

Discrete Time Process Algebra with Silent Step

J.C.M. Baeten¹ and J.A. Bergstra² and M.A. Reniers¹

¹ Department of Mathematics and Computing Science, Eindhoven University of Technology,
P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands

² Programming Research Group, University of Amsterdam, Kruislaan 403, NL-1098 SJ Amsterdam, The Netherlands
and Department of Philosophy, Utrecht University, Heidelberglaan 8, NL-3584 CS Utrecht, The Netherlands

Abstract

The axiom system ACP of [10] was extended to discrete time in [6]. Here, we proceed to define the silent step in this theory in branching bisimulation semantics [7, 15] rather than weak bisimulation semantics [11, 20]. The version using relative timing is discussed extensively, versions using absolute and parametric timing are presented in brief. A term model and a graph model are presented and soundness and completeness results are given. The time free theories BPA_b and BPA_b^* are embedded in the discrete time theories. Examples of the use of the relative time theory are given by means of some calculations on communicating buffers.

Note: Partial support received from ESPRIT Basic Research Action 7166, CONCUR2. This paper supersedes [4].

1 Introduction

Process algebra was introduced by Milner in the form of CCS [19]. The original design of CCS and of subsequent versions of process algebra such as ACP [10] and TCSP [14] involves no explicit notion of time. Time is present in the interpretation of sequential composition: in $p \cdot q$ (ACP notation) the process p should be executed before q . Process algebras can be introduced that support standardized features to incorporate a quantitative view on time. Time may be represented by means of non-negative reals, and actions can be given time stamps. This line is followed in [1] for ACP, in [21] for CCS and in [23] for CSP.

A second option is to divide time in slices indexed by natural numbers, to have an implicit or explicit time stamping mechanism that determines for each action the time slice in which it occurs and to have a time order within each slice only. This line has been followed in ATP [22], a process algebra that adds time slicing to a version of ACP based on action prefixing rather than sequential composition. Further, [16] has extended ACP with time slices whereas [21] have added these features to CCS. Following [2, 6], we use the phrase *discrete time* process algebra if an enumeration of time slices is used.

The objective of this paper is to extend the discrete time process algebra of ACP as given in [6] with the silent step τ . Silent steps have been a cornerstone of CCS since its introduction. Milner used weak bisimulation to model processes with silent step. Van Glabbeek and Weijland [15] introduced branching bisimulation which also deals with silent step, but is slightly less abstract. This will allow a notion of abstraction. We mention that [18] has extended ACP_ρ , the real time ACP of [1], with silent steps. We present three views on discrete time process algebra with silent step: a version using relative timing (which we discuss extensively), a version using absolute timing, and a version using parametric timing, that integrates the relative and absolute timing versions.

There are many practical uses conceivable for timed process algebras. In particular, we mention the TOOLBUS (see [9]). This TOOLBUS contains a program notation called T which is syntactically sugared discrete time process algebra. Programs in T are called T -scripts. The runtime system is also described in terms of discrete time process algebra. By using randomized symbolic execution the TOOLBUS implementation enacts that the axioms of process algebra can be viewed as correctness preserving transformations of T -scripts. A comparable part of discrete time process algebra that is used to describe T -scripts has also been used for the description of ϕ SDL, flat SDL, a subset of SDL that leaves out modularization and concentrates on timing aspects (see [12]). Discrete-time process algebra with relative timing has also been used for the formal specification of the I^2C -bus [13].

At this point we are happy to express our admiration for Milner's contributions. In process theory, but not just there, he has guided the work in such directions, that the perspective of relevance and application is almost self evident.

2 Discrete Time Process Algebra with Relative Timing

2.1 Basic Process Algebra with Time Free Actions

The signature of $\text{BPA}_{\text{drt}}^-$ has constants $cts(a)$ (for $a \in A$), denoting a in the current time slice, and $cts(\delta)$ denoting a deadlock at the end of the current time slice. Also, we have the immediate deadlock constant $\hat{\delta}$ introduced in [6]. This constant denotes an immediate and catastrophic deadlock. Within a time slice, there is no explicit mention of the passage of time, we can see the passage to the next time slice as a clock tick. Thus, the $cts(a)$ can be called non-delayable actions: the action must occur before the next clock tick.

The operators are alternative and sequential composition, and the *relative discrete time unit delay* σ_{rel} (the notation σ taken from [17]). The process $\sigma_{\text{rel}}(x)$ will start x after one clock tick, i.e., in the next time slice. In addition, we add the auxiliary operator ν_{rel} . This operator, called the *current slice time out* operator, or current slice operator in short, disallows an initial time step, it gives the part of a process that starts with an action in the current time slice. It was introduced in [2] in a setting without immediate deadlock, there, the notation $x \gg_{\text{dt}} 1$ was used for $\nu_{\text{rel}}(x)$. (The Greek letter ν sounds like "now"; this correspondence is even stronger in Dutch.)

The axioms of $\text{BPA}_{\text{drt}}^-$ are A1-A5, A6ID, A7ID, DCS1-DCS4, DCSID, DRT1-DRT5, DRTSID, as given in Table 1. The axiom DRT1 is the time factorization axiom of ATP ([22]): it says that the passage of time by itself cannot determine a choice. The addition of a silent step in strong bisimulation semantics now just amounts to the presence of a new constant $cts(\tau)$ with the same axioms as the $cts(a)$ constants. We write $A_\tau = A \cup \{\tau\}$, $A_\delta = A \cup \{\delta\}$, etc. The axioms DRT3 and DRT5 are derivable from the other axioms (see [24]). Note that $x + cts(\delta) = x$ for all closed terms x except those that are derivably equal to $\hat{\delta}$ (see Proposition 2.1.7), which implies that in a theory without immediate deadlock the law $x + cts(\delta) = x$ will hold.

The standard process algebra BPA_δ can be considered as an SRM specification (Subalgebra of Reduced Model, in the terminology of [3]) of $\text{BPA}_{\text{drt}}^-$: consider the initial algebra of $\text{BPA}_{\text{drt}}^-$, reduce the signature by omitting $\hat{\delta}$, σ_{rel} , ν_{rel} , then BPA_δ is a complete axiomatization of the reduced model, under the interpretation of a , δ (from BPA_δ) by $cts(a)$, $cts(\delta)$.

However, this is not the embedding of the time free theory into the discrete time theory that we prefer: it reduces the whole world to one time slice. Rather, we propose to view the time free actions as actions that can occur in any time slice. Following [6], we extend $\text{BPA}_{\text{drt}}^-$ to BPA_{drt} by introducing constants $ats(a)$ (for $a \in A_{\tau\delta}$). The constant $ats(a)$ executes a in an arbitrary time slice, followed by immediate termination. We define these constants in axiom ARTS (Any Relative Time Slice, see Table 1) using the operator σ_{rel}^* , the *time iteration* operator: $\sigma_{\text{rel}}^*(x)$ can start the execution of x in the current time slice, and can always delay to the next time slice. The defining axiom for this operator takes the form of a recursive equation (see axiom DRTI1, Discrete Relative Time Iteration, in Table 1). The presentation using the time iteration operator here differs slightly from the presentation in [6]. There, we used the unbounded start delay operator $\lfloor \rfloor^\omega$ of [22] instead; the unbounded start delay operator can be simply expressed in terms of the time iteration operator as follows: $\lfloor x \rfloor^\omega = \sigma_{\text{rel}}^*(\nu_{\text{rel}}(x))$. In order to prove identities for the time iteration operator, we need a restricted form of the Recursive Specification Principle RSP (see, e.g. [7]). We call this principle RSP(DRT) (see Table 2).

We give a semantics for the theory BPA_{drt} in terms of Plotkin-style operational rules. We have the following relations on the set of closed process expressions $T(\text{BPA}_{\text{drt}})$:

- action step $\subseteq T(\text{BPA}_{\text{drt}}) \times A_\tau \times T(\text{BPA}_{\text{drt}})$, notation $p \xrightarrow{a} p'$ (denotes action execution);
- action termination $\subseteq T(\text{BPA}_{\text{drt}}) \times A_\tau$, notation $p \xrightarrow{a} \surd$ (execution of a terminating action);
- time step $\subseteq T(\text{BPA}_{\text{drt}}) \times T(\text{BPA}_{\text{drt}})$, notation $p \xrightarrow{\sigma} p'$ (denotes passage to the next time slice);
- immediate deadlock $\subseteq T(\text{BPA}_{\text{drt}})$, notation $\text{ID}(p)$ (immediate deadlock, holds only for process expressions equal to $\hat{\delta}$).

We enforce the time factorization axiom DRT1 by phrasing the rules so that each process expression has at most one σ -step: in a transition system, each node has at most one outgoing σ -edge. The operational semantics in Table 3 uses predicates and negative premises. Still, using terminology and results of [25], the rules satisfy the *panth* format, and determine a unique transition relation on closed process expressions. Due to the presence of the immediate deadlock constant we have a notion of bisimulation which is slightly

$x + y = y + x$	A1	$x + \dot{\delta} = x$	A6ID
$(x + y) + z = x + (y + z)$	A2	$\dot{\delta} \cdot x = \dot{\delta}$	A7ID
$x + x = x$	A3		
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$\sigma_{\text{rel}}(x) + \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x + y)$	DRT1
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\sigma_{\text{rel}}(x) \cdot y = \sigma_{\text{rel}}(x \cdot y)$	DRT2
		$cts(\delta) \cdot x = cts(\delta)$	DRT3
$\nu_{\text{rel}}(cts(a)) = cts(a)$	DCS1	$cts(a) + cts(\delta) = cts(a)$	DRT4
$\nu_{\text{rel}}(x + y) = \nu_{\text{rel}}(x) + \nu_{\text{rel}}(y)$	DCS2	$\sigma_{\text{rel}}(x) + cts(\delta) = \sigma_{\text{rel}}(x)$	DRT5
$\nu_{\text{rel}}(x \cdot y) = \nu_{\text{rel}}(x) \cdot y$	DCS3	$\sigma_{\text{rel}}(\dot{\delta}) = cts(\delta)$	DRTSID
$\nu_{\text{rel}}(\sigma_{\text{rel}}(x)) = cts(\delta)$	DCS4	$\sigma_{\text{rel}}^*(x) = x + \sigma_{\text{rel}}(\sigma_{\text{rel}}^*(x))$	DRTI1
$\nu_{\text{rel}}(\dot{\delta}) = \dot{\delta}$	DCSID	$ats(a) = \sigma_{\text{rel}}^*(cts(a))$	ARTS

Table 1: Axioms of BPA_{drt} ($a \in A_{\tau\delta}$).

$$y = x + \sigma_{\text{rel}}(y) \implies y = \sigma_{\text{rel}}^*(x) \quad \text{RSP(DRT)}$$

Table 2: RSP(DRT).

