rangle \uparrow_{r'}}$$

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle}{\langle \sigma^0_{\mathsf{abs}}(x), p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}{\langle \sigma^0_{\mathsf{abs}}(x), p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}$$

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle}{\langle \sigma^r_{\mathsf{abs}}(x), p + r \rangle \xrightarrow{a}_{p'} \langle \sigma^r_{\mathsf{abs}}(x'), p + r \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}{\langle \sigma^r_{\mathsf{abs}}(x), p + r \rangle \xrightarrow{a}_{p'} \langle \surd, p + r \rangle}$$

$$\frac{q > p}{\langle \sigma^{q+r}_{\mathsf{abs}}(x), p \rangle \overset{r}{\longmapsto}_{p'} \langle \sigma^{q+r}_{\mathsf{abs}}(x), p + r \rangle} \qquad \frac{\neg(\langle x, 0 \rangle \uparrow_0)}{\langle \sigma^{q+r}_{\mathsf{abs}}(x), q \rangle \overset{r}{\longmapsto}_{q'} \langle \sigma^{q+r}_{\mathsf{abs}}(x), q + r \rangle}$$

$$\frac{\langle x, p \rangle \overset{r}{\longmapsto}_{p'} \langle x, p + r \rangle}{\langle \sigma^q_{\mathsf{abs}}(x), p + q \rangle \overset{r}{\longmapsto}_{p'} \langle \sigma^q_{\mathsf{abs}}(x), p + q + r \rangle} \qquad \frac{\langle x, p \rangle \uparrow_{p'}}{\langle \sigma^q_{\mathsf{abs}}(x), p + q \rangle \uparrow_{p'}}$$

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle}{\langle x + y, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle,\ \langle y + x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}{\langle x + y, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle,\ \langle y + x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}$$

$$\frac{\langle x, p \rangle \overset{r}{\longmapsto}_{p'} \langle x, p + r \rangle}{\langle x + y, p \rangle \overset{r}{\longmapsto}_{p'} \langle x + y, p + r \rangle,\ \langle y + x, p \rangle \overset{r}{\longmapsto}_{p'} \langle y + x, p + r \rangle} \qquad \frac{\langle x, p \rangle \uparrow_{p'},\ \langle y, p \rangle \uparrow_{p'}}{\langle x + y, p \rangle \uparrow_{p'}}$$

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle}{\langle x \cdot y, p \rangle \xrightarrow{a}_{p'} \langle x' \cdot y, p \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}{\langle x \cdot y, p \rangle \xrightarrow{a}_{p'} \langle y, p \rangle} \qquad \frac{\langle x, p \rangle \overset{r}{\longmapsto}_{p'} \langle x, p + r \rangle}{\langle x \cdot y, p \rangle \overset{r}{\longmapsto}_{p'} \langle x \cdot y, p + r \rangle} \qquad \frac{\langle x, p \rangle \uparrow_{p'}}{\langle x \cdot y, p \rangle \uparrow_{p'}}$$

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle,\ p' \in [b]}{\langle b :\rightarrow x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle,\ p' \in [b]}{\langle b :\rightarrow x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}$$

$$\frac{\langle x, p \rangle \overset{r}{\longmapsto}_{p'} \langle x, p + r \rangle,\ p' \in [b]}{\langle b :\rightarrow x, p \rangle \overset{r}{\longmapsto}_{p'} \langle b :\rightarrow x, p + r \rangle} \qquad \frac{\langle x, p \rangle \uparrow_{p'},\ p' \in [b]}{\langle b :\rightarrow x, p \rangle \uparrow_{p'}} \qquad \frac{p' \notin [b]}{\langle b :\rightarrow x, p \rangle \uparrow_{p'}}$$

Table 12

Rules for $\mathrm{BPA^{sat}C}$ ($a \in \mathsf{A}$, $r > 0$, $p, p', q, q' \geq 0$, $p' \leq p$, $q' \leq q$)

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle,\ q > p}{\langle \upsilon^q_{\mathsf{abs}}(x), p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle,\ q > p}{\langle \upsilon^q_{\mathsf{abs}}(x), p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}$$

$$\frac{\langle x, p \rangle \overset{r}{\longmapsto}_{p'} \langle x, p + r \rangle,\ q > p + r}{\langle \upsilon^q_{\mathsf{abs}}(x), p \rangle \overset{r}{\longmapsto}_{p'} \langle \upsilon^q_{\mathsf{abs}}(x), p + r \rangle} \qquad \frac{q \leq p}{\langle \upsilon^q_{\mathsf{abs}}(x), p \rangle \uparrow_{p'}} \qquad \frac{\langle x, p \rangle \uparrow_{p'},\ q > p}{\langle \upsilon^q_{\mathsf{abs}}(x), p \rangle \uparrow_{p'}}$$

$$\frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle}{\langle \overline{\upsilon}^{p'}_{\mathsf{abs}}(x), p \rangle \xrightarrow{a}_{p'} \langle x', p \rangle} \qquad \frac{\langle x, p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}{\langle \overline{\upsilon}^{p'}_{\mathsf{abs}}(x), p \rangle \xrightarrow{a}_{p'} \langle \surd, p \rangle}$$

$$\frac{q > p}{\langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), p \rangle \overset{r}{\longmapsto}_{p'} \langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), p + r \rangle} \qquad \frac{\neg(\langle x, q + r \rangle \uparrow_{q'})}{\langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), q \rangle \overset{r}{\longmapsto}_{q'} \langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), q + r \rangle}$$

$$\frac{\langle x, p \rangle \overset{r}{\longmapsto}_{p'} \langle x, p + r \rangle}{\langle \overline{\upsilon}^{p'}_{\mathsf{abs}}(x), p \rangle \overset{r}{\longmapsto}_{p'} \langle \overline{\upsilon}^{p'}_{\mathsf{abs}}(x), p + r \rangle} \qquad \frac{\langle x, p \rangle \uparrow_{p'}}{\langle \overline{\upsilon}^{p'}_{\mathsf{abs}}(x), p \rangle \uparrow_{p'}}$$

given in Tables 11 and 12. In the rules for the conditional operator, use is made of unary relations $p \in [\_]$ on conditions (for $p \in \mathbb{R}_{\geq 0}$). In Table 13, these relations are defined using rules in the style of structural operational semantics as well. The intended meaning of $p \in [b]$ is that p belongs to the time points at which condition b holds. Apart from the rules for the initialization operator $\overline{\upsilon}_{\mathsf{abs}}$, the rules for the operational semantics of $\mathrm{BPA^{sat}}$ (Table A.1) have been

Table 13
Rules for condition evaluation ($p, q \in \mathbb{R}_{\geq 0}$, $s \in \mathbb{R}$)

$\frac{}{p \in [\mathsf{t}]}$	$\frac{}{p \in [\mathsf{pt}(p)]}$	$\frac{p > s}{p \in [\mathsf{pt}_{>}(s)]}$	$\frac{p \notin [b]}{p \in [\neg b]}$	$\frac{p \in [b], p \in [b']}{p \in [b \wedge b']}$	$\frac{p \in [b]}{p \in [b \vee b'], p \in [b' \vee b]}$	$\frac{q \in [b]}{p \in [\overline{v}^{q}_{\mathsf{abs}}(b)]}$

adapted in a simple uniform way. The rules for the conditional operator ($:\!:\!\rightarrow$) express that the capabilities of a process $b :\!:\!\rightarrow x$ are those of x if it is initialized when b holds; and those of $\dot{\delta}$ if it is initialized when b does not hold. The rules for the initialization operator ($\overline{v}_{\mathsf{abs}}$) have been adapted to deal with the fact that the capabilities of x at time p are not necessarily taken over by $\overline{v}^{p'}_{\mathsf{abs}}(x)$ for all $p' \leq p$ in the presence of conditionals. The additional rules for ACP$^{\mathrm{sat}}$IC are obtained by adapting the additional rules for ACP$^{\mathrm{sat}}$I (Tables A.2 and A.3) in the same way.

Bisimulation on $\mathsf{RTTS}^{+}(\mathsf{A})$ is defined similar to bisimulation on $\mathsf{RTTS}(\mathsf{A})$. Like before, we obtain a model for ACP$^{\mathrm{sat}}$IC by identifying bisimilar processes.

3.3 Recursion

In this paper, we do not treat the addition of recursion to any of the presented versions of ACP with timing in detail. However, we describe in this subsection the addition of recursion to ACP$^{\mathrm{sat}}$IC in outline to make understanding of the specifications given in Section 3.4 easier.

In case of ACP$^{\mathrm{sat}}$IC, recursive specification, solution and guardedness are defined in a similar way as for ACP in Ref. [3].

Let V be a set of variables of sort P. A *recursive specification* $E = E(V)$ in ACP$^{\mathrm{sat}}$IC is a set of equations $E = \{X = t_X \mid X \in V\}$ where each t_X is a ACP$^{\mathrm{sat}}$IC term that only contains variables from V. A *solution* of a recursive specification $E(V)$ in ACP$^{\mathrm{sat}}$IC is a set of processes $\{p_X \mid X \in V\}$ in some model of ACP$^{\mathrm{sat}}$IC such that the equations of $E(V)$ hold if, for all $X \in V$, X stands for p_X. Mostly, we are interested in one particular variable $X \in V$. When adding recursion, we add constants $\langle X|E\rangle \colon \rightarrow \mathsf{P}$ for all recursive specifications $E(V)$ and all $X \in V$. For a fixed $E(V)$, the constants $\langle X|E\rangle$ for $X \in V$ make up a solution of $E(V)$.

Let t be a term containing a variable X. We call an occurrence of X in t *guarded* if t has a subterm of the form $\tilde{a} \cdot t'$ or $\sigma^{r}_{\mathsf{abs}}(t')$ with $r \in \mathbb{R}_{>0}$ and t' a term containing this occurrence of X. We call a recursive specification *guarded* if all occurrences of all its variables in the right-hand sides of all its equations are guarded or it can be rewritten to such a recursive specification using the axioms of ACP$^{\mathrm{sat}}$IC and its equations. The Recursive Specification Principle (RSP) states that every guarded recursive specification has a unique solution.

It is possible to obtain a model of $\mathrm{ACP^{sat}IC}$ with recursion in which every guarded recursive specification has a unique solution.

Let $E = \{X = t_X \mid X \in V\}$ be a recursive specification in $\mathrm{ACP^{sat}IC}$. Then roughly, the additional rules for the operational semantics of $\mathrm{ACP^{sat}IC}$ with recursion come down to looking upon $\langle X|E\rangle$ as the process t_X with, for all $Y \in V$, all occurrences of Y in t_X replaced by $\langle Y|E\rangle$. In the model of $\mathrm{ACP^{sat}IC}$ with recursion obtained in the same way as for $\mathrm{ACP^{sat}IC}$ (Section 3.2), every guarded recursive specification has a unique solution.

In the recursive specifications given in Section 3.4, we use equations of the form $X(p) = t$, with p ranging over some interval I of $\mathbb{R}_{\geq 0}$, for a system of equations with one equation for each $p \in I$. The advantage of this view is that the $X(p)$ do not have free variables and no complications arise with name clashes and α-conversion. It is possible to view such equations as single ones instead, but in that case terms with parameters have to be understood in detail.