$\text{ID}(\dot{\delta})$	$\frac{\text{ID}(x), \text{ID}(y)}{\text{ID}(x + y)}$	$\frac{\text{ID}(x)}{\text{ID}(x \cdot y)}$	$\frac{\text{ID}(x)}{\text{ID}(\nu_{\text{rel}}(x))}$
$cts(a) \xrightarrow{a} \checkmark$	$ats(a) \xrightarrow{a} \checkmark$	$ats(a) \xrightarrow{\sigma} ats(a)$	$ats(\delta) \xrightarrow{\sigma} ats(\delta)$
$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y}$	$\frac{x \xrightarrow{a} \checkmark}{x \cdot y \xrightarrow{a} y}$	$\frac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y}$	
$\frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x', y + x \xrightarrow{a} x'}$	$\frac{x \xrightarrow{a} \checkmark}{x + y \xrightarrow{a} \checkmark, y + x \xrightarrow{a} \checkmark}$		
$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} y'}{x + y \xrightarrow{\sigma} x' + y'}$	$\frac{x \xrightarrow{\sigma} x', y \xrightarrow{\sigma} \cancel{y'}}{x + y \xrightarrow{\sigma} x', y + x \xrightarrow{\sigma} x'}$		
$\frac{\neg \text{ID}(x)}{\sigma_{\text{rel}}(x) \xrightarrow{\sigma} x}$	$\frac{x \xrightarrow{a} x'}{\nu_{\text{rel}}(x) \xrightarrow{a} x'}$	$\frac{x \xrightarrow{a} \checkmark}{\nu_{\text{rel}}(x) \xrightarrow{a} \checkmark}$	
$\frac{x \xrightarrow{a} x'}{\sigma_{\text{rel}}^*(x) \xrightarrow{a} x'}$	$\frac{x \xrightarrow{a} \checkmark}{\sigma_{\text{rel}}^*(x) \xrightarrow{a} \checkmark}$	$\frac{x \xrightarrow{\sigma} x'}{\sigma_{\text{rel}}^*(x) \xrightarrow{\sigma} x' + \sigma_{\text{rel}}^*(x)}$	$\frac{x \xrightarrow{\sigma} \cancel{x}}{\sigma_{\text{rel}}^*(x) \xrightarrow{\sigma} \sigma_{\text{rel}}^*(x)}$

Table 3: Operational rules for BPA_{drt} ($a \in A_{\tau}$).

different from the usual *strong bisimulation*. This notion is called *strong tail bisimulation* in this paper. In the terminology of [25] this notion of strong tail bisimulation would be called a bisimulation anyway.

Definition 2.1.1 (Strong tail bisimulation) Two closed terms p and q are called *strongly tail bisimilar*, notation $p \Leftrightarrow q$, if there exists a symmetric binary relation R on closed terms, called *strong tail bisimulation*, relating p and q such that for all r, s with $R(r, s)$ we have:

1. if $r \xrightarrow{u} r' (u \in A_{\tau\sigma})$, then there exists a node s' such that $s \xrightarrow{u} s'$ and $R(r', s')$;
2. if $r \xrightarrow{a} \surd (a \in A_\tau)$, then $s \xrightarrow{a} \surd$;
3. if $\text{ID}(r)$, then $\text{ID}(s)$.

We state that strong tail bisimulation is a congruence with respect to the operators from the algebra BPA_{drt} . Following [24] we can prove that BPA_{drt} is a sound and complete axiomatization of the model of closed process expressions modulo strong tail bisimulation equivalence.

Now, we want to define a notion of bisimulation that takes into account the special status of the immediate deadlock and the silent step. We start from the definition of branching bisimulation in the time free case, and adapt this to our timed setting. An immediate deadlock term can only be related to another immediate deadlock term. As a consequence, we can have no term that is related to \surd . This is different from the usual definition of branching bisimulation of [7, 15]. In order to emphasize this fact, we call the relations to be defined branching *tail* bisimulations. The following definition is easier to read if we use a couple of abbreviations: first, we write $p \xrightarrow{(u)} p'$ if either $p \xrightarrow{u} p'$ or $u \equiv \tau$ and $p = p'$. Further, we write $p \Longrightarrow q$ if it is possible to reach q from p by executing a number of τ -steps (0 or more).

Definition 2.1.2 (Branching tail bisimulation) Two closed terms p and q are called *branching tail bisimilar*, notation $p \Leftrightarrow_{\text{bt}} q$, if there exists a symmetric binary relation R on closed terms, called *branching tail bisimulation*, relating p and q such that for all r, s with $R(r, s)$ we have:

1. if $r \xrightarrow{u} r' (u \in A_{\tau\sigma})$, then there exist s^*, s' such that $s \Longrightarrow s^* \xrightarrow{(u)} s'$ and $R(r, s^*)$ and $R(r', s')$;
2. if $r \xrightarrow{a} \surd (a \in A_\tau)$, then there exists s^* such that $s \Longrightarrow s^* \xrightarrow{a} \surd$ and $R(r, s^*)$;
3. if $\text{ID}(r)$, then $\text{ID}(s)$.

Actually, we gave the definition of *semi*-branching bisimulation here, as optimized in [8]. The present definition is shorter and easier to work with than the original definition in [15], and induces the same equivalence relation (see [8]).

Definition 2.1.3 (Rooted branching tail bisimulation) If R is a branching tail bisimulation, then we say that the related pair (p, q) satisfies the *root condition* if:

1. if $p \xrightarrow{u} p' (u \in A_{\tau\sigma})$, then there exists q' such that $q \xrightarrow{u} q'$ and $R(p', q')$;
2. if $p \xrightarrow{a} \surd (a \in A_\tau)$, then $q \xrightarrow{a} \surd$.

A term r is called σ -reachable from a term p iff there exists a $k \in \mathbb{N}$ such that $p \equiv p_0 \xrightarrow{\sigma} p_1 \xrightarrow{\sigma} \dots \xrightarrow{\sigma} p_k \equiv r$.

Two terms p and q are called *rooted branching tail bisimilar*, notation $p \Leftrightarrow_{\text{rbt}} q$, if there is a branching tail bisimulation relation R relating p and q such that whenever $R(r, s)$ and r is σ -reachable from p , then the pair (r, s) satisfies the root condition.

Thus the root condition amounts to the condition that the bisimulation is strong as long as no action is executed.

We can prove that rooted branching tail bisimulation is a congruence relation with respect to the operators from the algebra BPA_{drt} , thus obtaining the algebra $T(\text{BPA}_{\text{drt}})/\Leftrightarrow_{\text{rbt}}$ of closed process expressions modulo rooted branching tail bisimulation. We can establish that the algebras $T(\text{BPA}_{\text{drt}})/\Leftrightarrow$, $T(\text{BPA}_{\text{drt}})/\Leftrightarrow_{\text{bt}}$, and $T(\text{BPA}_{\text{drt}})/\Leftrightarrow_{\text{rbt}}$ satisfy all laws of $\text{BPA}_{\text{drt}}+\text{RSP}(\text{DRT})$.

Theorem 2.1.4 (Soundness) *The laws of $\text{BPA}_{\text{drt}}+\text{RSP}(\text{DRT})$ are valid in $T(\text{BPA}_{\text{drt}})/\Leftrightarrow$, $T(\text{BPA}_{\text{drt}})/\Leftrightarrow_{\text{bt}}$, and $T(\text{BPA}_{\text{drt}})/\Leftrightarrow_{\text{rbt}}$.*

Note that if we take the reduced model obtained by omitting the immediate deadlock constant, then we can define a notion of branching bisimulation without the tail condition, where a term can be related to \surd . Using the embedding of discrete time process algebra into real time process algebra given in [2, 6] we find that our notion of silent step in time is in line with the notion of timed branching bisimulation of [18].

Proposition 2.1.5 *The following identities are derivable from $BPA_{drt} + RSP(DRT)$:*

$\sigma_{rel}^*(x + y) = \sigma_{rel}^*(x) + \sigma_{rel}^*(y)$	DRTI2
$\sigma_{rel}^*(x \cdot y) = \sigma_{rel}^*(x) \cdot y$	DRTI3
$\sigma_{rel}^*(\sigma_{rel}(x)) = \sigma_{rel}(\sigma_{rel}^*(x))$	DRTI4
$\sigma_{rel}^*(\sigma_{rel}^*(x)) = \sigma_{rel}^*(x)$	DRTI5
$\sigma_{rel}^*(\delta) = ats(\delta)$	DRTIID
$\nu_{rel}(\sigma_{rel}^*(x)) = \nu_{rel}(x) + cts(\delta)$	DCSTI

Table 4: Derivable equations.

As it turns out these identities are used in the proof of the completeness theorem. By using those, the principle RSP(DRT) does not have to be used directly, but only in the proof of these identities. As a matter of fact we prove completeness for the algebra $BPA_{drt}\tau + DRTI2$ -DRTI5, DRTIID, DCSTI. The most important place where those identities are used is the Elimination Theorem. This theorem expresses that every closed term can be rewritten into a *basic term*, i.e., a closed term with a more restricted syntax. We define basic terms over BPA_{drt} in such a way that they closely resemble the process graphs (see Section 3), in the sense that in every sum-context there is at most one summand ready to perform a time step, and that time iteration does not occur in any sum-context. To achieve this we define auxiliary sets of basic terms $\nu(BPA_{drt})$, $\Sigma(BPA_{drt})$, and $\Delta(BPA_{drt})$.

Definition 2.1.6 (Basic terms) The set $B(BPA_{drt})$ of *basic terms* over BPA_{drt} is defined inductively by:

1. $\delta \in \nu(BPA_{drt})$;
2. $a \in A_{\tau\delta} \Rightarrow cts(a) \in \nu(BPA_{drt})$;
3. $a \in A_{\tau} \wedge p \in B(BPA_{drt}) \Rightarrow cts(a) \cdot p \in \nu(BPA_{drt})$;
4. $p, q \in \nu(BPA_{drt}) \wedge p \not\equiv \delta \wedge q \not\equiv \delta \Rightarrow p + q \in \nu(BPA_{drt})$;
5. $p \in B(BPA_{drt}) \wedge p \not\equiv \delta \Rightarrow \sigma_{rel}(p) \in \Sigma(BPA_{drt})$;
6. $p \in \nu(BPA_{drt}) \wedge p \not\equiv \delta \Rightarrow \sigma_{rel}^*(p) \in \Delta(BPA_{drt})$;
7. $\nu(BPA_{drt}) \subseteq B(BPA_{drt})$;
8. $\Sigma(BPA_{drt}) \subseteq B(BPA_{drt})$;
9. $\Delta(BPA_{drt}) \subseteq B(BPA_{drt})$;
10. $p \in \nu(BPA_{drt}) \wedge p \not\equiv \delta \wedge q \in \Sigma(BPA_{drt}) \Rightarrow p + q, q + p \in B(BPA_{drt})$.

With this definition of basic terms the normal form of $cts(b) + ats(a)$ is given by the term $(cts(b) + cts(a)) + \sigma_{rel}(\sigma_{rel}^*(cts(a)))$.

Proposition 2.1.7 *For closed BPA_{drt} -terms x such that $BPA_{drt} \not\vdash x = \delta$ we have $BPA_{drt} \vdash x + cts(\delta) = x$.*

Theorem 2.1.8 (Elimination) *For every closed BPA_{drt} -term p there exists a basic term q such that*

$$\begin{array}{l} A1, A2, A4, A5, A6ID, A7ID, DRT1-DRT3, DRTSID, DCS1-DCS4, DCSTI, \\ ARTS, DRTII-DRTI5, DRTIID, DCSTI \end{array} \vdash p = q.$$

Note that usually we are less careful with respect to the axioms used for the Elimination Theorem, we just write $BPA_{drt} \vdash p = q$. However, with the upcoming proof of completeness in mind (see Section 4), we explicitly list the axioms used in the Elimination Theorem.

Theorem 2.1.9 (Completeness) BPA_{drt} is a complete axiomatization of strong tail bisimulation on closed BPA_{drt} terms.

Now, the process algebra BPA_δ can be considered as an SRM specification of BPA_{drt} : reduce the signature of the initial algebra of BPA_{drt} by omitting δ , σ_{rel} , ν_{rel} , $cts(a)$ (for $a \in A_{\tau\delta}$), and σ_{rel}^* , then BPA_δ is a complete axiomatization of the reduced model, with the interpretation of a by $ats(a)$.

2.2 Axiomatization for Silent Step

Now we want to formulate algebraic laws for the silent step, that hold in the algebra $T(BPA_{drt})/\simeq_{rbt}$. We cannot just transpose the well-known laws, because of the special status of σ -steps. An example: $cts(a) \cdot (cts(\tau) \cdot (\sigma_{rel}(cts(b)) + \sigma_{rel}(cts(c))) + \sigma_{rel}(cts(b)))$ is not rooted branching tail bisimulation equivalent to $cts(a) \cdot (\sigma_{rel}(cts(b)) + \sigma_{rel}(cts(c)))$, as in the first term, the choice for b can be made by the execution of the σ -step, and in the second term, the choice must be made after the σ -step (however the second term is equal to $cts(a) \cdot \sigma_{rel}(cts(b) + cts(c))$). In order to ensure that we do not split terms that both have an initial σ -step, we use the current slice operator ν_{rel} . In Table 5, DRTB1 and DRTB2 are variants of the branching law B2: $x \cdot (\tau \cdot (y + z) + y) = x \cdot (y + z)$. In DRTB1, we have the case where the ‘loose’ term (y in B2) does not have an initial time step, in DRTB2 the other term (z in B2) does not have an initial time step.

We add $cts(\delta)$ to ensure that the expression following $cts(\tau)$ does not equal δ . A simple instance is the identity $x \cdot cts(\tau) \cdot cts(\tau) = x \cdot cts(\tau)$. However, we do not have the law $x \cdot cts(\tau) = x$ (the counterpart of the branching law B1), as $cts(a) \cdot cts(\tau) \cdot \delta$ must be distinguished from $cts(a) \cdot \delta$ (this can be appreciated in a setting with parallel composition: the first term allows execution of actions in the current time slice from a parallel component after the execution of a , the second term does not).

In [4] the conditional axiom $cts(a) \cdot x = cts(a) \cdot y \Rightarrow cts(a) \cdot (\sigma_{rel}(x) + \nu_{rel}(z)) = cts(a) \cdot (\sigma_{rel}(y) + \nu_{rel}(z))$ was introduced to omit a τ -step following one or more time steps. In this paper we replace this conditional axiom by DRTB3 (see Table 5). A simple instance is $cts(a) \cdot \sigma_{rel}(cts(\tau) \cdot cts(b)) = cts(a) \cdot \sigma_{rel}(cts(b))$.

We have one additional axiom for the constant $ats(\tau)$. In axiom DRTB4 it appears as $\sigma_{rel}^*(cts(\tau))$. Axiom DRTB4 can also be given as

$$x \cdot (ats(\tau) \cdot \sigma_{rel}^*(\nu_{rel}(y) + \nu_{rel}(z) + cts(\delta)) + \sigma_{rel}^*(\nu_{rel}(y))) = x \cdot \sigma_{rel}^*(\nu_{rel}(y) + \nu_{rel}(z) + cts(\delta))$$

by replacing $\sigma_{rel}^*(cts(\tau))$ by $ats(\tau)$ and using DRTI2, DRTI3, or as

$$x \cdot (ats(\tau) \cdot \lfloor y + z + cts(\delta) \rfloor^\omega + \lfloor y \rfloor^\omega) = x \cdot \lfloor y + z + cts(\delta) \rfloor^\omega$$

by also replacing $\sigma_{rel}^*(\nu_{rel}(y))$ by $\lfloor y \rfloor^\omega$, etc.

$x \cdot (cts(\tau) \cdot (\nu_{rel}(y) + z + cts(\delta)) + \nu_{rel}(y)) = x \cdot (\nu_{rel}(y) + z + cts(\delta))$	DRTB1
$x \cdot (cts(\tau) \cdot (\nu_{rel}(y) + z + cts(\delta)) + z) = x \cdot (\nu_{rel}(y) + z + cts(\delta))$	DRTB2
$x \cdot (\sigma_{rel}(cts(\tau) \cdot (y + cts(\delta))) + \nu_{rel}(z)) = x \cdot (\sigma_{rel}(y + cts(\delta)) + \nu_{rel}(z))$	DRTB3
$x \cdot \sigma_{rel}^*(cts(\tau) \cdot \sigma_{rel}^*(\nu_{rel}(y) + \nu_{rel}(z) + cts(\delta)) + \nu_{rel}(y))$	
$= x \cdot \sigma_{rel}^*(\nu_{rel}(y) + \nu_{rel}(z) + cts(\delta))$	DRTB4

Table 5: $BPA_{drt}\tau = BPA_{drt} + \text{DRTB1-4}$.

We find that rooted branching tail bisimulation is a congruence with respect to the operators from the algebra $BPA_{drt}\tau$, and that the laws of $BPA_{drt}\tau + \text{RSP}(DRT)$ are valid in $T(BPA_{drt}\tau)/\simeq_{rbt}$. Basic terms for $BPA_{drt}\tau$ are defined in the same way as basic terms for BPA_{drt} .

Theorem 2.2.1 (Soundness) *The laws of $BPA_{drt}\tau + \text{RSP}(DRT)$ are valid in $T(BPA_{drt}\tau)/\simeq_{rbt}$.*

Before we introduce a number of useful axioms for the silent step, we first define an auxiliary operator σ_{rel}^m . The term $\sigma_{rel}^m(x)$ represents the operator σ_{rel} applied to x m times, i.e., for $m \geq 0$:

$$\begin{aligned}\sigma_{rel}^0(x) &= x \\ \sigma_{rel}^{m+1}(x) &= \sigma_{rel}(\sigma_{rel}^m(x))\end{aligned}$$

The generalizations of the axioms for silent step are useful in the completeness proof in Section 4.

Proposition 2.2.2 (Other axioms for abstraction) *For closed $BPA_{drt\tau}$ -terms p, x, y, z , and $m \geq 0$, we have*

1. $BPA_{drt\tau} \vdash x \cdot \sigma_{rel}^m(cts(\tau) \cdot (v_{rel}(y) + z + cts(\delta)) + v_{rel}(y)) = x \cdot \sigma_{rel}^m(v_{rel}(y) + z + cts(\delta));$
2. $BPA_{drt\tau} \vdash x \cdot \sigma_{rel}^m(cts(\tau) \cdot (v_{rel}(y) + z + cts(\delta)) + z) = x \cdot \sigma_{rel}^m(v_{rel}(y) + z + cts(\delta));$
3. $BPA_{drt\tau} \vdash x \cdot \sigma_{rel}^m(\sigma_{rel}(cts(\tau) \cdot (v_{rel}(y) + z + cts(\delta)) + v_{rel}(y)) + v_{rel}(p))$
 $= x \cdot \sigma_{rel}^m(\sigma_{rel}(v_{rel}(y) + z + cts(\delta)) + v_{rel}(p));$
4. $BPA_{drt\tau} \vdash x \cdot \sigma_{rel}^m(\sigma_{rel}(cts(\tau) \cdot (v_{rel}(y) + z + cts(\delta)) + z) + v_{rel}(p))$
 $= x \cdot \sigma_{rel}^m(\sigma_{rel}(v_{rel}(y) + z + cts(\delta)) + v_{rel}(p));$
5. $BPA_{drt\tau} \vdash x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(cts(\tau) \cdot \sigma_{rel}^*(v_{rel}(y) + v_{rel}(z) + cts(\delta)) + v_{rel}(y))) + v_{rel}(p))$
 $= x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(v_{rel}(y) + v_{rel}(z) + cts(\delta))) + v_{rel}(p)).$

Let us consider the embedding of the time free theory BPA_{δ}^{τ} in $BPA_{drt\tau}$. If we reduce the initial algebra of $BPA_{drt\tau}$ by reducing the signature by omitting δ , $ats(a)$ (for $a \in A_{\tau\delta}$), σ_{rel} , v_{rel} , σ_{rel}^* , and interpret a by $cts(a)$, we do not obtain BPA_{δ}^{τ} of [7, 15]. The first branching law $x \cdot \tau = x$ will not hold, but instead $x \cdot \tau \cdot y = x \cdot y$. We can nevertheless obtain BPA_{δ}^{τ} as an SRM specification as follows. First we add constants $ctstau(a)$ (for $a \in A_{\tau\delta}$) defined by $ctstau(a) = cts(a) \cdot cts(\tau)$. Then, by omitting δ , $cts(a)$, $ats(a)$, σ_{rel} , v_{rel} , σ_{rel}^* and interpreting a as $ctstau(a)$, BPA_{δ}^{τ} becomes an SRM specification of $BPA_{drt\tau}$. The following proposition shows that analogues of DRTB1 and DRTB2 and in addition the first τ -law are valid for the new constants.

Proposition 2.2.3 *The following laws are derivable from $BPA_{drt\tau}$:*

1. $BPA_{drt\tau} \vdash ctstau(a) \cdot ctstau(\tau) = ctstau(a);$
2. $BPA_{drt\tau} \vdash x \cdot (ctstau(\tau) \cdot (v_{rel}(y) + z + cts(\delta)) + v_{rel}(y)) = x \cdot (v_{rel}(y) + z + cts(\delta));$
3. $BPA_{drt\tau} \vdash x \cdot (ctstau(\tau) \cdot (v_{rel}(y) + z + cts(\delta)) + z) = x \cdot (v_{rel}(y) + z + cts(\delta)).$

As a consequence, we can prove $x \cdot ctstau(\tau) = x$ for all closed terms. Thus, we have embedded the time free theory into the relative discrete time theory. Again, this is not the embedding of the time free theory into the discrete time theory that we prefer: the whole world is reduced to one time slice.

We cannot obtain the time free theory BPA_{δ}^{τ} as an SRM specification by interpreting a by $ats(a)$, since $ats(a) \cdot ats(\tau)$ is not branching tail bisimilar to $ats(a)$: the first term can still perform time steps after executing the action. Note that the second branching law is valid, as time iteration is the identity on all time free processes. In order to achieve an SRM specification, nonetheless, we will define a different interpretation of time free atoms in our timed setting. We define constants $atstau(a)$ as follows ($a \in A_{\tau\delta}$): $atstau(a) = ats(a) \cdot ats(\tau)$. If we reduce the initial algebra of $BPA_{drt\tau}$, with the additional constants $atstau(a)$, by omitting δ , $cts(a)$, $ats(a)$ (for $a \in A_{\tau\delta}$), σ_{rel} , v_{rel} , σ_{rel}^* , and we interpret a by $atstau(a)$, we do obtain BPA_{δ}^{τ} as an SRM specification. The following proposition shows that the analogue of DRTB4 and in addition the first τ -law are valid for the new constants.

Proposition 2.2.4 *The following laws are derivable from $BPA_{drt\tau}$ for $a \in A_{\tau}$:*

1. $atstau(a) \cdot atstau(\tau) = atstau(a);$
2. $x \cdot (atstau(\tau) \cdot \sigma_{rel}^*(v_{rel}(y + z)) + \sigma_{rel}^*(v_{rel}(y))) = x \cdot \sigma_{rel}^*(v_{rel}(y + z)).$

3 A graph model for $\text{BPA}_{\text{drt}}\tau$

In Section 4, we will prove that the present axiomatization of silent step is complete for the model of closed terms modulo rooted branching tail bisimulation. We do this by showing completeness for a model that is isomorphic to $T(\text{BPA}_{\text{drt}})/\cong_{\text{rbt}}$, namely a model consisting of process graphs.

In this section we proceed to define this graph model for $\text{BPA}_{\text{drt}}\tau$. An equivalence relation, called rooted branching tail bisimulation, is defined on the set of process graphs. Operators resembling the operators of the algebra both in name and in intention are defined on process graphs. Finally, we introduce an equivalent characterization of rooted branching tail bisimilarity in Section 3.4. This characterization of rooted branching tail bisimulation will be used in the completeness proof in Section 4.

3.1 Process graphs

We define a set of process graphs as in [7] with labels from $A_{\tau\sigma}$ satisfying the extra condition that every node has *at most one* outgoing σ -labeled edge. End-nodes can be labeled with a label \downarrow (for successful termination) or a label ID (for immediate deadlock). A σ -labeled edge may not lead to a successful termination node or an ID-labeled termination node.