3.4 Example

We will now use $\mathrm{ACP^{sat}IC}$ in an example concerning railroad crossings. Controlling a railroad crossing involves the behaviour of trains, a gate and a controller. We shall give (guarded recursive) specifications of the behaviour that is relevant to railroad crossing control. We take the following informal description of the time-dependent behaviour of the trains, the gate and the controller from Ref. [22] as the starting-point of our specifications. The example originates from Ref. [23].

When a train approaches the gate from a great distance its speed is between 48 m/s and 52 m/s. As soon as it passes a detector placed at 1000 m backward from the gate, an *app* signal is sent to the controller. The train may now slow down, but its speed stays between 40 m/s and 52 m/s, and pass the gate. As soon as it passes another detector placed at 100 m forward from the gate, an *exit* signal is sent to the controller. A new train may come after the current one has passed the second detector, but only at a distance greater than or equal to 1500 m. The gate is able to receive *lower* and *raise* signals from the controller at any time. As soon as the gate receives a *lower* signal, it lowers from 90° to 0° at a constant rate of 20° per second. As soon as it receives a *raise* signal, it raises from 0° to 90° at the same rate. The controller is able to receive *app* and *exit* signals from the train detectors at any time. When the controller receives an *app* signal, it takes at most 5 s before a *lower* signal is sent to the gate. When it receives an *exit* signal, it takes at most 5 s before a *raise* signal is sent to the gate. Because of fault tolerance considerations, *app* signals should always cause the gate to go down, and *exit* signals should be ignored while the gate is going down.

In the specifications given below, actions are used to model the acts of sending and receiving signals as well as the acts of passing the gate and completing the opening or the closing of the gate. In the specification of the behaviour of the gate, a ranges over the interval $[0, 90]$ of $\mathbb{R}_{\geq 0}$. In the specification of the behaviour of the controller, d ranges over the interval $[0, 5]$ of $\mathbb{R}_{\geq 0}$.

$$\begin{aligned}
\mathit{Trains} &= \textstyle\int_{t \in [0,\infty)} (\mathsf{pt}_{\leq}(t - \frac{400}{52}) :\!:\!\rightarrow \sigma^t_{\mathsf{abs}}(\widetilde{app}_{tr} \cdot T_{near})) \\
T_{near} &= \textstyle\int_{t \in [0,\infty)} ((\mathsf{pt}_{\leq}(t - \frac{1000}{52}) \land \mathsf{pt}_{\geq}(t - \frac{1000}{40})) :\!:\!\rightarrow \sigma^t_{\mathsf{abs}}(\widetilde{pass} \cdot T_{past})) \\
T_{past} &= \textstyle\int_{t \in [0,\infty)} ((\mathsf{pt}_{\leq}(t - \frac{100}{52}) \land \mathsf{pt}_{\geq}(t - \frac{100}{40})) :\!:\!\rightarrow \sigma^t_{\mathsf{abs}}(\widetilde{exit}_{tr} \cdot \mathit{Trains}))
\end{aligned}$$

Some simple calculations give us the lower and upper bounds for the times at which a train may pass the detectors and the gate. If a train goes at time t_0 from one point to another point at a distance d with a speed between v_l and v_h, then the lower and upper bounds for the time t at which the train passes the latter point are couched by the assertions $t_0 + \frac{d}{v_h} \leq t$ and $t \leq t_0 + \frac{d}{v_l}$, respectively. The conditions used in the specification given above are modelled on the equivalent assertions $t_0 \leq t - \frac{d}{v_h}$ and $t_0 \geq t - \frac{d}{v_l}$. There is only a lower bound in case of the first detector because the train that comes after the current one may be at any distance greater than or equal to 400 m backward from the first detector.

$$\begin{aligned}
\mathit{Gate} &= \textstyle\int_{t \in [0,\infty)} \sigma^t_{\mathsf{abs}}(\widetilde{lower}_g \cdot G_{dn}(90) + \widetilde{raise}_g \cdot \mathit{Gate}) \\
G_{dn}(a) &= \textstyle\int_{t \in [0,\infty)} (\mathsf{pt}(t - \frac{a}{20}) :\!:\!\rightarrow \sigma^t_{\mathsf{abs}}(\widetilde{ready} \cdot G_{cl}) + \mathsf{pt}_{\geq}(t - \frac{a}{20}) :\!:\!\rightarrow \\
&\qquad \sigma^t_{\mathsf{abs}}(\widetilde{lower}_g \cdot G_{dn}(a - 20t) + \widetilde{raise}_g \cdot G_{up}(a - 20t))) \\
G_{cl} &= \textstyle\int_{t \in [0,\infty)} \sigma^t_{\mathsf{abs}}(\widetilde{lower}_g \cdot G_{cl} + \widetilde{raise}_g \cdot G_{up}(0)) \\
G_{up}(a) &= \textstyle\int_{t \in [0,\infty)} (\mathsf{pt}(t - \frac{90-a}{20}) :\!:\!\rightarrow \sigma^t_{\mathsf{abs}}(\widetilde{ready} \cdot \mathit{Gate}) + \mathsf{pt}_{\geq}(t - \frac{90-a}{20}) :\!:\!\rightarrow \\
&\qquad \sigma^t_{\mathsf{abs}}(\widetilde{lower}_g \cdot G_{dn}(a + 20t) + \widetilde{raise}_g \cdot G_{up}(a + 20t)))
\end{aligned}$$

While the gate is going up or down, its angle a is relevant to its behaviour. When a controller signal is received, the time passed since the previous controller signal was received determines the new angle.

$$\begin{aligned}
\mathit{Cntr} &= \textstyle\int_{t \in [0,\infty)} \sigma^t_{\mathsf{abs}}(\widetilde{app}_c \cdot C_{dn}(0) + \widetilde{exit}_c \cdot C_{up}(0)) \\
C_{dn}(d) &= \textstyle\int_{t \in [0,\infty)} (\mathsf{pt}_{\geq}(t - (5 - d)) :\!:\!\rightarrow \\
&\qquad \sigma^t_{\mathsf{abs}}(\widetilde{lower}_c \cdot \mathit{Cntr} + \widetilde{app}_c \cdot C_{dn}(d + t) + \widetilde{exit}_c \cdot C_{dn}(d + t))) \\
C_{up}(d) &= \textstyle\int_{t \in [0,\infty)} (\mathsf{pt}_{\geq}(t - (5 - d)) :\!:\!\rightarrow
\end{aligned}$$

$$\sigma^{t}_{\mathsf{abs}}(\widetilde{\widetilde{raise}}_c \cdot Cntr + \widetilde{\widetilde{app}}_c \cdot C_{dn}(0) + \widetilde{\widetilde{exit}}_c \cdot C_{up}(d+t)))$$

While the controller is preparing for sending a signal to the gate in response to a detector signal, the delay d of the response is relevant to its behaviour. When another detector signal is received, the time passed since the previous detector signal was received determines the new delay.

Let the communication function γ be such that

$$\gamma(app_{tr}, app_c) = app, \gamma(exit_{tr}, exit_c) = exit,$$
$$\gamma(lower_c, lower_g) = lower, \gamma(raise_c, raise_g) = raise$$

and γ is undefined otherwise. Then the railroad crossing system is described by

$$\partial_H(Trains \parallel Cntr \parallel Gate)$$

where

$$H = \{app_{tr}, app_c, exit_{tr}, exit_c, lower_c, lower_g, raise_c, raise_g\}$$

Analysis of this term can provide answers to various basic questions about the system. It can, for example, be simplified to a term which shows that (1) a train can only pass the gate when the gate is closed, (2) the gate opens after a train has left the track unless a new train has entered the track and (3) the system reacts adequately when a new train enters the track while the gate is going up. We do not give an account of the simplification here. It involves the use of various standard process algebraic techniques, such as linearization of guarded recursive specifications and expansion of parallel composition (see e.g. Ref. [22]), of which the treatment in the setting of ACP$^{\mathrm{sat}}$IC would go beyond the scope of this paper.

4 Discrete time and time-dependent conditions

In this section, we briefly review ACP$^{\mathrm{dat}}\surd$, the discrete time counterpart of ACP$^{\mathrm{sat}}$I$\surd$ presented in Ref. [8], and add a conditional operator with time-dependent conditions to it. In Section 5, we show that the resulting theory, called ACP$^{\mathrm{dat}}\surd$C, can be embedded in ACP$^{\mathrm{sat}}$I$\surd$C. In ACP$^{\mathrm{dat}}\surd$C, the conditions are essentially the same as the conditions introduced earlier in Ref. [15]. First, in Section 4.1, we review ACP$^{\mathrm{dat}}\surd$. After that, in Section 4.2, we extend ACP$^{\mathrm{dat}}\surd$ to ACP$^{\mathrm{dat}}\surd$C.

In this subsection, we briefly review $\mathrm{ACP}^{\mathrm{dat}}$, a discrete time process algebra with absolute timing, and its extension with initial abstraction. A more detailed account is given in Ref. [8]. The axioms – extracted from Ref. [8] – are given in Appendix B.

$\mathrm{ACP}^{\mathrm{dat}}$ is a conservative extension of $\mathrm{ACP}_{\mathrm{dat}}$ [15]. In $\mathrm{ACP}^{\mathrm{dat}}$, 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 $n \leq p < n+1$".

In $\mathrm{ACP}^{\mathrm{dat}}$, we have the constants $\underline{a}$ and $\underline{\delta}$ instead of $\tilde{a}$ and $\tilde{\tilde{\delta}}$. The constants $\underline{a}$ and $\underline{\delta}$ stand for a in time slice 1 and a deadlock in time slice 1, respectively. The operators σ_{abs}, υ_{abs} and $\overline{\upsilon}_{\mathsf{abs}}$ have a natural number instead of a non-negative real number as their first argument. The process $\sigma^n_{\mathsf{abs}}(x)$ is the process x shifted in time by n on the discrete time scale. The process $\upsilon^n_{\mathsf{abs}}(x)$ is the part of x that starts to perform actions before time slice $n+1$. The process $\overline{\upsilon}^n_{\mathsf{abs}}(x)$ 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 $\mathrm{ACP}^{\mathrm{dat}}$, we do not have a discrete time counterpart of ν_{abs}. Unlike before in the case of real time, we can use $\upsilon^1_{\mathsf{abs}}$ instead. The initial abstraction operator $\surd_{\mathsf{d}}$ is the discrete counterpart of $\surd_{\mathsf{s}}$. This means that $\surd_{\mathsf{d}} i \mathbin{.} F$, where i is a variable ranging over $\mathbb{N}$ and F is a term that may contain free variables, denotes a function $f : \mathbb{N} \to \mathsf{P}$ that satisfies $f(n) = \overline{\upsilon}^n_{\mathsf{abs}}(f(n))$ for all $n \in \mathbb{N}$. In the resulting theory, called $\mathrm{ACP}^{\mathrm{dat}}\surd$, the sort P of processes is replaced by the sort P^* of parametric time processes.

We denote elements of $\mathbb{N}$ by m, m', n, n'. We assume that an infinite set of time variables ranging over $\mathbb{N}$ has been given, and denote them by $i, j, \ldots$. We denote terms of $\mathrm{ACP}^{\mathrm{dat}}\surd$ by $F, G, \ldots$.