Definition 3.1.1 A *process graph* is a quintuple $\langle N, E, r, \downarrow, \text{ID} \rangle$, where N is a non-empty set of nodes, $E \subseteq N \times A_{\tau\sigma} \times N$ is a set of labeled edges, $r \in N$ is the root node, $\downarrow \subseteq N$ and $\text{ID} \subseteq N$ are disjoint sets of successful termination nodes and immediate deadlock nodes respectively. The following criteria must hold for every process graph:

- every node can have at most one outgoing σ -edge (time factorization): for all $n \in N$, if there exist nodes s and t such that $n \xrightarrow{\sigma} s$ and $n \xrightarrow{\sigma} t$, then $s = t$;
- a successful termination or an immediate deadlock node cannot have an outgoing edge: for all $n \in \downarrow \cup \text{ID}$: $\{n \xrightarrow{u} s \mid u \in A_{\tau\sigma} \wedge s \in N\} = \emptyset$;
- a successful termination or an immediate deadlock node cannot have an incoming σ -edge: for all $n \in \downarrow \cup \text{ID}$, there does not exist a node $s \in N$ such that $s \xrightarrow{\sigma} n \in E$.

Actually, for our purposes we will only need graphs that are interpretations of closed terms. Such graphs always have finite node sets, and the only cycles that occur are of the form $s \xrightarrow{\sigma} s$. The class of all such process graphs is denoted by \mathbf{G} . We obtain the set of process graphs \mathbf{G}^+ by requiring in addition that the root node cannot have a successful termination label, i.e., $r \notin \downarrow$. For a process graph $g = \langle N, E, r, \downarrow, \text{ID} \rangle$ and a node $s \in N$, the subgraph of g with s as its root node, notation $(g)_s$, is defined as follows: $(g)_s = \langle N, E, s, \downarrow, \text{ID} \rangle$, where it is understood that only the nodes and edges that are reachable from s are relevant. For a set S and $E \subseteq N \times A_{\tau\sigma} \times N$, we define $S \otimes E = \{(s, n_1) \xrightarrow{u} (s, n_2) \mid s \in S \wedge n_1 \xrightarrow{u} n_2 \in E\}$. For a node $s \in N$ and $E \subseteq N \times A_{\tau\sigma} \times N$, we define $E[s \mapsto t]$ to be the set E with all occurrences of s replaced by node t . This notation is extended to replace nodes from a set S simultaneously as follows: $E[s \in S \mapsto t]$. Note that s can occur in t .

3.2 Tail Bisimulations

Definition 3.2.1 Two graphs $g = \langle N_g, E_g, r_g, \downarrow_g, \text{ID}_g \rangle$ and $h = \langle N_h, E_h, r_h, \downarrow_h, \text{ID}_h \rangle$ are called *strongly tail bisimilar*, notation $g \cong h$, if there exists a symmetric binary relation R , called *strong tail bisimulation*, on the nodes of g and h such that for all nodes r, s with $R(r, s)$ we have:

1. the roots of g and h are related, i.e., $R(r_g, r_h)$;
2. if $r \xrightarrow{u} r' \in E_g \cup E_h$ ($u \in A_{\tau\sigma}$), then there exists a node s' such that $s \xrightarrow{u} s' \in E_g \cup E_h$ and $R(r', s')$;
3. if $r \in \downarrow_g \cup \downarrow_h$, then $s \in \downarrow_g \cup \downarrow_h$;
4. if $r \in \text{ID}_g \cup \text{ID}_h$, then $s \in \text{ID}_g \cup \text{ID}_h$.

Definition 3.2.2 Two graphs g and h are called *branching tail bisimilar*, notation $g \cong_{\text{bt}} h$, if there exists a symmetric binary relation R , called *branching tail bisimulation*, on the nodes of g and h such that for all nodes r, s with $R(r, s)$ we have:

1. the roots of g and h are related;
2. if $r \xrightarrow{u} r' \in E_g \cup E_h$ ($u \in A_{\tau\sigma}$), then there exist nodes s^*, s' such that $s \implies s^* \xrightarrow{(u)} s' \in E_g \cup E_h$ and $R(r, s^*)$ and $R(r', s')$;
3. if $r \in \downarrow_g \cup \downarrow_h$, then $s \in \downarrow_g \cup \downarrow_h$;
4. if $r \in \text{ID}_g \cup \text{ID}_h$, then $s \in \text{ID}_g \cup \text{ID}_h$.

Definition 3.2.3 If R is a branching tail bisimulation, then we say that the related pair (r, s) satisfies the *root condition* iff whenever $r \xrightarrow{u} r' \in E_g \cup E_h$ ($u \in A_{\tau\sigma}$), then there exists a node s' such that $s \xrightarrow{u} s' \in E_g \cup E_h$ and $R(r', s')$.

Two graphs g and h are called *rooted branching tail bisimilar*, notation $g \Leftrightarrow_{\text{rbt}} h$, if there is a branching tail bisimulation relation R between the nodes of g and h such that if $R(r, s)$ and r is σ -reachable from the root of g or s is σ -reachable from the root of h , then the pair (r, s) satisfies the root condition.

3.3 Isomorphism of the term model and the graph model

In this section we will define mappings *graph* and *term* which associate a process graph to a closed term and a closed term to a graph respectively. These mappings are such that they are each other's inverse modulo strong tail bisimulation. This results in a theorem that states that the term deduction model and the graph model are isomorphic. As a consequence the soundness result can be transferred from the term model to the graph model and the completeness result to come can be transformed from the graph model to the term model.

In the following definitions we define operators on the graph model which are both in appearance and in intention similar to the operators of the algebra. In these definitions we assume that the sets of nodes of the graphs to be composed are disjoint. This assumption is fair since we are only interested in process graphs modulo graph isomorphism (see Section 3.4 for a definition of graph isomorphism).

Definition 3.3.1 The following constants are defined on the graph model, for $a \in A_\tau$:

$$\begin{aligned}
\hat{\delta} &= \langle \{r\}, \emptyset, r, \emptyset, \{r\} \rangle; \\
cts(\delta) &= \langle \{r\}, \emptyset, r, \emptyset, \emptyset \rangle; \\
cts(a) &= \langle \{r, s\}, \{r \xrightarrow{a} s\}, r, \{s\}, \emptyset \rangle; \\
ats(\delta) &= \langle \{r\}, \{r \xrightarrow{\sigma} r\}, r, \emptyset, \emptyset \rangle; \\
ats(a) &= \langle \{r, s\}, \{r \xrightarrow{\sigma} r, r \xrightarrow{a} s\}, r, \{s\}, \emptyset \rangle.
\end{aligned}$$

Definition 3.3.2 Let $g = \langle N, E, r, \downarrow, \text{ID} \rangle$. Assume that $\hat{r} \notin N$. Then $\sigma_{\text{rel}}(g)$ is defined as follows:

1. if $r \in \text{ID}$, then $\sigma_{\text{rel}}(g) = \langle \{r\}, \emptyset, r, \emptyset, \emptyset \rangle$;
2. otherwise, $\sigma_{\text{rel}}(g) = \langle N \cup \{\hat{r}\}, E \cup \{\hat{r} \xrightarrow{\sigma} r\}, \hat{r}, \downarrow, \text{ID} \rangle$.

Definition 3.3.3 Let $g = \langle N, E, r, \downarrow, \text{ID} \rangle$. Then $\nu_{\text{rel}}(g)$ is defined as follows:

$$\nu_{\text{rel}}(g) = \langle N, E \setminus \{r \xrightarrow{\sigma} s \mid s \in N\}, r, \downarrow, \text{ID} \rangle.$$

Definition 3.3.4 Let $g = \langle N_g, E_g, r_g, \downarrow_g, \text{ID}_g \rangle$ and $h = \langle N_h, E_h, r_h, \downarrow_h, \text{ID}_h \rangle$. Then $g \cdot h$ is defined as follows:

$$\begin{aligned}
g \cdot h &= \langle \\
&\quad (N_g \setminus \downarrow_g) \cup (\downarrow_g \times N_h) \\
&\quad , \quad E_g[f \in \downarrow_g \mapsto (f, r_h)] \cup \downarrow_g \otimes E_h \\
&\quad , \quad r_g \\
&\quad , \quad \downarrow_g \times \downarrow_h \\
&\quad , \quad \text{ID}_g \cup (\downarrow_g \times \text{ID}_h) \\
&\quad \rangle.
\end{aligned}$$

Before we can define $+$ on the graph model, we first have to define a root unwinding map.

Definition 3.3.5 The mapping ρ is for all $g = \langle N, E, r, \downarrow, \text{ID} \rangle$, defined as follows ($u \in A_{\tau\sigma}$):

$$\rho(g) = \langle N \cup \{\hat{r}\}, E \cup \{\hat{r} \xrightarrow{u} s \mid r \xrightarrow{u} s \in E\}, \hat{r}, \downarrow, \text{ID} \cup \{\hat{r} \mid r \in \text{ID}\} \rangle.$$

Definition 3.3.6 Let $g = \langle N_g, E_g, r_g, \downarrow_g, \text{ID}_g \rangle$ and $h = \langle N_h, E_h, r_h, \downarrow_h, \text{ID}_h \rangle$. Then $g + h$ is defined as follows:

1. if $r_g \in \text{ID}_g$, then $g + h = h$;
2. if $r_h \in \text{ID}_h$, then $g + h = g$;
3. if $r_g \notin \text{ID}_g$ and $r_h \notin \text{ID}_h$, then

(a) if $r_g \xrightarrow{\sigma} r_g \in E_g$ and $r_h \xrightarrow{\sigma} r_h \in E_h$, then

$$g + h = \langle \begin{array}{l} (N_g \setminus \{r_g\}) \cup (N_h \setminus \{r_h\}) \cup \{r_{g+h}\} \\ , \quad E_g[r_g \mapsto r_{g+h}] \cup E_h[r_h \mapsto r_{g+h}] \\ , \quad r_{g+h} \\ , \quad \downarrow_g \cup \downarrow_h \\ , \quad \text{ID}_g \cup \text{ID}_h \end{array} \rangle;$$

(b) if $r_g \xrightarrow{\sigma} s_g \in E_g$ and $r_h \xrightarrow{\sigma} s_h \in E_h$ for some $s_g \in N_g$ and $s_h \in N_h$ such that $s_g \neq r_g$ or $s_h \neq r_h$, then suppose that $\rho(g) = \langle N'_g, E'_g, r'_g, \downarrow'_g, \text{ID}'_g \rangle$ and $\rho(h) = \langle N'_h, E'_h, r'_h, \downarrow'_h, \text{ID}'_h \rangle$. Suppose that $r'_g \xrightarrow{\sigma} s'_g$ and $r'_h \xrightarrow{\sigma} s'_h$. Suppose that $(g)_{s'_g} + (h)_{s'_h} = \langle N, E, r, \downarrow, \text{ID} \rangle$. Then,

$$g + h = \langle \begin{array}{l} N'_g \cup N'_h \cup N \cup \{r'_{g+h}\} \\ , \quad E'_g \setminus \{r'_g \xrightarrow{\sigma} s'_g\} \cup E'_h \setminus \{r'_h \xrightarrow{\sigma} s'_h\} \cup E \cup \{r'_{g+h} \xrightarrow{\sigma} r\} \\ , \quad r'_{g+h} \\ , \quad \downarrow'_g \cup \downarrow'_h \cup \downarrow \\ , \quad \text{ID}'_g \cup \text{ID}'_h \cup \text{ID} \end{array} \rangle;$$

(c) otherwise, suppose that $\rho(g) = \langle N'_g, E'_g, r'_g, \downarrow'_g, \text{ID}'_g \rangle$ and $\rho(h) = \langle N'_h, E'_h, r'_h, \downarrow'_h, \text{ID}'_h \rangle$. Then,

$$g + h = \langle \begin{array}{l} (N'_g \setminus \{r'_g\}) \cup (N'_h \setminus \{r'_h\}) \cup \{r'_{g+h}\} \\ , \quad E'_g[r'_g \mapsto r'_{g+h}] \cup E'_h[r'_h \mapsto r'_{g+h}] \\ , \quad r'_{g+h} \\ , \quad \downarrow'_g \cup \downarrow'_h \\ , \quad \text{ID}'_g \cup \text{ID}'_h \end{array} \rangle.$$

Definition 3.3.7 Let $g = \langle N, E, r, \downarrow, \text{ID} \rangle$. Then σ_{rel}^* is defined inductively as follows:

1. if $r \xrightarrow{\sigma} r$, then $\sigma_{\text{rel}}^*(g) = g$;
2. if $r \xrightarrow{\sigma} s$ for some $s \neq r$, then define $\nu = \langle N, E \setminus \{r \xrightarrow{\sigma} s\}, r, \downarrow, \text{ID} \rangle$ and $\psi = (g)_s$. Suppose $\sigma_{\text{rel}}^*(\nu + \psi) = \langle N', E', r', \downarrow', \text{ID}' \rangle$. Then,

$$\sigma_{\text{rel}}^*(g) = \langle N \cup N', E \setminus \{r \xrightarrow{\sigma} s\} \cup E' \cup \{r \xrightarrow{\sigma} r'\}, r, \downarrow \cup \downarrow', \text{ID} \cup \text{ID}' \rangle;$$

3. otherwise, if $r \in \text{ID}$, then $\sigma_{\text{rel}}^*(g) = \langle \{r\}, \{r \xrightarrow{\sigma} r\}, r, \emptyset, \emptyset \rangle$, else $\sigma_{\text{rel}}^*(g) = \langle N, E \cup \{r \xrightarrow{\sigma} r\}, r, \downarrow, \text{ID} \rangle$.