Axiom systems

The axiom system of $\mathrm{BPA}^{\mathrm{dat}}$ consists of the equations given in Table B.1. The axiom system of $\mathrm{ACP}^{\mathrm{dat}}$ consists of the axioms of $\mathrm{BPA}^{\mathrm{dat}}$ and the equations given in Table B.2. The axiom system of $\mathrm{ACP}^{\mathrm{dat}}\surd$ consists of the axioms of $\mathrm{ACP}^{\mathrm{dat}}$ and the equations given in Table B.3. For a discussion of the axioms of $\mathrm{BPA}^{\mathrm{dat}}$, $\mathrm{ACP}^{\mathrm{dat}}$ and $\mathrm{ACP}^{\mathrm{dat}}\surd$, see Ref. [8].

In case a discrete time scale is used, we use a variant of real time transition systems, called *discrete time transition systems*, with only relations $\langle\_, p\rangle \xrightarrow{a} \langle\_, p\rangle$, $\langle\_, p\rangle \xrightarrow{a} \langle\surd, p\rangle$, $\langle\_, p\rangle \overset{r}{\mapsto} \langle\_, q\rangle$ and $\langle\_, p\rangle \uparrow$ for $p, q \in \mathbb{N}$, $r \in \mathbb{N}_{>0}$. We write $\mathsf{DTTS}(\mathsf{A})$ for the set of all discrete time transition systems over A. Associating a transition system in $\mathsf{DTTS}(\mathsf{A})$ with a closed term t of BPA$^{\mathrm{dat}}$ and ACP$^{\mathrm{dat}}$ proceeds in essentially the same way as associating a transition system in $\mathsf{RTTS}(\mathsf{A})$ with a closed term t of BPA$^{\mathrm{sat}}$ and ACP$^{\mathrm{sat}}$. The only difference is that in the rules for the operational semantics of BPA$^{\mathrm{dat}}$ and ACP$^{\mathrm{dat}}$ all numbers involved are restricted to $\mathbb{N}$. For ACP$^{\mathrm{dat}}\surd$, we have to extend $\mathsf{DTTS}(\mathsf{A})$ to the function space

$$\mathsf{DTTS}^*(\mathsf{A}) = \{f : \mathbb{N} \to \mathsf{DTTS}(\mathsf{A}) \mid \forall n \in \mathbb{N} \bullet f(n) = \overline{\upsilon}^{n}_{\mathsf{abs}}(f(n))\}$$

4.2 Conditionals with time-dependent conditions

We add a conditional operator with time-dependent conditions to ACP$^{\mathrm{dat}}\surd$. The time-dependent conditions introduced here were originally introduced in Ref. [15] (see also Ref. [24]).

First of all, we introduce time-dependent conditions for the discrete time case. We have the *in time slice* operator sl and the *in time slice greater than* operator $\mathsf{sl}_>$ instead of pt and $\mathsf{pt}_>$. The operator $\overline{\upsilon}_{\mathsf{abs}}$ has a natural number instead of a non-negative real number as its first argument.

For a time-dependent condition b, $\overline{\upsilon}^{n}_{\mathsf{abs}}(b)$ is either t or f, determined by whether b holds in time slice $n+1$ or not. For $n \in \mathbb{N}$, the condition $\mathsf{sl}(n)$ holds only in time slice n and the condition $\mathsf{sl}_>(n)$ holds in all time slices greater than n. For $m \in \mathbb{N}_{>0}$, the condition $\mathsf{sl}(-m)$ never holds and the condition $\mathsf{sl}_>(-m)$ always holds. We also have the initial abstraction operator $\surd_{\mathsf{d}}$, instead of $\surd_{\mathsf{s}}$, for conditions.

We join time-dependent conditions with parametric time processes by means of the conditional operator $:\!:\!\to$. In ACP$^{\mathrm{dat}}\surd$C, we have, in addition to the above-mentioned constants and operators on $\mathbb{B}^*$, the constants and operators of ACP$^{\mathrm{dat}}\surd$ and the *conditional* operator $:\!:\!\to\, : \mathbb{B}^* \times \mathsf{P}^* \to \mathsf{P}^*$.

Initialized in a time slice $n+1$ where the condition b holds, the process $b :\!:\!\to x$ proceeds as the process x initialized in time slice $n+1$; and initialized in a time slice $n+1$ where the condition b does not hold, it proceeds as the process $\dot{\delta}$ initialized in time slice $n+1$.

Table 14
Axioms for conditions ($n, n' \geq 0$, $m > 0$, i not free in D)

$\overline{v}^n_{\mathsf{abs}}(\mathsf{t}) = \mathsf{t}$	CDAI1	$\surd_{\mathsf{d}} j \,.\, D = \surd_{\mathsf{d}} i \,.\, D[i/j]$	CDIA1
$\overline{v}^n_{\mathsf{abs}}(\mathsf{f}) = \mathsf{f}$	CDAI2	$\overline{v}^n_{\mathsf{abs}}(\surd_{\mathsf{d}} i \,.\, C) = \overline{v}^n_{\mathsf{abs}}(C[n/i])$	CDIA2
$\overline{v}^n_{\mathsf{abs}}(\mathsf{sl}(n+1)) = \mathsf{t}$	CDAI3	$\surd_{\mathsf{d}} i \,.\, (\surd_{\mathsf{d}} j \,.\, C) = \surd_{\mathsf{d}} i \,.\, C[i/j]$	CDIA3
$\overline{v}^n_{\mathsf{abs}}(\mathsf{sl}((n+1)-m)) = \mathsf{f}$	CDAI4	$D = \surd_{\mathsf{d}} i \,.\, D$	CDIA4
$\overline{v}^n_{\mathsf{abs}}(\mathsf{sl}((n+1)+m)) = \mathsf{f}$	CDAI5	$(\forall n \in \mathbb{N} \bullet \overline{v}^n_{\mathsf{abs}}(b) = \overline{v}^n_{\mathsf{abs}}(b')) \Rightarrow b = b'$	CDIA5
$\overline{v}^n_{\mathsf{abs}}(\mathsf{sl}_{>}((n+1)-m)) = \mathsf{t}$	CDAI6	$\neg(\surd_{\mathsf{d}} i \,.\, C) = \surd_{\mathsf{d}} i \,.\, \neg C$	CDIA6
$\overline{v}^n_{\mathsf{abs}}(\mathsf{sl}_{>}((n+1)+n')) = \mathsf{f}$	CDAI7	$(\surd_{\mathsf{d}} i \,.\, C) \wedge D = \surd_{\mathsf{d}} i \,.\, (C \wedge \overline{v}^i_{\mathsf{abs}}(D))$	CDIA7
$\overline{v}^n_{\mathsf{abs}}(\neg b) = \neg \overline{v}^n_{\mathsf{abs}}(b)$	CDAI8	$(\surd_{\mathsf{d}} i \,.\, C) \vee D = \surd_{\mathsf{d}} i \,.\, (C \vee \overline{v}^i_{\mathsf{abs}}(D))$	CDIA8
$\overline{v}^n_{\mathsf{abs}}(b \wedge b') = \overline{v}^n_{\mathsf{abs}}(b) \wedge \overline{v}^n_{\mathsf{abs}}(b')$	CDAI9		
$\overline{v}^n_{\mathsf{abs}}(b \vee b') = \overline{v}^n_{\mathsf{abs}}(b) \vee \overline{v}^n_{\mathsf{abs}}(b')$	CDAI10		

Table 15
Axioms for conditionals ($n \geq 0$, i not free in D and G)

$\mathsf{t} :\to x = x$	SGC1
$\mathsf{f} :\to x = \dot{\delta}$	SGC2ID
$\overline{v}^n_{\mathsf{abs}}(b :\to x) = \overline{v}^n_{\mathsf{abs}}(b) :\to \overline{v}^n_{\mathsf{abs}}(x) + \sigma^n_{\mathsf{abs}}(\dot{\delta})$	DASGC1
$x = \sum_{k \in [0,n]} (\mathsf{sl}(k+1) :\to \overline{v}^k_{\mathsf{abs}}(x)) + \mathsf{sl}_{>}(n+1) :\to x$	DASGC2
$b :\to \dot{\delta} = \dot{\delta}$	SGC3ID
$b :\to \sigma^n_{\mathsf{abs}}(x) + \sigma^n_{\mathsf{abs}}(\dot{\delta}) = \surd_{\mathsf{d}} i \,.\, \sigma^n_{\mathsf{abs}}(\overline{v}^i_{\mathsf{abs}}(b) :\to x)$	DASGC3
$b :\to (x + y) = b :\to x + b :\to y$	SGC4
$b :\to x \cdot y = (b :\to x) \cdot y$	SGC5
$(b \vee b') :\to x = b :\to x + b' :\to x$	SGC6
$b :\to (b' :\to x) = (b \wedge b') :\to x$	SGC7
$b :\to v^n_{\mathsf{abs}}(x) = v^n_{\mathsf{abs}}(b :\to x)$	DASGC4
$b :\to (x \mathbin{\|\!\lfloor} y) = (b :\to x) \mathbin{\|\!\lfloor} (b :\to y)$	DASGC5
$b :\to (x \mid y) = (b :\to x) \mid (b :\to y)$	DASGC6
$b :\to \partial_H(x) = \partial_H(b :\to x)$	DASGC7
$D :\to (\surd_{\mathsf{d}} i \,.\, F) = \surd_{\mathsf{d}} i \,.\, (\overline{v}^i_{\mathsf{abs}}(D) :\to F)$	DASGC8
$(\surd_{\mathsf{d}} i \,.\, C) :\to G = \surd_{\mathsf{d}} i \,.\, (C :\to \overline{v}^i_{\mathsf{abs}}(G))$	DASGC9

Axiom system

The axiom system of $\mathrm{ACP}^{\mathrm{dat}}\surd\mathrm{C}$ consists of the axioms of $\mathrm{ACP}^{\mathrm{dat}}\surd$ and the equations given in Tables 6, 14 and 15.

Semantics

In Table 16, the conditional operator is defined on $\mathsf{DTTS}^*(\mathsf{A})$ in terms of the conditional operator, restricted to the conditions t and f, on discrete time transition systems (see also Section 3.1). Additionally, the operators introduced for conditions are defined on $\mathbb{B}^*$. In this table, we use γ to denote elements of $\mathbb{N} \to \mathbb{B}^*$ and t is a variable ranging over $\mathbb{N}$.

Table 16
Definition of conditional operator on DTTS^* ($n \in \mathbb{N}$, $k \in \mathbb{Z}$)

$c :\!\!\to f = \lambda t \,.\, (c(t) :\!\!\to f(t))$	$\neg c = \lambda t \,.\, \neg(c(t))$
$\mathsf{t} = \lambda t \,.\, \mathsf{t}$	$c \wedge d = \lambda t \,.\, (c(t) \wedge d(t))$
$\mathsf{f} = \lambda t \,.\, \mathsf{f}$	$c \vee d = \lambda t \,.\, (c(t) \vee d(t))$
$\mathsf{sl}(k) = \lambda t \,.\, (\mathsf{if}\ t+1 = k\ \mathsf{then}\ \mathsf{t}\ \mathsf{else}\ \mathsf{f})$	$\overline{v}^n_{\mathsf{abs}}(c) = c(n)$
$\mathsf{sl}_{>}(k) = \lambda t \,.\, (\mathsf{if}\ t+1 > k\ \mathsf{then}\ \mathsf{t}\ \mathsf{else}\ \mathsf{f})$	$\surd_{\mathsf{d}}{}^*(\gamma) = \lambda t \,.\, \overline{v}^t_{\mathsf{abs}}(\gamma(t))$

Table 17
Definitions of discrete time operators ($a \in \mathsf{A}_\delta$, $n \in \mathbb{N}$, $k \in \mathbb{Z}$)

$\underline{a} = \int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\tilde{a})$	$\mathsf{sl}(k) = \mathsf{pt}_{\geq}(k-1) \wedge \mathsf{pt}_{<}(k)$
$\sigma^n_{\mathsf{abs}}(x) = \sigma^n_{\mathsf{abs}}(x)$	$\mathsf{sl}_{>}(k) = \mathsf{pt}_{\geq}(k)$
$v^n_{\mathsf{abs}}(x) = v^n_{\mathsf{abs}}(x)$	$\surd_{\mathsf{d}} i \,.\, C = \surd_{\mathsf{s}} v \,.\, C[\lfloor v \rfloor / i]$
$\overline{v}^n_{\mathsf{abs}}(x) = \overline{v}^n_{\mathsf{abs}}(x)$	
$\surd_{\mathsf{d}} i \,.\, F = \surd_{\mathsf{s}} v \,.\, F[\lfloor v \rfloor / i]$	

Table 18
Definition of discretization ($a \in \mathsf{A}_\delta$, $p \in \mathbb{R}_{\geq 0}$, $s \in \mathbb{R}$)

$\mathcal{D}(\dot{\delta}) = \dot{\delta}$	$\mathcal{D}(\mathsf{t}) = \mathsf{t}$
$\mathcal{D}(\tilde{a}) = \underline{a}$	$\mathcal{D}(\mathsf{f}) = \mathsf{f}$
$\mathcal{D}(\sigma^p_{\mathsf{abs}}(x)) = \sigma^{\lfloor p \rfloor}_{\mathsf{abs}}(\mathcal{D}(x))$	$\mathcal{D}(\mathsf{pt}(s)) = \mathsf{sl}(\lfloor s+1 \rfloor)$
$\mathcal{D}(x + y) = \mathcal{D}(x) + \mathcal{D}(y)$	$\mathcal{D}(\mathsf{pt}_{>}(s)) = \mathsf{sl}_{>}(\lfloor s \rfloor)$
$\mathcal{D}(x \cdot y) = \mathcal{D}(x) \cdot \mathcal{D}(y)$	$\mathcal{D}(\neg b) = \neg \mathcal{D}(b)$
$\mathcal{D}(b :\!\!\to x) = \mathcal{D}(b) :\!\!\to \mathcal{D}(x)$	$\mathcal{D}(b \wedge b') = \mathcal{D}(b) \wedge \mathcal{D}(b')$
$\mathcal{D}(\int_{v \in V} F) = \int_{v \in V} \mathcal{D}(F)$	$\mathcal{D}(b \vee b') = \mathcal{D}(b) \vee \mathcal{D}(b')$
$\mathcal{D}(\surd_{\mathsf{s}} v \,.\, F) = \surd_{\mathsf{s}} v \,.\, \mathcal{D}(F)$	$\mathcal{D}(\overline{v}^p_{\mathsf{abs}}(b)) = \overline{v}^{\lfloor p \rfloor}_{\mathsf{abs}}(\mathcal{D}(b))$
	$\mathcal{D}(\surd_{\mathsf{s}} v \,.\, C) = \surd_{\mathsf{s}} v \,.\, \mathcal{D}(C)$

5 Embedding

In this section, we will show that ACP$^{\mathrm{dat}}$√C can be embedded in ACP$^{\mathrm{sat}}$I√C. We will establish the existence of an embedding as follows. We give explicit definitions of the constants and operators in the signature of ACP$^{\mathrm{dat}}$√C that are not in the signature of ACP$^{\mathrm{sat}}$I√C and we prove that for closed terms the axioms of ACP$^{\mathrm{dat}}$√C are derivable from the axioms of ACP$^{\mathrm{sat}}$I√C and the explicit definitions. The soundness of this method is discussed in Ref. [8]. The explicit definitions needed are given in Table 17.

Before we establish the existence of an embedding, we first take another look at the connection between ACP$^{\mathrm{sat}}$I√C and ACP$^{\mathrm{dat}}$√C by introducing the notion of a discretized real time process. Discrete time processes can be viewed as real time processes that are discretized. We define the notion of a discretized real time process in terms of the auxiliary *discretization* operators $\mathcal{D}: \mathsf{P}^* \to \mathsf{P}^*$ and $\mathcal{D}: \mathbb{B}^* \to \mathbb{B}^*$ of which the defining axioms are given in Table 18. In Ref. [8], discretization is also defined on the domain of the model of ACP$^{\mathrm{sat}}$I√C from Section 3.1. A real time process x is a *discretized* real time process, written

Table 19

Properties of discretized processes and conditions ($a \in \mathsf{A}_\delta$, $n \in \mathbb{N}$, $k \in \mathbb{Z}$)

$\dot{\delta}, \underline{a} \in \mathsf{DIS}$	$\mathsf{t}, \mathsf{f}, \mathsf{sl}(k), \mathsf{sl}_{>}(k) \in \mathsf{DIS}$
$x \in \mathsf{DIS} \Rightarrow \sigma^n_{\mathsf{abs}}(x), \upsilon^n_{\mathsf{abs}}(x), \overline{\upsilon}^n_{\mathsf{abs}}(x), \partial_H(x) \in \mathsf{DIS}$	$b \in \mathsf{DIS} \Rightarrow \neg b, \overline{\upsilon}^n_{\mathsf{abs}}(b) \in \mathsf{DIS}$
$x, y \in \mathsf{DIS} \Rightarrow x + y, x \cdot y, x \parallel y, x \mathbin{\lfloor\!\lfloor} y, x \mid y \in \mathsf{DIS}$	$b, b' \in \mathsf{DIS} \Rightarrow b \wedge b', b \vee b' \in \mathsf{DIS}$
$b \in \mathsf{DIS}, x \in \mathsf{DIS} \Rightarrow b :\rightarrow x \in \mathsf{DIS}$	$(\forall n \in \mathbb{N} \bullet C[n/i] \in \mathsf{DIS}) \Rightarrow \surd_{\mathsf{d}} i \,.\, C \in \mathsf{DIS}$
$(\forall n \in \mathbb{N} \bullet F[n/i] \in \mathsf{DIS}) \Rightarrow \surd_{\mathsf{d}} i \,.\, F \in \mathsf{DIS}$	$b \in \mathsf{DIS} \Rightarrow \mathcal{D}(b) \in \mathsf{DIS}$
$(\forall p \in V \bullet F[p/v] \in \mathsf{DIS}) \Rightarrow \int_{v \in V} F \in \mathsf{DIS}$	
$x \in \mathsf{DIS} \Rightarrow \mathcal{D}(x) \in \mathsf{DIS}$	

$x \in \mathsf{DIS}$, if $x = \mathcal{D}(x)$. The notion of a discretized real time condition is defined in the same way. The relevant closure properties of discretized real time processes and discretized real time conditions are given in Table 19. Hence, restriction of the domain of the model of ACP$^{\mathrm{sat}}$I$\surd$C to the discretized elements yields a subalgebra of that model. Because we will prove that for closed terms the axioms of ACP$^{\mathrm{dat}}\surd$C are derivable from the axioms of ACP$^{\mathrm{sat}}$I$\surd$C and the explicit definitions, this subalgebra induces a model of ACP$^{\mathrm{dat}}\surd$C.

The following lemmas present other useful properties of discrete time processes. These lemmas are used to shorten the calculations in the proof of Theorem 11.

Lemma 8 *In* ACP$^{\mathrm{sat}}$I$\surd$C*:*

(1) for each closed term b of sort $\mathbb{B}^*$ generated by the embedded constants and operators of ACP$^{\mathrm{dat}}\surd$C*, $b = \surd_{\mathsf{s}} v \,.\, \overline{\upsilon}^{\lfloor v \rfloor}_{\mathsf{abs}}(b)$;*
(2) for each closed term t of sort P^* generated by the embedded constants and operators of ACP$^{\mathrm{dat}}\surd$C*, $t = \surd_{\mathsf{s}} v \,.\, \overline{\upsilon}^{\lfloor v \rfloor}_{\mathsf{abs}}(t)$.*

Lemma 9 *For each $p \in \mathbb{R}_{\geq 0}$ and closed term t of* ACP$^{\mathrm{sat}}$I$\surd$C *generated by the embedded constants and operators of* ACP$^{\mathrm{dat}}\surd$C*, there exists a closed term t' such that $\overline{\upsilon}^p_{\mathsf{abs}}(t) = \sigma^p_{\mathsf{abs}}(t')$, $t' = \overline{\upsilon}^0_{\mathsf{abs}}(t')$, and if $p \in [0,1)$ and $\overline{\upsilon}^p_{\mathsf{abs}}(t) \neq \sigma^p_{\mathsf{abs}}(\dot{\delta})$, $t' = t' + \sigma^{1-p}_{\mathsf{abs}}(\dot{\delta})$ and $\overline{\upsilon}^p_{\mathsf{abs}}(t + \underline{\delta}) = \sigma^p_{\mathsf{abs}}(t' + \tilde{\delta})$.*

Lemma 10 *For each closed term t of* ACP$^{\mathrm{sat}}$I$\surd$C *generated by the embedded constants and operators of* ACP$^{\mathrm{dat}}\surd$C*, there exists a term t' containing no other free variable than w such that $\upsilon^1_{\mathsf{abs}}(t + \underline{\delta}) = \surd_{\mathsf{s}} w \,.\, \int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t') + \tilde{\delta})$.*

Lemmas 8.2, 9 and 10 are lemmas 7, 9 and 10, respectively, from Ref. [8] adapted to the case with conditionals. It suffices to extend the proofs of those lemmas with the case that t is of the form $b :\rightarrow t'$. This is outlined in Appendix C.

Lemma 8 points out that for a real time process corresponding to a discrete time process, the initialization time can always be taken to be a discrete point in time. Lemma 9 shows that for a real time process corresponding to a discrete time process, and for $p \in [0,1)$ such that the whole process is able to reach time p, the part of the process that starts to perform actions at time p or later is able to reach any time $q \in [p,1)$. Lemma 10 indicates that for a real time process corresponding to a discrete time process, the part of the process that starts to perform actions before time 1 can be regarded as a real time process that starts to perform actions at time 0 shifted in time by any $p \in [0,1)$ – and parametrized by the initialization time of the whole process.

The existence of an embedding of $\mathrm{ACP}^{\mathrm{dat}}\surd\mathrm{C}$ in $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}\surd\mathrm{C}$ is now established by proving the following theorem.