This is a well-founded inductive definition since the length of the path from r consisting of σ -steps only (not σ -loops) decreases.

The mapping *graph* which associates to a closed $\text{BPA}_{\text{drt}}\tau$ term a process graph is defined inductively as can be expected by using the previously given operators on process graphs. Applying the mapping *graph* to an arbitrary closed $\text{BPA}_{\text{drt}}\tau$ term always gives a graph without cycles other than σ -loops. Such graphs will be called root unwound process graphs on several occasions.

Definition 3.3.8 The mapping $term : \mathbf{G}^+ \rightarrow T(\text{BPA}_{\text{drt}}\tau)$ is for all process graphs $g = \langle N, E, r, \downarrow, \text{ID} \rangle$ defined inductively by:

$$term(g) = \begin{cases} \sigma_{\text{rel}}^*(term(\langle N, E \setminus \{r \xrightarrow{\sigma} r\}, r, \downarrow, \text{ID} \rangle)) & \text{if } r \xrightarrow{\sigma} r \in E; \\ \delta & \text{if } r \xrightarrow{\sigma} r \notin E \wedge r \in \text{ID}; \\ cts(\delta) & \text{if } r \xrightarrow{\sigma} r \notin E \wedge r \notin \text{ID} \wedge E = \emptyset; \\ \sum_{r \xrightarrow{u} s \in E} term_g(r \xrightarrow{u} s) & \text{otherwise,} \end{cases}$$

where $term_g : E \rightarrow T(\text{BPA}_{\text{drt}}\tau)$ is, for $a \in A_\tau$, defined by

$$\begin{aligned} term_g(r \xrightarrow{a} s) &= \begin{cases} cts(a) & \text{if } s \in \downarrow; \\ cts(a) \cdot term((g)_s) & \text{otherwise;} \end{cases} \\ term_g(r \xrightarrow{\sigma} s) &= \sigma_{\text{rel}}(term((g)_s)). \end{aligned}$$

Theorem 3.3.9 (Isomorphy) $T(\text{BPA}_{\text{drt}}\tau)/\cong$ and \mathbf{G}/\cong are isomorphic.

3.4 Alternative characterization of the bisimulations

We proceed to find formulations for our bisimulation relations in terms of colourings, analogous to [15]. Let \mathbf{C} be a given set, the set of colours. A *coloured graph* is a process graph with colours $C \in \mathbf{C}$ as labels attached at the nodes.

A *concrete coloured trace* of a coloured graph g is a finite sequence $(C_0, a_1, C_1, \dots, C_{k-1}, a_k, C_k)$ for which there exists a path $r_0 \xrightarrow{a_1} r_1 \xrightarrow{a_2} \dots \xrightarrow{a_k} r_k$ from the root node r_0 of g such that r_i has colour C_i .

A *concrete consistent colouring* of a set of process graphs is a colouring of their nodes with the property that two nodes (also from the same graph) have the same colour only if they have the same concrete trace set and the same label. Two process graphs g and h are *concrete coloured trace equivalent*, notation $g \equiv_{cc} h$, if, for some concrete consistent colouring on $\{g, h\}$, they have the same concrete coloured trace set, or equivalently, the root nodes have the same colour. As in [15], we have the following theorem.

Theorem 3.4.1 For process graphs $g, h \in \mathbf{G}$, we have $g \cong h$ iff $g \equiv_{cc} h$.

An *abstract coloured trace* of a coloured graph is a sequence of the form $(C_0, a_1, C_1, a_2, \dots, a_k, C_k)$ which is obtained from a concrete coloured trace of this graph by replacing all subsequences of the form $(C, \tau, C, \tau, \dots, \tau, C)$ by C .

An *abstract consistent colouring* of a set of graphs is a colouring of their nodes with the property that two nodes have the same colour only if they have the same abstract coloured trace set and the same label. For two coloured graphs g and h we write $g \equiv_c h$, if, for some abstract consistent colouring on $\{g, h\}$, they have the same abstract coloured trace set. In the sequel, we leave out the adjectives ‘abstract’.

Theorem 3.4.2 For process graphs $g, h \in \mathbf{G}$, we have $g \cong_{bt} h$ iff $g \equiv_c h$.

A *rooted coloured trace* of a coloured graph is a sequence of the form $(C_0, a_1, C_1, a_2, \dots, a_k, C_k)$ which is obtained from a concrete coloured trace of this graph by replacing all non- σ -reachable subsequences $(C, \tau, C, \tau, \dots, \tau, C)$ by C .

A *rooted consistent colouring* of a set of graphs is a colouring of their nodes with the property that two nodes have the same colour only if they have the same rooted coloured trace set and the same label. For two root unwound coloured graphs g and h we write $g \equiv_{rc} h$, if, for some rooted consistent colouring on $\{g, h\}$, they have the same rooted coloured trace set.

Theorem 3.4.3 For process graphs $g, h \in \mathbf{G}$, we have $g \cong_{rbt} h$ iff $g \equiv_{rc} h$.

It is possible to colour the nodes of a root unwound process graph g in such a way that two nodes have the same colour iff they can be related by a rooted branching tail autobisimulation on g . This colouring is rooted and consistent. The largest rooted branching tail autobisimulation relation of a graph is called its *canonical colouring*.

Definition 3.4.4 Let g be a root unwound process graph and consider its canonical colouring with colour set \mathbf{C} . Let $N(g)$, the *normal form* of g , be the graph which can be found by contracting all nodes with the same colour and removing τ -loops. More precisely:

1. $N(g)$ has colours $C \in \mathbf{C}$ as its nodes;
2. $N(g)$ has an edge $C \xrightarrow{a} C'$ ($a \in A_\sigma$) iff g has an edge $r \xrightarrow{a} r'$ such that r has colour C and r' has colour C' ;
3. $N(g)$ has an edge $C \xrightarrow{\tau} C'$ iff $C \neq C'$ and g has an edge $r \xrightarrow{\tau} r'$ and r has colour C and r' has colour C' ;
4. C has a label \downarrow iff g has a node r with label \downarrow and r has colour C ;
5. C has a label ID iff g has a node r with label ID and r has colour C .

For all root unwound process graphs $g \in \mathbf{G}$, we have $g \Leftrightarrow_{\text{rbt}} N(g)$.

Definition 3.4.5 (Graph Isomorphism) A *graph isomorphism* is a bijective relation R between the nodes of two process graphs $g = \langle N_g, E_g, r_g, \downarrow_g, \text{ID}_g \rangle$ and $h = \langle N_h, E_h, r_h, \downarrow_h, \text{ID}_h \rangle$ such that:

1. the roots of g and h are related by R , i.e., $R(r_g, r_h)$;
2. if $R(r, s)$ and $R(r', s')$ then $r \xrightarrow{a} r' \in E_g$ iff $s \xrightarrow{a} s' \in E_h$;
3. if $R(r, s)$ then $r \in \downarrow_g$ iff $s \in \downarrow_h$ and $r \in \text{ID}_g$ iff $s \in \text{ID}_h$.

Two graphs are *isomorphic*, notation $g \simeq h$, iff there exists an isomorphism between them. Note that \simeq is a congruence relation on process graphs. Note also that only the nodes which are reachable from the root nodes of g and h are taken into account. As a consequence we have that a process graph g is graph isomorphic with the graph consisting of the edges and nodes of g that are reachable from the root node of g .

As in [15] we have the following Normal Form Theorem.

Theorem 3.4.6 (Normal form theorem) *Let g and h be root unwound process graphs that are in normal form. Then $g \Leftrightarrow_{\text{rbt}} h$ if and only if $g \simeq h$.*

4 Completeness of the Axiomatization

4.1 Introduction

In this section we will prove, following the approach of [15], that $\text{BPA}_{\text{drt}}\tau$ is a complete axiomatization of rooted branching tail bisimulation on process graphs from \mathbf{G}^+ . The basic idea in the completeness proof is to establish a graph rewriting system on process graphs, which is confluent and strongly normalizing (up to graph isomorphism), and for which every rewrite step preserves rooted branching tail bisimilarity. Confluence and strong normalization together are needed to ensure that for every process graph there is a unique normal form up to graph isomorphism. We want the rewrite steps of the graph rewriting system to preserve rooted branching tail bisimilarity since for every such rewrite step we can prove (rather easily) that the corresponding terms are derivably equal (see Theorem 4.4.3). Furthermore, we will show that the normal forms with respect to the graph rewriting system are normal forms in the sense of Section 3.4. As a consequence we have that the normal forms with respect to the graph rewriting system of two rooted branching tail bisimilar process graphs are isomorphic. All that remains is to show that for every term p , the mapping *term* \circ *graph* is an identity modulo derivability. This is the sole subject of Section 4.3.

4.2 The Graph Rewriting System

The graph rewriting system will be such that it is capable of performing two transformations on process graphs. Firstly, it can contract two nodes which are essentially the same. Such nodes are called *double nodes*. Secondly, it can contract a τ -edge that is redundant. Such τ -edges are called *manifestly inert*.

Definition 4.2.1 (Double Nodes) A pair (r, s) , with $r \neq s$, of nodes in a process graph $g = \langle N, E, r, \downarrow, \text{ID} \rangle$ is called a pair of *double nodes* if

- for all nodes $t \neq r, s$ and labels $u \in A_{\tau\sigma}$: $r \xrightarrow{u} t \in E$ iff $s \xrightarrow{u} t \in E$;
- $r \xrightarrow{\sigma} r \in E$ iff $s \xrightarrow{\sigma} s \in E$;
- $r \in \downarrow$ iff $s \in \downarrow$;
- $r \in \text{ID}$ iff $s \in \text{ID}$.

The following proposition formalizes the statement that double nodes are essentially the same: their sub-graphs are isomorphic.

Proposition 4.2.2 *Let g be a process graph. If (r, s) is a pair of double nodes in g , then $(g)_r \simeq (g)_s$.*

A τ -edge between two nodes r and s is called manifestly inert if r and s agree with respect to the labels and σ -loops and if any other outgoing edge of r is also an outgoing edge of s .

Definition 4.2.3 (Manifestly inert τ -edges) An edge $r \xrightarrow{\tau} s$ in a process graph g is *manifestly inert* if r is not σ -reachable from the root node of g , r and s have the same label (if any), and for all nodes t and labels $a \in A_{\tau\sigma}$ such that $(a, t) \neq (\tau, s)$ and $(a, t) \neq (\sigma, r)$: $r \xrightarrow{a} t$ implies $s \xrightarrow{a} t$, and $r \xrightarrow{\sigma} r$ iff $s \xrightarrow{\sigma} s$.

Basically, for an edge to be manifestly inert means that it can be removed with one of the axioms for silent step. The next theorem states that in order to obtain a normal form for a process graph we only have to repeatedly unify a pair of double nodes or contract a manifestly inert τ -edge.