Theorem 11 (Embedding $\mathbf{ACP}^{\mathrm{dat}}\surd\mathbf{C}$ in $\mathbf{ACP}^{\mathrm{sat}}\mathbf{I}\surd\mathbf{C}$) *For closed terms, the axioms of* $\mathrm{ACP}^{\mathrm{dat}}\surd\mathrm{C}$ *are derivable from the axioms of* $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}\surd\mathrm{C}$ *and the explicit definitions of the constants and operators* $\underline{a}$, σ_{abs}, υ_{abs}, $\overline{\upsilon}_{\mathsf{abs}}$, $\surd_{\mathsf{d}}$ *(for processes as well as conditions),* sl *and* $\mathsf{sl}_{>}$ *in Table 17.*

This is Theorem 12 from Ref. [8] adapted to the case with conditionals. Because some lemmas used in the proof of that theorem had to be adapted to the case with conditionals as well, minor changes to the proofs for some axioms of $\mathrm{ACP}^{\mathrm{dat}}\surd$ are needed. What remains to be shown is that the additional axioms for conditionals are derivable for closed terms. This is outlined in Appendix C.

6 Concluding remarks

We extended the main real time version of ACP presented in Ref. [8] with conditionals in which the condition depends on time. We illustrated how this extension can be used by means of an example concerning a simple hybrid system, namely a railroad crossing system. We also extended the main discrete time version of ACP presented in Ref. [8] with conditionals in which the condition depends on time. The conditions introduced in this case are essentially the same as the ones originally introduced in Ref. [15]. We demonstrated that the presented real time version of ACP with time-dependent conditions and conditionals generalizes the presented discrete time version of ACP with time-dependent conditions and conditionals.

The discrete time version of ACP with time-dependent conditions and conditionals presented in Ref. [15] cannot be embedded in the one presented here – although the conditions introduced are essentially the same. The reason is that one of the auxiliary operators used in Ref. [15] for the axiomatization of the time-dependent conditions and conditionals, viz. the spectrum tail operator μ, cannot be explicitly defined in the version presented here. We refrained from introducing an additional operator making this operator explicitly definable because its usefulness in practice remains doubtful.

In Section 5, we introduced the discretization operator to define the notion of a discretized real time process. However, this is not the only application of this operator. Having a closed term t denoting some real time process, one often obtains by apposite change of the time scale a closed term t' denoting a discretized real time process, i.e. $t' = \mathcal{D}(t')$. In that case, the process can safely be considered at a more abstract level where time is measured with finite precision, i.e. on a discrete time scale. This means that analysis of the real time process t can be replaced by analysis of the discrete time process $\mathcal{D}(t')$. The point here is that the abstraction made in the discrete time case makes processes better amenable to analysis.

It is frequently useful to abstract fully from the timing aspects of a process at a certain stage of its analysis. This is, for example, the case in the analysis of a railroad crossing system outlined in Section 3.4. Further extension of the real time and discrete time versions of ACP presented in this paper with time abstraction appears to be important to make them suitable for being applied in a fully formal way.

A Semantics of $\mathrm{ACP}^{\mathrm{sat}}$ and its extensions

The structural operational semantics of $\mathrm{BPA}^{\mathrm{sat}}$ is described by the rules given in Table A.1. The structural operational semantics of $\mathrm{ACP}^{\mathrm{sat}}$ is described by the rules given in Tables A.1 and A.2. The additional rules for integration are given in Table A.3.

In Table A.4, the constants and operators of $\mathrm{ACP}^{\mathrm{sat}}\mathrm{I}\surd$ are defined on $\mathsf{RTTS}^*(\mathsf{A})$. We use f and g to denote elements of $\mathsf{RTTS}^*(\mathsf{A})$ and φ to denote elements of $\mathbb{R}_{\geq 0} \to \mathsf{RTTS}^*(\mathsf{A})$. We use λ-notation for functions, t and t' are variables ranging over $\mathbb{R}_{\geq 0}$. We write $f(t)*g$ for the real time transition system obtained from $f(t)$ by replacing $\langle s, p\rangle \xrightarrow{a} \langle \surd, p\rangle$ by $\langle s, p\rangle \xrightarrow{a} \langle s', p\rangle$, where s' is the root state of $g(p)$, whenever s is reachable from the root state of $f(t)$.

B Axioms of $\mathrm{ACP}^{\mathrm{dat}}$ and discrete initial abstraction

The axiom system of $\mathrm{BPA}^{\mathrm{dat}}$ consists of the equations given in Table B.1. The axiom system of $\mathrm{ACP}^{\mathrm{dat}}$ consists of the equations given in Tables B.1 and B.2. The axioms for discrete initial abstraction are given in Table B.3.

Table A.1
Rules for operational semantics of $\mathrm{BPA}^{\mathrm{sat}}$ ($a \in \mathsf{A}$, $r > 0$, $p, q \geq 0$)

$$\frac{}{\langle \dot{\delta}, p\rangle \uparrow} \qquad \frac{}{\langle \tilde{\delta}, r\rangle \uparrow} \qquad \frac{}{\langle \tilde{a}, 0\rangle \xrightarrow{a} \langle \surd, 0\rangle} \qquad \frac{}{\langle \tilde{a}, r\rangle \uparrow}$$

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle}{\langle \sigma^{0}_{\mathsf{abs}}(x), p\rangle \xrightarrow{a} \langle x', p\rangle} \qquad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle}{\langle \sigma^{0}_{\mathsf{abs}}(x), p\rangle \xrightarrow{a} \langle \surd, p\rangle}$$

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle}{\langle \sigma^{r}_{\mathsf{abs}}(x), p + r\rangle \xrightarrow{a} \langle \sigma^{r}_{\mathsf{abs}}(x'), p + r\rangle} \qquad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle}{\langle \sigma^{r}_{\mathsf{abs}}(x), p + r\rangle \xrightarrow{a} \langle \surd, p + r\rangle}$$

$$\frac{q > p}{\langle \sigma^{q+r}_{\mathsf{abs}}(x), p\rangle \xmapsto{r} \langle \sigma^{q+r}_{\mathsf{abs}}(x), p + r\rangle} \qquad \frac{\neg(\langle x, 0\rangle \uparrow)}{\langle \sigma^{q+r}_{\mathsf{abs}}(x), q\rangle \xmapsto{r} \langle \sigma^{q+r}_{\mathsf{abs}}(x), q + r\rangle}$$

$$\frac{\langle x, p\rangle \xmapsto{r} \langle x, p + r\rangle}{\langle \sigma^{q}_{\mathsf{abs}}(x), p + q\rangle \xmapsto{r} \langle \sigma^{q}_{\mathsf{abs}}(x), p + q + r\rangle} \qquad \frac{\langle x, p\rangle \uparrow}{\langle \sigma^{q}_{\mathsf{abs}}(x), p + q\rangle \uparrow}$$

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle}{\begin{array}{c}\langle x + y, p\rangle \xrightarrow{a} \langle x', p\rangle, \\ \langle y + x, p\rangle \xrightarrow{a} \langle x', p\rangle\end{array}} \qquad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle}{\begin{array}{c}\langle x + y, p\rangle \xrightarrow{a} \langle \surd, p\rangle, \\ \langle y + x, p\rangle \xrightarrow{a} \langle \surd, p\rangle\end{array}} \qquad \frac{\langle x, p\rangle \xmapsto{r} \langle x, p + r\rangle}{\begin{array}{c}\langle x + y, p\rangle \xmapsto{r} \langle x + y, p + r\rangle, \\ \langle y + x, p\rangle \xmapsto{r} \langle y + x, p + r\rangle\end{array}} \qquad \frac{\langle x, p\rangle \uparrow, \langle y, p\rangle \uparrow}{\langle x + y, p\rangle \uparrow}$$

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle}{\langle x \cdot y, p\rangle \xrightarrow{a} \langle x' \cdot y, p\rangle} \qquad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle}{\langle x \cdot y, p\rangle \xrightarrow{a} \langle y, p\rangle} \qquad \frac{\langle x, p\rangle \xmapsto{r} \langle x, p + r\rangle}{\langle x \cdot y, p\rangle \xmapsto{r} \langle x \cdot y, p + r\rangle} \qquad \frac{\langle x, p\rangle \uparrow}{\langle x \cdot y, p\rangle \uparrow}$$

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle, q > p}{\langle \upsilon^{q}_{\mathsf{abs}}(x), p\rangle \xrightarrow{a} \langle x', p\rangle} \qquad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle, q > p}{\langle \upsilon^{q}_{\mathsf{abs}}(x), p\rangle \xrightarrow{a} \langle \surd, p\rangle}$$

$$\frac{\langle x, p\rangle \xmapsto{r} \langle x, p + r\rangle, q > p + r}{\langle \upsilon^{q}_{\mathsf{abs}}(x), p\rangle \xmapsto{r} \langle \upsilon^{q}_{\mathsf{abs}}(x), p + r\rangle} \qquad \frac{q \leq p}{\langle \upsilon^{q}_{\mathsf{abs}}(x), p\rangle \uparrow} \qquad \frac{\langle x, p\rangle \uparrow, q > p}{\langle \upsilon^{q}_{\mathsf{abs}}(x), p\rangle \uparrow}$$

$$\frac{\langle x, p\rangle \xrightarrow{a} \langle x', p\rangle, q \leq p}{\langle \overline{\upsilon}^{q}_{\mathsf{abs}}(x), p\rangle \xrightarrow{a} \langle x', p\rangle} \qquad \frac{\langle x, p\rangle \xrightarrow{a} \langle \surd, p\rangle, q \leq p}{\langle \overline{\upsilon}^{q}_{\mathsf{abs}}(x), p\rangle \xrightarrow{a} \langle \surd, p\rangle}$$

$$\frac{q > p}{\langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), p\rangle \xmapsto{r} \langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), p + r\rangle} \qquad \frac{\neg(\langle x, q + r\rangle \uparrow)}{\langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), q\rangle \xmapsto{r} \langle \overline{\upsilon}^{q+r}_{\mathsf{abs}}(x), q + r\rangle}$$

$$\frac{\langle x, p\rangle \xmapsto{r} \langle x, p + r\rangle, q \leq p + r}{\langle \overline{\upsilon}^{q}_{\mathsf{abs}}(x), p\rangle \xmapsto{r} \langle \overline{\upsilon}^{q}_{\mathsf{abs}}(x), p + r\rangle} \qquad \frac{\langle x, p\rangle \uparrow, q \leq p}{\langle \overline{\upsilon}^{q}_{\mathsf{abs}}(x), p\rangle \uparrow}$$