Theorem 4.2.4 *A process graph $g \in \mathbf{G}$ without double nodes and without manifestly inert τ -edges is in normal form.*

Proof. Let g be a finite process graph which is not in normal form. Then it has at least one pair of different nodes with the same colour with respect to the canonical colouring. The *depth* of a node s in g is defined to be the number of edges in its longest path not counting σ -loops. Then, we define the *combined depth* of two nodes as the sum of their depths. We mention the following property of the depth of a node: if $r \xrightarrow{u} s$ ($r \neq s$), then $d(s) < d(r)$. We will use it in the remainder of the proof.

Choose a pair (r, s) of different, but equally coloured nodes in g with minimal combined depth. Now we have the following trivial claim: if r' and s' are nodes in g which have the same colour and moreover $d(r') + d(s') < d(r) + d(s)$, then $r' = s'$ (*). If this would not be the case then clearly (r, s) is not a minimal pair.

Assume, without loss of generality, $d(s) \leq d(r)$. Now, we prove the following two properties:

1. if $r \xrightarrow{u} t$ and $(u, t) \neq (\tau, s)$, then $s \xrightarrow{u} t$, and
2. if $s \xrightarrow{u} t$, then $r \xrightarrow{\tau} s$ or $r \xrightarrow{u} t$.

Ad 1. Suppose that $r \xrightarrow{u} t$ and that $(u, t) \neq (\tau, s)$. Since r and s have the same colour we find that either (1) $u = \tau$ and t has the same colour as r and s , or (2) s has the coloured trace $(C(r), u, C(t))$. For the first case we find $t = s$ from (*) and $d(t) < d(r)$. This is in contradiction with the assumption $(u, t) \neq (\tau, s)$. For the second case we reason as follows. Suppose that $s \xrightarrow{\tau} p$ with $C(p) = C(s)$. Then from $d(p) < d(s)$ we have $p = r$. Then we also have the following contradiction: $d(p) < d(s) \leq d(r)$. So clearly, $u \neq \tau$. Suppose that $s \xrightarrow{u} p$ for some node p such that $C(p) = C(t)$. Since $d(p) < d(s)$ and $d(t) < d(r)$, we have $d(p) + d(t) < d(s) + d(r)$. We also have $C(p) = C(t)$ and $C(s) = C(r)$. But then we have by (*) that $t = p$. So clearly $s \xrightarrow{u} t$ is an edge in g .

Ad 2. Suppose that $s \xrightarrow{u} t$. Then we have $d(t) < d(s)$. But then also $d(t) + d(r) < d(s) + d(r)$. If s and t have the same colour, then by (*) we have $r = t$. This is in contradiction with $d(t) < d(s) \leq d(r)$. So clearly s and t do not have the same colour. Then $(C(s), u, C(t))$ is a coloured trace of r , since s and r have the same colour. Now two cases can be distinguished. First, suppose that $r \xrightarrow{\tau} p$ for some node p such that $C(p) = C(r)$. Then $d(p) < d(r)$ and also $d(p) + d(s) < d(r) + d(s)$. Hence, from (*) we have $p = s$. So we have $r \xrightarrow{\tau} s$. Second, suppose that $r \xrightarrow{u} p$ ($u \neq \tau$) for some node p with $C(p) = C(t)$.

Since $d(p) < d(r)$ and $d(t) < d(s)$, we have $d(p) + d(t) < d(r) + d(s)$. From (*) we obtain $p = t$. So $r \xrightarrow{u} t$. Concluding, either $r \xrightarrow{\tau} s$ or $r \xrightarrow{u} t$.

From the two properties we just proved it follows that if $r \xrightarrow{\tau} s$ is an edge in g , then it is manifestly inert, and if $r \xrightarrow{\tau} s$ is not an edge in g , then (r, s) is a pair of double nodes.

The previous theorem can serve as a source of inspiration for defining the graph rewriting system.

Definition 4.2.5 The rewriting relation \rightarrow is defined by the following one-step reductions:

1. sharing a pair of double nodes (r, s) : replace all edges $t \xrightarrow{u} r$ by $t \xrightarrow{u} s$ (if not already there, otherwise just remove $t \xrightarrow{u} r$) and remove r with all its outgoing edges from the graph.
2. contracting a manifestly inert τ -edge $r \xrightarrow{\tau} s$: replace all edges $t \xrightarrow{a} r$ by $t \xrightarrow{a} s$ (if not already there, otherwise just remove $t \xrightarrow{a} r$) and remove r with all its outgoing edges from the graph.

Theorem 4.2.6 *The graph rewrite relation \rightarrow has the following properties:*

1. if $g \in \mathbf{G}^+$ and $g \rightarrow h$, then $h \in \mathbf{G}^+$;
2. \rightarrow preserves rooted branching tail bisimilarity, i.e., if $g \rightarrow h$, then $g \Leftrightarrow_{\text{rbt}} h$;
3. the graph rewrite system is strongly normalizing;
4. the graph rewrite system is confluent (up to graph isomorphism).

Proof. We will omit the proof for the first property. For the second property, we will show that rooted branching tail bisimilarity is preserved under the application of each of the two rewrite rules. (1) Sharing a pair of double nodes. Suppose that (r, s) is a pair of double nodes in g . Then \rightarrow identifies the nodes r and s , i.e., it removes the node r . Then the relation $R = Id(h) \cup \{(r, s), (s, r)\}$ is clearly a rooted branching tail bisimulation between g and h . (2) Contracting a manifestly inert τ -edge. Suppose that $r \xrightarrow{\tau} s$ is a manifestly inert τ -edge. The relation $R = Id(h) \cup \{(r, s), (s, r)\}$ is a rooted branching tail bisimulation between g and h .

For the third property we reason as follows. Strong normalization of the graph rewrite system follows immediately from the following observations: (1) every finite process graph has only finitely many nodes, and (2) in every application of a rewrite rule the number of nodes decreases with one.

For the proof of the fourth property, suppose that g has two normal forms g_1 and g_2 . Then both g_1 and g_2 are without double nodes and without manifestly inert τ -edges by Theorem 4.2.4. From the second item of this theorem and $g \rightarrow^* g_1$, $g \rightarrow^* g_2$, we have $g \Leftrightarrow_{\text{rbt}} g_1$ and $g \Leftrightarrow_{\text{rbt}} g_2$ and since rooted branching tail bisimilarity is an equivalence relation also $g_1 \Leftrightarrow_{\text{rbt}} g_2$. From the Normal Form Theorem (Theorem 3.4.6) we obtain $g_1 \simeq g_2$.

4.3 Correspondence between \mathbf{G}^+ and $\text{BPA}_{\text{drt}}\tau$

As already announced we need to show that the mapping $\text{term} \circ \text{graph}$ is an identity modulo derivability. For basic terms this turns out to be easy. The extension of this result to closed terms in general is based on the observation that for every axiom used in the Elimination Theorem the graphs corresponding to the left-hand side and the right-hand side are isomorphic, and hence the corresponding terms are derivably equal.

Proposition 4.3.1 *Let $g = \langle N, E, r, \downarrow, ID \rangle$ and $s \in N$. Then $s \in ID$ iff $\text{term}((g)_s) = \hat{\delta}$.*

Proposition 4.3.2 *If $g, h \in \mathbf{G}^+$ and $g \simeq h$, then $A1, A2 \vdash \text{term}(g) = \text{term}(h)$.*

Proposition 4.3.3 *For closed $\text{BPA}_{\text{drt}}\tau$ terms p and q , we have*

1. if $p \in B(\text{BPA}_{\text{drt}}\tau)$, then $A1, A2, DRT4, DRT5 \vdash \text{term}(\text{graph}(p)) = p$;
2. if $A1, A2, A4, A5, A6ID, A7ID, DRT1-DRT3, DRTSID, DCS1-DCS4, DCSID, ARTS, DRTI1-DRTI5, DRTIID, DCSTI \vdash p = q$, then $\text{graph}(p) \simeq \text{graph}(q)$.

Theorem 4.3.4 For closed $BPA_{drt}\tau$ -terms p we have

$$A1, A2, A4, A5, A6ID, A7ID, DRT1-DRT3, DRTSID, DCS1-DCS4, DCSID, \vdash \text{term}(\text{graph}(p)) = p. \\ \text{ARTS}, \text{DRTI1-DRTI5}, \text{DRTIID}, \text{DCSTI}$$

Proof. By the Elimination Theorem we have the existence of a basic term q such that

$$A1, A2, A4, A5, A6ID, A7ID, DRT1-DRT3, DRTSID, \vdash p = q. \\ \text{DCS1-DCS4}, \text{DCSID}, \text{ARTS}, \text{DRTI1-DRTI5}, \text{DRTIID}, \text{DCSTI}$$

Then we obtain from Proposition 4.3.3.2 that $\text{graph}(p) \simeq \text{graph}(q)$. Then, by Proposition 4.3.2, we have $A1, A2 \vdash \text{term}(\text{graph}(p)) = \text{term}(\text{graph}(q))$. By Proposition 4.3.3.1 we also have $A1, A2, \text{DRT4}, \text{DRT5} \vdash \text{term}(\text{graph}(q)) = q$. So,

$$A1, A2, A4, A5, A6ID, A7ID, DRT1-DRT3, DRTSID, \vdash \text{term}(\text{graph}(p)) = \text{term}(\text{graph}(q)) = q = p. \\ \text{DCS1-DCS4}, \text{DCSID}, \text{ARTS}, \text{DRTI1-DRTI5}, \text{DRTIID}, \text{DCSTI}$$

4.4 Every rewrite step corresponds to a proof step

Proposition 4.4.1 Let (r, s) be a pair of double nodes in a process graph g . Then we have $A1, A2 \vdash \text{term}((g)_r) = \text{term}((g)_s)$.

Proof. This proposition follows immediately from Proposition 4.2.2 and Proposition 4.3.2.

Proposition 4.4.2 Let $r \xrightarrow{\tau} s$ be a manifestly inert τ -edge in a process graph g . Let x be a closed $BPA_{drt}^- \tau$ -term. Then

1. $BPA_{drt}^- \tau \vdash x \cdot \sigma_{rel}^m(\text{term}((g)_r)) = x \cdot \sigma_{rel}^m(\text{term}((g)_s))$;
2. $BPA_{drt} \tau \vdash x \cdot \sigma_{rel}^m(\sigma_{rel}(\text{term}((g)_r)) + \nu_{rel}(y)) = x \cdot \sigma_{rel}^m(\sigma_{rel}(\text{term}((g)_s)) + \nu_{rel}(y))$.

Proof. We will only give the proof for the second property. The proof for the first property is similar (though easier). Since $r \xrightarrow{\tau} s$, we have $\neg \text{ID}(r)$, and hence $\neg \text{ID}(s)$. Then we have by Proposition 4.3.1 that $\text{term}((g)_s) \neq \delta$ and therefore by Proposition 2.1.7, we have $\text{term}((g)_s) = \text{term}((g)_s) + \text{cts}(\delta)$. Two cases can be distinguished: (1) r does not have a σ -loop, or (2) r does have a σ -loop. We only show the derivation for the second case. Then $\text{term}((g)_r) = \sigma_{rel}^*(\text{cts}(\tau) \cdot \text{term}((g)_s) + q)$ and $\text{term}((g)_s) = \sigma_{rel}^*(p + q)$ for some basic terms p and q such that neither p nor q has a σ -summand. Then

$$\begin{aligned} BPA_{drt} \tau \vdash & x \cdot \sigma_{rel}^m(\sigma_{rel}(\text{term}((g)_r)) + \nu_{rel}(y)) \\ &= x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(\text{cts}(\tau) \cdot \text{term}((g)_s) + q)) + \nu_{rel}(y)) \\ &= x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(\text{cts}(\tau) \cdot \sigma_{rel}^*(p + q) + q)) + \nu_{rel}(y)) \\ &= x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(\text{cts}(\tau) \cdot \sigma_{rel}^*(p + q + \text{cts}(\delta)) + q)) + \nu_{rel}(y)) \\ &= x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(p + q + \text{cts}(\delta))) + \nu_{rel}(y)) \\ &= x \cdot \sigma_{rel}^m(\sigma_{rel}(\sigma_{rel}^*(p + q)) + \nu_{rel}(y)) \\ &= x \cdot \sigma_{rel}^m(\sigma_{rel}(\text{term}((g)_s)) + \nu_{rel}(y)). \end{aligned}$$

Theorem 4.4.3 If $g \rightarrow h$, then $BPA_{drt} \tau \vdash \text{term}(g) = \text{term}(h)$.