Table A.2
Additional rules for $\mathrm{ACP}^{\mathrm{sat}}$ $(a,b,c \in \mathsf{A},\ r>0,\ p\geq 0)$

$$\frac{\langle x,p\rangle \xrightarrow{a} \langle x',p\rangle, \neg(\langle y,p\rangle\uparrow)}{\langle x \parallel y,p\rangle \xrightarrow{a} \langle x'\parallel y,p\rangle, \langle y\parallel x,p\rangle \xrightarrow{a}\langle y\parallel x',p\rangle, \langle x \mathbin{\lfloor\!\lfloor} y,p\rangle \xrightarrow{a} \langle x'\parallel y,p\rangle}$$

$$\frac{\langle x,p\rangle \xrightarrow{a} \langle \surd,p\rangle, \neg(\langle y,p\rangle\uparrow)}{\langle x \parallel y,p\rangle \xrightarrow{a} \langle y,p\rangle, \langle y\parallel x,p\rangle \xrightarrow{a}\langle y,p\rangle, \langle x \mathbin{\lfloor\!\lfloor} y,p\rangle \xrightarrow{a} \langle y,p\rangle}$$

$$\frac{\langle x,p\rangle \xrightarrow{a} \langle x',p\rangle, \langle y,p\rangle \xrightarrow{b} \langle y',p\rangle, \gamma(a,b)=c}{\langle x \parallel y,p\rangle \xrightarrow{c} \langle x'\parallel y',p\rangle, \langle x\mid y,p\rangle \xrightarrow{c}\langle x'\parallel y',p\rangle}$$

$$\frac{\langle x,p\rangle \xrightarrow{a} \langle x',p\rangle, \langle y,p\rangle \xrightarrow{b} \langle \surd,p\rangle, \gamma(a,b)=c}{\begin{array}{c}\langle x \parallel y,p\rangle \xrightarrow{c} \langle x',p\rangle, \langle y\parallel x,p\rangle \xrightarrow{c}\langle x',p\rangle,\\ \langle x\mid y,p\rangle \xrightarrow{c}\langle x',p\rangle, \langle y\mid x,p\rangle \xrightarrow{c}\langle x',p\rangle\end{array}} \qquad \frac{\langle x,p\rangle \xrightarrow{a} \langle \surd,p\rangle, \langle y,p\rangle \xrightarrow{b} \langle \surd,p\rangle, \gamma(a,b)=c}{\langle x \parallel y,p\rangle \xrightarrow{c} \langle \surd,p\rangle, \langle x\mid y,p\rangle \xrightarrow{c}\langle \surd,p\rangle}$$

$$\frac{\langle x,p\rangle \stackrel{r}{\longmapsto} \langle x,p+r\rangle, \langle y,p\rangle \stackrel{r}{\longmapsto} \langle y,p+r\rangle}{\begin{array}{c}\langle x \parallel y,p\rangle \stackrel{r}{\longmapsto} \langle x\parallel y,p+r\rangle, \langle x \mathbin{\lfloor\!\lfloor} y,p\rangle \stackrel{r}{\longmapsto} \langle x \mathbin{\lfloor\!\lfloor} y,p+r\rangle,\\ \langle x\mid y,p\rangle \stackrel{r}{\longmapsto} \langle x\mid y,p+r\rangle\end{array}} \qquad \frac{\langle x,p\rangle\uparrow}{\begin{array}{c}\langle x\parallel y,p\rangle\uparrow, \langle y\parallel x,p\rangle\uparrow, \langle x \mathbin{\lfloor\!\lfloor} y,p\rangle\uparrow,\\ \langle y \mathbin{\lfloor\!\lfloor} x,p\rangle\uparrow, \langle x\mid y,p\rangle\uparrow, \langle y\mid x,p\rangle\uparrow\end{array}}$$

$$\frac{\langle x,p\rangle \xrightarrow{a} \langle x',p\rangle, a\notin H}{\langle \partial_H(x),p\rangle \xrightarrow{a} \langle \partial_H(x'),p\rangle} \qquad \frac{\langle x,p\rangle \xrightarrow{a} \langle \surd,p\rangle, a\notin H}{\langle \partial_H(x),p\rangle \xrightarrow{a} \langle \surd,p\rangle} \qquad \frac{\langle x,p\rangle \stackrel{r}{\longmapsto} \langle x,p+r\rangle}{\langle \partial_H(x),p\rangle \stackrel{r}{\longmapsto} \langle \partial_H(x),p+r\rangle} \qquad \frac{\langle x,p\rangle\uparrow}{\langle \partial_H(x),p\rangle\uparrow}$$

$$\frac{\langle x,0\rangle \xrightarrow{a} \langle x',0\rangle}{\langle \nu_{\mathsf{abs}}(x),0\rangle \xrightarrow{a} \langle x',0\rangle} \qquad \frac{\langle x,0\rangle \xrightarrow{a} \langle \surd,0\rangle}{\langle \nu_{\mathsf{abs}}(x),0\rangle \xrightarrow{a} \langle \surd,0\rangle} \qquad \frac{\langle x,0\rangle\uparrow}{\langle \nu_{\mathsf{abs}}(x),0\rangle\uparrow} \qquad \frac{}{\langle \nu_{\mathsf{abs}}(x),r\rangle\uparrow}$$

Table A.3
Rules for integration $(a\in\mathsf{A},\ r>0,\ p,q\geq 0)$

$$\frac{\langle P[q/v],p\rangle \xrightarrow{a} \langle P',p\rangle, q\in V}{\langle \int_{v\in V} P,p\rangle \xrightarrow{a} \langle P',p\rangle} \qquad \frac{\langle P[q/v],p\rangle \xrightarrow{a} \langle \surd,p\rangle, q\in V}{\langle \int_{v\in V} P,p\rangle \xrightarrow{a} \langle \surd,p\rangle}$$

$$\frac{\langle P[q/v],p\rangle \stackrel{r}{\longmapsto} \langle P[q/v],p+r\rangle, q\in V}{\langle \int_{v\in V} P,p\rangle \stackrel{r}{\longmapsto} \langle \int_{v\in V} P,p+r\rangle} \qquad \frac{\langle\langle P[q/v],p\rangle\uparrow\rangle_{q\in V}}{\langle \int_{v\in V} P,p\rangle\uparrow}$$

Table A.4
Definition of operators on RTTS^* $(a\in\mathsf{A}_\delta,\ p\in\mathbb{R}_{\geq 0})$

$\dot\delta = \lambda t\,.\,\dot\delta$	$f \parallel g = \lambda t\,.\,(f(t)\parallel g(t))$
$\tilde a = \lambda t\,.\,\overline{\upsilon}^t_{\mathsf{abs}}(\tilde a)$	$f \mathbin{\lfloor\!\lfloor} g = \lambda t\,.\,(f(t)\mathbin{\lfloor\!\lfloor} g(t))$
$\sigma^p_{\mathsf{abs}}(f) = \lambda t\,.\,\overline{\upsilon}^t_{\mathsf{abs}}(\sigma^p_{\mathsf{abs}}(f(0)))$	$f \mid g = \lambda t\,.\,(f(t)\mid g(t))$
$f+g = \lambda t\,.\,(f(t)+g(t))$	$\partial_H(f) = \lambda t\,.\,\partial_H(f(t))$
$f\cdot g = \lambda t\,.\,(f(t)*g)$	$\nu_{\mathsf{abs}}(f) = \lambda t\,.\,\overline{\upsilon}^t_{\mathsf{abs}}(\nu_{\mathsf{abs}}(f(t)))$
$\upsilon^p_{\mathsf{abs}}(f) = \lambda t\,.\,\overline{\upsilon}^t_{\mathsf{abs}}(\upsilon^p_{\mathsf{abs}}(f(t)))$	$\int^*(V,\varphi) = \lambda t\,.\,\int(V,\lambda t'\,.\,\varphi(t')(t))$
$\overline{\upsilon}^p_{\mathsf{abs}}(f) = f(p)$	$\surd_{\mathsf{s}}^*(\varphi) = \lambda t\,.\,\overline{\upsilon}^t_{\mathsf{abs}}(\varphi(t))$

Table B.1
Axioms for $\mathrm{BPA}^{\mathrm{dat}}$ ($a \in \mathsf{A}_\delta$)

$x + y = y + x$	A1	$\upsilon^n_{\mathsf{abs}}(\dot{\delta}) = \dot{\delta}$	DATO0
$(x + y) + z = x + (y + z)$	A2	$\upsilon^0_{\mathsf{abs}}(x) = \dot{\delta}$	DATO1
$x + x = x$	A3	$\upsilon^{n+1}_{\mathsf{abs}}(\underline{a}) = \underline{a}$	DATO2
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$\upsilon^{m+n}_{\mathsf{abs}}(\sigma^n_{\mathsf{abs}}(x)) = \sigma^n_{\mathsf{abs}}(\upsilon^m_{\mathsf{abs}}(x))$	DATO3
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\upsilon^n_{\mathsf{abs}}(x + y) = \upsilon^n_{\mathsf{abs}}(x) + \upsilon^n_{\mathsf{abs}}(y)$	DATO4
$x + \dot{\delta} = x$	A6ID	$\upsilon^n_{\mathsf{abs}}(x \cdot y) = \upsilon^n_{\mathsf{abs}}(x) \cdot y$	DATO5
$\dot{\delta} \cdot x = \dot{\delta}$	A7ID		
		$\overline{\upsilon}^0_{\mathsf{abs}}(\dot{\delta}) = \dot{\delta}$	DAI0a
$\sigma^0_{\mathsf{abs}}(x) = \overline{\upsilon}^0_{\mathsf{abs}}(x)$	DAT1	$\overline{\upsilon}^{n+1}_{\mathsf{abs}}(\dot{\delta}) = \sigma^{n+1}_{\mathsf{abs}}(\dot{\delta})$	DAI0b
$\sigma^m_{\mathsf{abs}}(\sigma^n_{\mathsf{abs}}(x)) = \sigma^{m+n}_{\mathsf{abs}}(x)$	DAT2	$\overline{\upsilon}^0_{\mathsf{abs}}(\underline{a}) = \underline{a}$	DAI1
$\sigma^n_{\mathsf{abs}}(x) + \sigma^n_{\mathsf{abs}}(y) = \sigma^n_{\mathsf{abs}}(x + y)$	DAT3	$\overline{\upsilon}^{n+1}_{\mathsf{abs}}(\underline{a}) = \sigma^{n+1}_{\mathsf{abs}}(\dot{\delta})$	DAI2
$\sigma^n_{\mathsf{abs}}(x) \cdot \upsilon^n_{\mathsf{abs}}(y) = \sigma^n_{\mathsf{abs}}(x \cdot \dot{\delta})$	DAT4	$\overline{\upsilon}^{m+n}_{\mathsf{abs}}(\sigma^n_{\mathsf{abs}}(x)) = \sigma^n_{\mathsf{abs}}(\overline{\upsilon}^m_{\mathsf{abs}}(\overline{\upsilon}^0_{\mathsf{abs}}(x)))$	DAI3
$\sigma^n_{\mathsf{abs}}(x) \cdot (\upsilon^n_{\mathsf{abs}}(y) + \sigma^n_{\mathsf{abs}}(z)) = \sigma^n_{\mathsf{abs}}(x \cdot \overline{\upsilon}^0_{\mathsf{abs}}(z))$	DAT5	$\overline{\upsilon}^n_{\mathsf{abs}}(x + y) = \overline{\upsilon}^n_{\mathsf{abs}}(x) + \overline{\upsilon}^n_{\mathsf{abs}}(y)$	DAI4
$\sigma^n_{\mathsf{abs}}(\dot{\delta}) \cdot x = \sigma^n_{\mathsf{abs}}(\dot{\delta})$	DAT6	$\overline{\upsilon}^n_{\mathsf{abs}}(x \cdot y) = \overline{\upsilon}^n_{\mathsf{abs}}(x) \cdot y$	DAI5
$\sigma^1_{\mathsf{abs}}(\dot{\delta}) = \underline{\delta}$	DAT7		
$\underline{a} + \underline{\delta} = \underline{a}$	A6DAa		

Table B.2
Additional axioms for $\mathrm{ACP}^{\mathrm{dat}}$ ($a, b \in \mathsf{A}_\delta$, $c \in \mathsf{A}$)