Proof. Suppose that $g \rightarrow h$. This must be due to either the sharing of a pair of double nodes, or the contraction of a manifestly inert τ -edge. Each of these two rewritings in turn is built up from more elementary rewritings. These are the removal of an unreachable node, the replacement of an edge by another one, or the removal of an edge. With respect to the removal of unreachable parts of process graphs we have that the original graph and the resulting graph are isomorphic since unreachable nodes are not taken into account. For the sharing of a pair of double nodes (r, s) or the contraction of a manifestly inert τ -edge $r \xrightarrow{\tau} s$ the following cases can be distinguished: (1) r has no incoming edges, (2) r does have an incoming edge $t \xrightarrow{u} r$ and s does not have an incoming edge $t \xrightarrow{u} s$, or (3) r has an incoming edge $t \xrightarrow{u} r$ and s has an incoming edge $t \xrightarrow{u} s$. For double nodes we give the derivations for the second case and for manifestly inert τ -edges we give the derivations for the third case:

- Let (r, s) be a pair of double nodes in g such that r has an incoming edge $t \xrightarrow{u} r$ and such that s does not have an incoming edge $t \xrightarrow{u} s$. In the case that $u \in A_\tau$, we have that for some basic term p : if t does not have a σ -loop, then $\text{term}((g)_t) = \text{cts}(u) \cdot \text{term}((g)_r) + p$ and $\text{term}((h)_t) = \text{cts}(u) \cdot \text{term}((h)_s) + p$, or, if t does have a σ -loop, then $\text{term}((g)_t) = \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_r) + p)$ and $\text{term}((h)_t) = \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((h)_s) + p)$. Furthermore we have by construction that $(g)_s \simeq (h)_s$, and therefore by Proposition 4.3.2, $A1, A2 \vdash \text{term}((g)_s) = \text{term}((h)_s)$. Since (r, s) is a pair of double nodes we have from Proposition 4.4.1 that $A1, A2 \vdash \text{term}((g)_r) = \text{term}((g)_s)$. Combining these gives us, in the case that t does not have a σ -loop:

$$\begin{aligned} \text{BPA}_{\text{drt}}\tau &\vdash \text{term}((g)_t) \\ &= \text{cts}(u) \cdot \text{term}((g)_r) + p \\ &= \text{cts}(u) \cdot \text{term}((g)_s) + p \\ &= \text{cts}(u) \cdot \text{term}((h)_s) + p \\ &= \text{term}((h)_t), \end{aligned}$$

and in the other case:

$$\begin{aligned} \text{BPA}_{\text{drt}}\tau &\vdash \text{term}((g)_t) \\ &= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_r) + p) \\ &= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_s) + p) \\ &= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((h)_s) + p) \\ &= \text{term}((h)_t). \end{aligned}$$

If, on the other hand $u = \sigma$, then we have that for some basic term p : $\text{term}((g)_t) = \sigma_{\text{rel}}(\text{term}((g)_r)) + p$ and $\text{term}((h)_t) = \sigma_{\text{rel}}(\text{term}((h)_s)) + p$. Please note that t cannot have a σ -loop since every node may have at most one outgoing σ -edge. We have, by construction, $(g)_s \simeq (h)_s$, and therefore by Proposition 4.3.2 also $A1, A2 \vdash \text{term}((g)_s) = \text{term}((h)_s)$. Finally, since (r, s) is a pair of double nodes we have by Proposition 4.4.1 that $A1, A2 \vdash \text{term}((g)_r) = \text{term}((g)_s)$. Combining these gives us:

$$\begin{aligned} \text{BPA}_{\text{drt}}\tau &\vdash \text{term}((g)_t) \\ &= \sigma_{\text{rel}}(\text{term}((g)_r)) + p \\ &= \sigma_{\text{rel}}(\text{term}((g)_s)) + p \\ &= \sigma_{\text{rel}}(\text{term}((h)_s)) + p \\ &= \text{term}((h)_t). \end{aligned}$$

- Let $r \xrightarrow{\tau} s$ be a manifestly inert τ -edge in g such that r has an incoming edge $t \xrightarrow{u} r$ and such that s also has an incoming edge $t \xrightarrow{u} s$. Then, since t cannot have two outgoing σ -edges, we have $u \in A_\tau$. Two cases are distinguished: t does not have a σ -loop and t does have a σ -loop. First, suppose that t does not have a σ -loop. Then, we have that for some basic term p : $\text{term}((g)_t) = \text{cts}(u) \cdot \text{term}((g)_r) + \text{cts}(u) \cdot \text{term}((g)_s) + p$ and $\text{term}((h)_t) = \text{cts}(u) \cdot \text{term}((h)_s) + p$. Furthermore, we have by construction that $(g)_s \simeq (h)_s$, and therefore by Proposition 4.3.2 that $A1, A2 \vdash \text{term}((g)_s) = \text{term}((h)_s)$. Since $r \xrightarrow{\tau} s$ is a manifestly inert τ -edge, we have by Proposition 4.4.2 that $\text{BPA}_{\text{drt}}\tau \vdash \text{cts}(u) \cdot \text{term}((g)_r) = \text{cts}(u) \cdot \text{term}((g)_s)$. Combining these gives us:

$$\begin{aligned} \text{BPA}_{\text{drt}}\tau &\vdash \text{term}((g)_t) \\ &= \text{cts}(u) \cdot \text{term}((g)_r) + \text{cts}(u) \cdot \text{term}((g)_s) + p \\ &= \text{cts}(u) \cdot \text{term}((g)_s) + \text{cts}(u) \cdot \text{term}((g)_s) + p \\ &= \text{cts}(u) \cdot \text{term}((g)_s) + p \\ &= \text{cts}(u) \cdot \text{term}((h)_s) + p \\ &= \text{term}((h)_t). \end{aligned}$$

Second, suppose that t does have a σ -loop. Then, we have that for some basic term p : $\text{term}((g)_t) = \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_r) + \text{cts}(u) \cdot \text{term}((g)_s) + p)$ and $\text{term}((h)_t) = \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((h)_s) + p)$. Furthermore, we have by construction that $(g)_s \simeq (h)_s$, and therefore by Proposition 4.3.2 that $A1, A2 \vdash \text{term}((g)_s) = \text{term}((h)_s)$. Since $r \xrightarrow{\tau} s$ is a manifestly inert τ -edge, we have by Proposition 4.4.2 that

$\text{BPA}_{\text{drt}}\tau \vdash \text{cts}(u) \cdot \text{term}((g)_r) = \text{cts}(u) \cdot \text{term}((g)_s)$. Combining these gives us:

$$\begin{aligned}
\text{BPA}_{\text{drt}}\tau &\vdash \text{term}((g)_r) \\
&= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_r) + \text{cts}(u) \cdot \text{term}((g)_s) + p) \\
&= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_s) + \text{cts}(u) \cdot \text{term}((g)_s) + p) \\
&= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((g)_s) + p) \\
&= \sigma_{\text{rel}}^*(\text{cts}(u) \cdot \text{term}((h)_s) + p) \\
&= \text{term}((h)_r).
\end{aligned}$$

Theorem 4.4.4 (Completeness) *$\text{BPA}_{\text{drt}}\tau$ is a complete axiomatization of rooted branching tail bisimilarity on process graphs.*

Proof. Let p and q be arbitrary closed $\text{BPA}_{\text{drt}}\tau$ terms. Suppose that $\text{graph}(p) \Leftrightarrow_{\text{rbt}} \text{graph}(q)$. Then we must prove that $\text{BPA}_{\text{drt}}\tau \vdash p = q$. Let g and h be the unique normal forms with respect to \rightarrow of $\text{graph}(p)$ and $\text{graph}(q)$ respectively. Then, by Theorem 4.2.6, we find $g \Leftrightarrow_{\text{rbt}} \text{graph}(p) \Leftrightarrow_{\text{rbt}} \text{graph}(q) \Leftrightarrow_{\text{rbt}} h$. By Theorem 4.2.4 it follows that g and h must be in normal form in the sense of Section 3.4. Then, by Theorem 3.4.6, we have $g \simeq h$. Thus, by Proposition 4.3.2, we have $\text{BPA}_{\text{drt}}\tau \vdash \text{term}(g) = \text{term}(h)$. Also, by Theorem 4.3.4, we have $\text{BPA}_{\text{drt}}\tau \vdash p = \text{term}(\text{graph}(p))$ and $\text{BPA}_{\text{drt}}\tau \vdash \text{term}(\text{graph}(q)) = q$. By Theorem 4.4.3, we have $\text{BPA}_{\text{drt}}\tau \vdash \text{term}(\text{graph}(p)) = \text{term}(g)$ and $\text{BPA}_{\text{drt}}\tau \vdash \text{term}(h) = \text{term}(\text{graph}(q))$. Combining these gives us $\text{BPA}_{\text{drt}}\tau \vdash p = \text{term}(\text{graph}(p)) = \text{term}(g) = \text{term}(h) = \text{term}(\text{graph}(q)) = q$.

5 Additional Operators

In this section we will consider some extensions of $\text{BPA}_{\text{drt}}\tau$. First a simple notion of time free projection will be introduced. This operator is useful when relating a specification without timing information to a specification with timing information. Also, an operator for parallel composition with communication is defined.

5.1 Time Abstraction

We have embedded the time free theory into the discrete time theory. We call a process *time free* if it can be generated from the constants $\text{ats}(a)$, $\text{ats}(\delta)$, $\text{ats}(\tau)$ and the operators $+$, \cdot . In this section, we define an operator that throws away all timing information, so that the image of a discrete time closed term will always be a time free term. This operator is called *time abstraction* or *time free projection*, and is denoted by π_{tf} (see [13]). We present axioms in Table 6, operational rules in Table 7. All axioms in Table 6 are sound with respect to strong and rooted branching tail bisimulation on the term model. From these axioms one can easily derive $\pi_{\text{tf}}(\text{ats}(a)) = \text{ats}(a)$.

$$\begin{array}{ll}
\pi_{\text{tf}}(\delta) = \text{ats}(\delta) & \pi_{\text{tf}}(x + y) = \pi_{\text{tf}}(x) + \pi_{\text{tf}}(y) \\
\pi_{\text{tf}}(\text{cts}(a)) = \text{ats}(a) & \pi_{\text{tf}}(x \cdot y) = \pi_{\text{tf}}(x) \cdot \pi_{\text{tf}}(y) \\
\pi_{\text{tf}}(\sigma_{\text{rel}}(x)) = \pi_{\text{tf}}(x) & \pi_{\text{tf}}(\sigma_{\text{rel}}^*(x)) = \pi_{\text{tf}}(x)
\end{array}$$

Table 6: Axioms for time abstraction ($a \in A_{\tau\delta}$).