$x \parallel y = x \mathbin{\lfloor\!\lfloor} y + y \mathbin{\lfloor\!\lfloor} x + x \mid y$	CM1	$\underline{a} \mid \underline{b} = \underline{c}$ if $\gamma(a, b) = c$	CF1DA
$\dot{\delta} \mathbin{\lfloor\!\lfloor} x = \dot{\delta}$	CMID1	$\underline{a} \mid \underline{b} = \underline{\delta}$ if $\gamma(a, b)$ undefined	CF2DA
$x \mathbin{\lfloor\!\lfloor} \dot{\delta} = \dot{\delta}$	CMID2		
$\underline{a} \mathbin{\lfloor\!\lfloor} (x + \underline{\delta}) = \underline{a} \cdot (x + \underline{\delta})$	CM2DA	$\partial_H(\dot{\delta}) = \dot{\delta}$	D0
$\underline{a} \cdot x \mathbin{\lfloor\!\lfloor} (y + \underline{\delta}) = \underline{a} \cdot (x \parallel (y + \underline{\delta}))$	CM3DA	$\partial_H(\underline{a}) = \underline{a}$ if $a \notin H$	D1DA
$\sigma^n_{\mathsf{abs}}(x) \mathbin{\lfloor\!\lfloor} (\upsilon^n_{\mathsf{abs}}(y) + \sigma^n_{\mathsf{abs}}(z)) = \sigma^n_{\mathsf{abs}}(x \mathbin{\lfloor\!\lfloor} z)$	DACM2	$\partial_H(\underline{a}) = \underline{\delta}$ if $a \in H$	D2DA
$(x + y) \mathbin{\lfloor\!\lfloor} z = x \mathbin{\lfloor\!\lfloor} z + y \mathbin{\lfloor\!\lfloor} z$	CM4	$\partial_H(\sigma^n_{\mathsf{abs}}(x)) = \sigma^n_{\mathsf{abs}}(\partial_H(x))$	DAD
$\dot{\delta} \mid x = \dot{\delta}$	CMID3	$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3
$x \mid \dot{\delta} = \dot{\delta}$	CMID4	$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4
$\underline{a} \cdot x \mid \underline{b} = (\underline{a} \mid \underline{b}) \cdot x$	CM5DA		
$\underline{a} \mid \underline{b} \cdot x = (\underline{a} \mid \underline{b}) \cdot x$	CM6DA		
$\underline{a} \cdot x \mid \underline{b} \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$	CM7DA		
$(\upsilon^1_{\mathsf{abs}}(x) + \underline{\delta}) \mid \sigma^{n+1}_{\mathsf{abs}}(y) = \underline{\delta}$	DACM3		
$\sigma^{n+1}_{\mathsf{abs}}(x) \mid (\upsilon^1_{\mathsf{abs}}(y) + \underline{\delta}) = \underline{\delta}$	DACM4		
$\sigma^n_{\mathsf{abs}}(x) \mid \sigma^n_{\mathsf{abs}}(y) = \sigma^n_{\mathsf{abs}}(x \mid y)$	DACM5		
$(x + y) \mid z = x \mid z + y \mid z$	CM8		
$x \mid (y + z) = x \mid y + x \mid z$	CM9		

Table B.3
Axioms for discrete initial abstraction (i not free in G)

$\surd_{\mathsf{d}} j \,.\, G = \surd_{\mathsf{d}} i \,.\, G[i/j]$	DIA1	$v^n_{\mathsf{abs}}(\surd_{\mathsf{d}} i \,.\, F) = \surd_{\mathsf{d}} i \,.\, v^n_{\mathsf{abs}}(F)$	DIA10
$\overline{v}^n_{\mathsf{abs}}(\surd_{\mathsf{d}} i \,.\, F) = \overline{v}^n_{\mathsf{abs}}(F[n/i])$	DIA2	$(\surd_{\mathsf{d}} i \,.\, F) \mathbin{\|\!\lfloor} G = \surd_{\mathsf{d}} i \,.\, (F \mathbin{\|\!\lfloor} \overline{v}^i_{\mathsf{abs}}(G))$	DIA11
$\surd_{\mathsf{d}} i \,.\, (\surd_{\mathsf{d}} j \,.\, F) = \surd_{\mathsf{d}} i \,.\, F[i/j]$	DIA3	$G \mathbin{\|\!\lfloor} (\surd_{\mathsf{d}} i \,.\, F) = \surd_{\mathsf{d}} i \,.\, (\overline{v}^i_{\mathsf{abs}}(G) \mathbin{\|\!\lfloor} F)$	DIA12
$G = \surd_{\mathsf{d}} i \,.\, G$	DIA4	$(\surd_{\mathsf{d}} i \,.\, F) \mid G = \surd_{\mathsf{d}} i \,.\, (F \mid \overline{v}^i_{\mathsf{abs}}(G))$	DIA13
$(\forall n \in \mathbb{N} \bullet \overline{v}^n_{\mathsf{abs}}(x) = \overline{v}^n_{\mathsf{abs}}(y)) \Rightarrow x = y$	DIA5	$G \mid (\surd_{\mathsf{d}} i \,.\, F) = \surd_{\mathsf{d}} i \,.\, (\overline{v}^i_{\mathsf{abs}}(G) \mid F)$	DIA14
$\sigma^n_{\mathsf{abs}}(\underline{a}) \cdot x = \sigma^n_{\mathsf{abs}}(\underline{a}) \cdot \overline{v}^n_{\mathsf{abs}}(x)$	DIA6	$\partial_H(\surd_{\mathsf{d}} i \,.\, F) = \surd_{\mathsf{d}} i \,.\, \partial_H(F)$	DIA15
$\sigma^n_{\mathsf{abs}}(\surd_{\mathsf{d}} i \,.\, F) = \sigma^n_{\mathsf{abs}}(F[0/i])$	DIA7		
$(\surd_{\mathsf{d}} i \,.\, F) + G = \surd_{\mathsf{d}} i \,.\, (F + \overline{v}^i_{\mathsf{abs}}(G))$	DIA8		
$(\surd_{\mathsf{d}} i \,.\, F) \cdot G = \surd_{\mathsf{d}} i \,.\, (F \cdot G)$	DIA9		

C Outline of proofs

Proof of Lemma 8 Lemma 8.1: it is easy to prove by induction on the structure of b that $b = \surd_{\mathsf{s}} v \,.\, \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(b)$. Lemma 8.2: this is Lemma 7 from Ref. [8] for the case with conditionals. Therefore, it suffices to extend the proof by induction on the structure of t with the case that t is of the form $b \mathbin{::\!\rightarrow} t'$:

$$
\begin{aligned}
& b \mathbin{::\!\rightarrow} t' \overset{\text{IH}}{=} b \mathbin{::\!\rightarrow} \surd_{\mathsf{s}} v \,.\, \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(t') \overset{\text{SASGC10}}{=} \\
& \surd_{\mathsf{s}} v \,.\, (\overline{v}^{v}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(t')) \overset{\text{A6ID},(*),\text{DISTR+}}{=} \\
& \surd_{\mathsf{s}} v \,.\, (\overline{v}^{v}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(t') + \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(\overset{\bullet}{\delta})) \overset{\text{Lemma 8.1,CSIA2,SI18}}{=} \\
& \surd_{\mathsf{s}} v \,.\, (\overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(t') + \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(\overset{\bullet}{\delta})) \overset{\text{SAI0,SAT1}}{=} \\
& \surd_{\mathsf{s}} v \,.\, (\overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(t') + \sigma^{\lfloor v \rfloor}_{\mathsf{abs}}(\overset{\bullet}{\delta})) \overset{\text{SASGC1}}{=} \surd_{\mathsf{s}} v \,.\, \overline{v}^{\lfloor v \rfloor}_{\mathsf{abs}}(b \mathbin{::\!\rightarrow} t')
\end{aligned}
$$

(*) We make use of the proof for the case that t is of the form $\overset{\bullet}{\delta}$. □

Proof of Lemma 9 Lemma 9 is Lemma 9 from Ref. [8] adapted to the case with conditionals. The condition $\overline{v}^{p}_{\mathsf{abs}}(t) \neq \sigma^{p}_{\mathsf{abs}}(\overset{\bullet}{\delta})$ needed in the case with conditionals implies the condition $t \neq \overset{\bullet}{\delta}$ used in Ref. [8]. There, observing that the lemma would follow immediately, we only proved by induction on the structure of t that there exists a t' such that: (1) $\overline{v}^{p}_{\mathsf{abs}}(t) = \sigma^{p}_{\mathsf{abs}}(t')$ and (2) if $p \in [0,1)$ and $\overline{v}^{p}_{\mathsf{abs}}(t) \neq \sigma^{p}_{\mathsf{abs}}(\overset{\bullet}{\delta})$, $t' = t' + \sigma^{1-p}_{\mathsf{abs}}(\overset{\bullet}{\delta})$. Here, it suffices to extend that proof with the case that t is of the form $b \mathbin{::\!\rightarrow} t'$:

$$
\begin{aligned}
(1)\ & \overline{v}^{p}_{\mathsf{abs}}(b \mathbin{::\!\rightarrow} t') \overset{\text{SASGC1}}{=} \overline{v}^{p}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} \overline{v}^{p}_{\mathsf{abs}}(t') + \sigma^{p}_{\mathsf{abs}}(\overset{\bullet}{\delta}) \overset{\text{IH}}{=} \\
& \overline{v}^{p}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} \sigma^{p}_{\mathsf{abs}}(t'') + \sigma^{p}_{\mathsf{abs}}(\overset{\bullet}{\delta}) \overset{\text{SASGC3}}{=} \\
& \surd_{\mathsf{s}} v \,.\, \sigma^{p}_{\mathsf{abs}}(\overline{v}^{v}_{\mathsf{abs}}(\overline{v}^{p}_{\mathsf{abs}}(b)) \mathbin{::\!\rightarrow} t'') \overset{\text{SI18,SIA4}}{=} \sigma^{p}_{\mathsf{abs}}(\overline{v}^{p}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} t'') \\
(2)\ & \overline{v}^{p}_{\mathsf{abs}}(b \mathbin{::\!\rightarrow} t') \neq \sigma^{p}_{\mathsf{abs}}(\overset{\bullet}{\delta}) \overset{\text{SGC1,SGC2ID,SASGC1}}{\Rightarrow} \\
& \overline{v}^{p}_{\mathsf{abs}}(b) = \mathsf{t} \text{ and } \overline{v}^{p}_{\mathsf{abs}}(t') \neq \sigma^{p}_{\mathsf{abs}}(\overset{\bullet}{\delta}) \\
& \text{By the induction hypothesis,} \\
& \overline{v}^{p}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} t'' = \overline{v}^{p}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} (t'' + \sigma^{1-p}_{\mathsf{abs}}(\overset{\bullet}{\delta})) \overset{\text{SGC1}}{=} \\
& \overline{v}^{p}_{\mathsf{abs}}(b) \mathbin{::\!\rightarrow} t'' + \sigma^{1-p}_{\mathsf{abs}}(\overset{\bullet}{\delta})
\end{aligned}
$$

□

Proof of Lemma 10 Lemma 10 is Lemma 10 from Ref. [8] adapted to the case with conditionals. The form $\surd_{\mathsf{s}} w \cdot \int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t') + \tilde{\delta})$ realizable in the case with conditionals generalizes the form $\int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t') + \tilde{\delta})$ obtained in Ref. [8]. Hence, it suffices to extend the proof by induction on the structure of t with the case that t is of the form $b :\!\!:\!\!\to t'$:

$$
\begin{aligned}
&\upsilon^1_{\mathsf{abs}}(b :\!\!:\!\!\to t' + \underline{\delta}) \overset{\text{SATO4,SASGC4}}{=} b :\!\!:\!\!\to \upsilon^1_{\mathsf{abs}}(t') + \upsilon^1_{\mathsf{abs}}(\underline{\delta}) \overset{\text{SGC1,6,BOOL4,6}}{=}\\
&b :\!\!:\!\!\to \upsilon^1_{\mathsf{abs}}(t') + b :\!\!:\!\!\to \upsilon^1_{\mathsf{abs}}(\underline{\delta}) + \upsilon^1_{\mathsf{abs}}(\underline{\delta}) \overset{\text{SATO0,3,6}}{=}\\
&b :\!\!:\!\!\to \upsilon^1_{\mathsf{abs}}(t') + b :\!\!:\!\!\to \upsilon^1_{\mathsf{abs}}(\underline{\delta}) + \underline{\delta} \overset{\text{SGC4,SATO4}}{=} b :\!\!:\!\!\to \upsilon^1_{\mathsf{abs}}(t' + \underline{\delta}) + \underline{\delta} \overset{\text{IH}}{=}\\
&b :\!\!:\!\!\to \int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t'') + \tilde{\delta}) + \underline{\delta} \overset{\text{SAT3,INT10}}{=}\\
&b :\!\!:\!\!\to \int_{v \in [0,1)} (\sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t'')) + \underline{\delta}) + \underline{\delta} \overset{\text{SGC1,6,BOOL4,6}}{=}\\
&b :\!\!:\!\!\to \int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t'')) + \underline{\delta} \overset{\text{SASGC9}}{=} \int_{v \in [0,1)} (b :\!\!:\!\!\to \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t''))) + \underline{\delta} \overset{\text{INT10}}{=}\\
&\int_{v \in [0,1)} (b :\!\!:\!\!\to \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t'')) + \sigma^v_{\mathsf{abs}}(\tilde{\delta})) \overset{\text{A6ID,SAT3}}{=}\\
&\int_{v \in [0,1)} (b :\!\!:\!\!\to \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(t'')) + \sigma^v_{\mathsf{abs}}(\dot{\delta}) + \sigma^v_{\mathsf{abs}}(\tilde{\delta})) \overset{\text{SASGC3,SIA4}}{=}\\
&\int_{v \in [0,1)} (\surd_{\mathsf{s}} w \cdot \sigma^v_{\mathsf{abs}}(\overline{\upsilon}^w_{\mathsf{abs}}(b) :\!\!:\!\!\to \nu_{\mathsf{abs}}(t'')) + \surd_{\mathsf{s}} w \cdot \sigma^v_{\mathsf{abs}}(\tilde{\delta})) \overset{\text{DISTR+,SAT3}}{=}\\
&\int_{v \in [0,1)} \surd_{\mathsf{s}} w \cdot \sigma^v_{\mathsf{abs}}(\overline{\upsilon}^w_{\mathsf{abs}}(b) :\!\!:\!\!\to \nu_{\mathsf{abs}}(t'') + \tilde{\delta}) \overset{\text{SASGC8,SIA17}}{=}\\
&\surd_{\mathsf{s}} w \cdot \int_{v \in [0,1)} \sigma^v_{\mathsf{abs}}(\nu_{\mathsf{abs}}(\overline{\upsilon}^w_{\mathsf{abs}}(b) :\!\!:\!\!\to t'') + \tilde{\delta})
\end{aligned}
$$

□

Proof of Theorem 11 Theorem 11 is Theorem 12 from Ref. [8] adapted to the case with conditionals. In Ref. [8], it is shown that the axioms of ACP$^{\text{dat}}\surd$ are derivable for closed terms from the axioms of ACP$^{\text{sat}}$I$\surd$ and the explicit definitions of the constants and operators $\underline{a}$, σ_{abs}, υ_{abs}, $\overline{\upsilon}_{\mathsf{abs}}$ and $\surd_{\mathsf{d}}$ (for processes) in Table 17. In Ref. [8], use is made of two lemmas that do not go through for the extension with conditionals, viz. Lemmas 9 and 10 from that paper. In the case with conditionals, Lemmas 9 and 10 from this paper have to be used instead. Fortunately, this requires only minor changes to the proofs for four axioms, viz. CM2DA, CM3DA, DACM3 and DACM4. What remains to be shown is that the additional axioms for conditionals are derivable for closed terms. This is nontrivial for the following axioms: CDAI3-CDAI7, CDIA1-CDIA8, DASGC2, DASGC3, DASGC8 and DASGC9. However, the proofs for most of these axioms are either similar to proofs for axioms of ACP$^{\text{dat}}\surd$ (CDIA1-CDIA8, DASGC8 and DASGC9) or simpler than most of those proofs (CDAI3-CDAI7 and DASGC3). Therefore, we only give an idea of the proofs.

The proofs for axioms CDAI3-CDAI7 require little effort. They involve short calculations using axioms BOOL1-BOOL7 and CSAI1-CSAI10.

The proofs for axioms CDIA1-CDIA5 are analogous to the proofs for DIA1-DIA5 in Ref. [8] – axioms CSIA1-CSIA5 are used instead of axioms SIA1-SIA5. The proof for axiom CDIA6 is similar to the proof for DIA10 in Ref. [8] – axiom CSIA6 is used instead of axiom SIA10.
The proof for axioms CDIA7 and CDIA8 are similar to the proof for DIA8 in Ref. [8] – axioms CSIA7 and CSIA8 are used instead of axiom SIA8. Distributivity of initial abstraction over $\wedge$ and $\vee$ is needed, but that can be derived as in the case of $+$.

The proof for axiom DASGC2 goes as follows. First, prove (1) $\mathsf{sl}(n+1) :\!:\!\rightarrow x = \int_{v \in [n,n+1)} (\mathsf{pt}(v) :\!:\!\rightarrow x)$, mainly by short calculations using axioms BOOL1-BOOL7 and CSAI1-CSAI10, and (2) $x = x + b :\!:\!\rightarrow x$, by application of axioms SGC1, SGC6 and BOOL4. Then, having proven equations (1) and (2), the proof for axiom DASGC2 involves mainly application of axiom SASGC2, these equations and the following immediate consequence of Lemma 8.2 and axiom SIA2: $\overline{v}^{p}_{\mathsf{abs}}(\overline{v}^{p}_{\mathsf{abs}}(t)) = \overline{v}^{p}_{\mathsf{abs}}(\overline{v}^{\lfloor p \rfloor}_{\mathsf{abs}}(t))$.

The proof for axiom DASGC3 is very easy. It consists of applying axiom SASGC3 and the following immediate consequence of Lemma 8.1 and axioms CSIA2 and SI18: $\overline{v}^{p}_{\mathsf{abs}}(b) = \overline{v}^{\lfloor p \rfloor}_{\mathsf{abs}}(b)$.

The proofs for axioms DASGC8 and DASGC9 are again similar to the proof for DIA8 – axioms SASGC10 and SASGC11 are used instead of axiom SIA8. Distributivity of initial abstraction over $:\!:\!\rightarrow$ is needed, but that can be derived as in the case of $+$.

References

[1] J.A. Bergstra, J.W. Klop, Process algebra for synchronous communication, Information and Control 60 (1984) 109–137.

[2] J.C.M. Baeten, W.P. Weijland, Process Algebra, Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press, 1990.

[3] J.C.M. Baeten, C. Verhoef, Concrete process algebra, in: S. Abramsky, D. Gabbay, T.S.E. Maibaum (Eds.), Handbook of Logic in Computer Science, Volume IV, Oxford University Press, 1995, pp. 149–268.

[4] R. Milner, A Calculus of Communicating Systems, LNCS 92, Springer-Verlag, 1980.

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

[6] S.D. Brookes, C.A.R. Hoare, A.W. Roscoe, A theory of communicating sequential processes, Journal of the ACM 31 (1984) 560–599.

[7] C.A.R. Hoare, Communicating Sequential Processes, Prentice-Hall, 1985.

[8] J.C.M. Baeten, C.A. Middelburg, Process algebra with timing: Real time and discrete time, in: J.A. Bergstra, A. Ponse, S.A. Smolka (Eds.), Handbook of Process Algebra, Elsevier, 2001, Ch. 10.

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

[10] J.C.M. Baeten, J.A. Bergstra, Real space process algebra, Formal Aspects of Computing 5 (6) (1993) 481–529.

[11] X. Nicollin, J. Sifakis, The algebra of timed processes ATP: Theory and application, Information and Computation 114 (1994) 131–178.

[12] L. Chen, An interleaving model for real-time systems, in: A. Nerode, M. Taitslin (Eds.), Symposium on Logical Foundations of Computer Science, LNCS 620, Springer-Verlag, 1992, pp. 81–92.

[13] F. Moller, C. Tofts, A temporal calculus of communicating systems, in: J.C.M. Baeten, J.W. Klop (Eds.), CONCUR'90, LNCS 458, Springer-Verlag, 1990, pp. 401–415.

[14] Wang Yi, Real-time behaviour of asynchronous agents, in: J.C.M. Baeten, J.W. Klop (Eds.), CONCUR'90, LNCS 458, Springer-Verlag, 1990, pp. 502–520.

[15] J.C.M. Baeten, J.A. Bergstra, Discrete time process algebra, Formal Aspects of Computing 8 (2) (1996) 188–208.

[16] G.D. Plotkin, A structural approach to operational semantics, Tech. Rep. DAIMI FN-19, University of Aarhus, Department of Computer Science (1981).

[17] A.S. Klusener, Completeness in real-time process algebra, in: J.C.M. Baeten, J.F. Groote (Eds.), CONCUR'91, LNCS 527, Springer-Verlag, 1991, pp. 376–392.

[18] J.A. Hillebrand, The ABP and CABP – a comparison of performances in real time process algebra, in: A. Ponse, C. Verhoef, S.F.M. van Vlijmen (Eds.), Algebra of Communicating Processes 1994, Workshop in Computing Series, Springer-Verlag, 1995, pp. 124–147.

[19] J.J. Vereijken, Fischer's protocol in timed process algebra, in: A. Ponse, C. Verhoef, S.F.M. van Vlijmen (Eds.), Algebra of Communicating Processes 1995, Report 95-14, Eindhoven University of Technology, Department of Computing Science, 1995, pp. 245–284.

[20] W.J. Fokkink and C. Verhoef. A conservative look at operational semantics with variable binding, Information and Computation, 146 (1998) 24–54.

[21] C.A. Middelburg, Variable binding operators in transition system specifications, Journal of Logic and Algebraic Programming 47 (2001) 15–45.

[22] J.F. Groote, J.J. van Wamel, Analysis of three hybrid systems in timed μCRL, Science of Computer Programming 39 (2001) 215–247.

[23] R. Alur, T.A. Henzinger, P.-H. Ho, Automatic symbolic verification of embedded systems, IEEE Transactions on Software Engineering 22 (1996) 181–201.

[24] J.C.M. Baeten, J.A. Bergstra, Discrete time process algebra: Absolute time, relative time and parametric time, Fundamenta Informaticae 29 (1/2) (1997) 51–76.