5.2 Merge with Communication

The extension of the theory $\text{BPA}_{\text{drt}}\tau$ with parallel composition, with or without communication, can be done along the lines of [4]. We present the extension with parallel composition with communication. In fact the axioms presented in this paper are almost identical to those presented in [4]. The only differences are that

$\frac{x \xrightarrow{a} x'}{\pi_{\text{tf}}(x) \xrightarrow{a} \pi_{\text{tf}}(x')}$	$\frac{x \xrightarrow{a} \surd}{\pi_{\text{tf}}(x) \xrightarrow{a} \surd}$	$\pi_{\text{tf}}(x) \xrightarrow{\sigma} \pi_{\text{tf}}(x)$
$\frac{x \xrightarrow{\sigma} x', \pi_{\text{tf}}(x') \xrightarrow{a} x''}{\pi_{\text{tf}}(x) \xrightarrow{a} x''}$	$\frac{x \xrightarrow{\sigma} x', \pi_{\text{tf}}(x') \xrightarrow{a} \surd}{\pi_{\text{tf}}(x) \xrightarrow{a} \surd}$	

Table 7: Operational rules for time abstraction ($a \in A_\tau$).

some minor mistakes are corrected (see [24]). The additional syntax has binary operators \parallel (merge), \ll (left merge), and $|$ (communication merge) and unary operators ∂_H (encapsulation, for $H \subseteq A$), and τ_I (abstraction, for $I \subseteq A$). We present axioms for $\text{ACP}_{\text{drt}\tau}$ in Table 8 and Table 9. The operational rules are exactly those given in [4] and are therefore omitted. We assume given a partial, commutative and associative communication function $\gamma : A \times A \rightarrow A$.

$x \parallel y = x \ll y + y \ll x + x y$	$cts(a) cts(b) = cts(\gamma(a, b))$ if defined
$x \ll \delta = \delta$	$cts(a) cts(b) = cts(\delta)$ otherwise
$\delta \ll x = \delta$	$cts(a) cts(b) \cdot x = (cts(a) cts(b)) \cdot x$
$(x + y) \ll z = x \ll z + y \ll z$	$cts(a) \cdot x cts(b) = (cts(a) cts(b)) \cdot x$
$cts(a) \ll (x + cts(\delta)) = cts(a) \cdot (x + cts(\delta))$	$cts(a) \cdot x cts(b) \cdot y = (cts(a) cts(b)) \cdot (x \parallel y)$
$cts(a) \cdot x \ll (y + cts(\delta)) = cts(a) \cdot (x \parallel (y + cts(\delta)))$	$(x + y) z = x z + y z$
$\sigma_{\text{rel}}(x) \ll (\sigma_{\text{rel}}(y) + \nu_{\text{rel}}(z)) = \sigma_{\text{rel}}(x \ll y)$	$x (y + z) = x y + x z$
$x \delta = \delta$	$\sigma_{\text{rel}}(x) (\nu_{\text{rel}}(z) + cts(\delta)) = cts(\delta)$
$\delta x = \delta$	$(\nu_{\text{rel}}(y) + cts(\delta)) \sigma_{\text{rel}}(z) = cts(\delta)$
	$\sigma_{\text{rel}}(x) \sigma_{\text{rel}}(z) = \sigma_{\text{rel}}(x y)$

Table 8: Axioms for parallel composition in $\text{ACP}_{\text{drt}\tau}$ ($a \in A_\tau$).

$\partial_H(\delta) = \delta$	$\tau_I(\delta) = \delta$
$\partial_H(cts(a)) = cts(a)$	$\tau_I(cts(a)) = cts(a)$
$\partial_H(cts(a)) = cts(\delta)$	$\tau_I(cts(a)) = cts(\tau)$
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$
$\partial_H(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\partial_H(x))$	$\tau_I(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\tau_I(x))$

Table 9: Axioms for renaming operators in $\text{ACP}_{\text{drt}\tau}$ ($a \in A_\tau$).

Proposition 5.2.1 *We mention the following identities which are useful in the calculations to come:*

1. $\sigma_{\text{rel}}(x) \parallel \sigma_{\text{rel}}(y) = \sigma_{\text{rel}}(x \parallel y)$;
2. $ats(a) \cdot x \ll [y]^\omega = ats(a) \cdot (x \parallel [y]^\omega)$;
3. $cts(a) \cdot x \ll \sigma_{\text{rel}}(y) = cts(a) \cdot (x \parallel \sigma_{\text{rel}}(y))$;
4. $cts(a) \cdot x \parallel \sigma_{\text{rel}}(y) = cts(a) \cdot (x \parallel \sigma_{\text{rel}}(y))$;
5. $ats(a) \cdot x \ll cts(b) \cdot y = cts(a) \cdot (x \parallel cts(b) \cdot y)$;

6. $ats(a) \mid ats(b) = ats(c)$ if $\gamma(a, b) = c$, and $ats(a) \mid ats(b) = ats(\delta)$ otherwise;
7. $ats(a) \cdot x \mid ats(b) \cdot y = (ats(a) \mid ats(b)) \cdot (x \parallel y)$;
8. $ats(a) \cdot x \mid cts(b) \cdot y = (cts(a) \mid cts(b)) \cdot (x \parallel y)$;
9. $\partial_H(ats(a)) = ats(a)$ if $a \notin H$, and $\partial_H(ats(a)) = ats(\delta)$ if $a \in H$;
10. $\tau_I(ats(a)) = ats(a)$ if $a \notin I$, and $\tau_I(ats(a)) = ats(\tau)$ if $a \in I$.

Proposition 5.2.2 *The following identity, useful in verifications, can be proved for all closed terms x, y :*

$$cts(a) \cdot (cts(\tau) \cdot (x + cts(\delta)) \parallel (y + cts(\delta))) = cts(a) \cdot ((x + cts(\delta)) \parallel (y + cts(\delta))).$$

5.3 Some Simple Calculations: Communicating Buffers

In this section, we give some simple calculations in order to illustrate the use of our discrete time theory. In the setting with unbounded start delay instead of time iteration these calculations can be found in [5]. We keep formulas compact by writing \underline{a} instead of $cts(a)$ and a instead of $ats(a)$ (this is in line with notation used in [2, 13]).

The communication format follows the so-called *standard communication function*. Suppose we have given two finite sets, the set of messages or data D , and the set of ports P . For each $d \in D$ and $i \in P$, we have atomic actions $r_i(d)$, $s_i(d)$, $c_i(d)$ (denoting *receive*, *send* and *communicate d along i*) and the only defined communications are $\gamma(r_i(d), s_i(d)) = \gamma(s_i(d), r_i(d)) = c_i(d)$. In time free process algebra, there is the following standard specification of a one-item buffer with input port i and output port j :

$$B^{ij} = \sum_{d \in D} r_i(d) \cdot s_j(d) \cdot B^{ij}$$

A straightforward calculation (see e.g. [7], page 106) shows that the composition of two such buffers in sequence gives a two-item buffer. In the following, we consider three timed versions of this buffer. In each case, only one input per time slice is possible.

We can define a channel that allows one input in every time slice, and outputs with no delay, with input port i and output port j by the following recursive equation:

$$C^{ij} = \sum_{d \in D} r_i(d) \cdot \underline{s_j(d)} \cdot \sigma_{\text{rel}}(C^{ij})$$

We see $C^{ij} = \sigma_{\text{rel}}^*(\nu_{\text{rel}}(C^{ij})) = \lfloor C^{ij} \rfloor^\omega$. With $H = \{r_2(d), s_2(d) \mid d \in D\}$ we can derive:

$$\begin{aligned} & \partial_H(C^{12} \parallel C^{23}) \\ = & \partial_H(C^{12} \parallel C^{23}) + \partial_H(C^{23} \parallel C^{12}) + \partial_H(C^{12} \mid C^{23}) \\ = & \sum_{d \in D} \partial_H(r_1(d) \cdot \underline{s_2(d)} \cdot \sigma_{\text{rel}}(C^{12}) \parallel \lfloor C^{23} \rfloor^\omega) \\ & + \sum_{d \in D} \partial_H(r_2(d) \cdot \underline{s_3(d)} \cdot \sigma_{\text{rel}}(C^{23}) \parallel \lfloor C^{12} \rfloor^\omega) \\ & + \sum_{d, e \in D} \partial_H(r_1(d) \cdot \underline{s_2(d)} \cdot \sigma_{\text{rel}}(C^{12}) \mid r_2(e) \cdot \underline{s_3(e)} \cdot \sigma_{\text{rel}}(C^{23})) \\ = & \sum_{d \in D} r_1(d) \cdot \partial_H(\underline{s_2(d)} \cdot \sigma_{\text{rel}}(C^{12}) \parallel C^{23}) + \sum_{d \in D} \delta \cdot \partial_H(\underline{s_3(d)} \cdot \sigma_{\text{rel}}(C^{23}) \parallel C^{12}) \\ & + \sum_{d, e \in D} \delta \cdot \partial_H(\underline{s_2(d)} \cdot \sigma_{\text{rel}}(C^{12}) \parallel \underline{s_3(e)} \cdot \sigma_{\text{rel}}(C^{23})) \\ = & \sum_{d \in D} r_1(d) \cdot (\partial_H(\underline{s_2(d)} \cdot \sigma_{\text{rel}}(C^{12}) \parallel C^{23}) + \partial_H(C^{23} \parallel \underline{s_2(d)} \cdot \sigma_{\text{rel}}(C^{12}))) \end{aligned}$$

$$\begin{aligned}
& + \partial_H \left(\underline{\underline{s_2(d)}} \cdot \sigma_{\text{rel}}(C^{12}) \mid C^{23} \right) + \delta \\
= & \sum_{d \in D} r_1(d) \cdot \left(\underline{\underline{\delta}} + \sum_{e \in D} \partial_H \left(r_2(e) \cdot \underline{\underline{s_3(e)}} \cdot \sigma_{\text{rel}}(C^{23}) \right) \parallel \underline{\underline{s_2(d)}} \cdot \sigma_{\text{rel}}(C^{12}) \right. \\
& \left. + \sum_{e \in D} \partial_H \left(\underline{\underline{s_2(d)}} \cdot \sigma_{\text{rel}}(C^{12}) \mid r_2(e) \cdot \underline{\underline{s_3(e)}} \cdot \sigma_{\text{rel}}(C^{23}) \right) \right) \\
= & \sum_{d \in D} r_1(d) \cdot \left(\underline{\underline{\delta}} + \underline{\underline{\delta}} + \underline{\underline{c_2(d)}} \cdot \partial_H \left(\sigma_{\text{rel}}(C^{12}) \parallel \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}}(C^{23}) \right) \right) \\
= & \sum_{d \in D} r_1(d) \cdot \underline{\underline{c_2(d)}} \cdot \underline{\underline{s_3(d)}} \cdot \partial_H \left(\sigma_{\text{rel}}(C^{12}) \parallel \sigma_{\text{rel}}(C^{23}) \right) \\
= & \sum_{d \in D} r_1(d) \cdot \underline{\underline{c_2(d)}} \cdot \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}} \left(\partial_H(C^{12} \parallel C^{23}) \right).
\end{aligned}$$

Next, we put $I = \{c_2(d) \mid d \in D\}$ and obtain the following recursive equation:

$$\tau_I \circ \partial_H(C^{12} \parallel C^{23}) = \sum_{d \in D} r_1(d) \cdot \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}}(\tau_I \circ \partial_H(C^{12} \parallel C^{23})).$$

We see that the composition behaves again as a no-delay channel, with input port 1 and output port 3.

It is interesting to see what happens if we change the previous specification slightly. Consider

$$D^{ij} = \sum_{d \in D} r_i(d) \cdot \sigma_{\text{rel}} \left(\underline{\underline{s_j(d)}} \cdot D^{ij} \right).$$

This specification describes a buffer with capacity one and delay between input and output of one time unit. The composition $\partial_H(D^{12} \parallel D^{23})$ (with H as above) satisfies the following recursive specification:

$$\begin{aligned}
X &= \sum_{d \in D} r_1(d) \cdot \sigma_{\text{rel}} \left(\underline{\underline{c_2(d)}} \right) \cdot X_d, \\
X_d &= \sum_{e \in D} r_1(e) \cdot \sigma_{\text{rel}} \left(\underline{\underline{s_3(d)}} \right) \cdot \underline{\underline{c_2(e)}} \cdot X_e \\
&+ \sigma_{\text{rel}} \left(\underline{\underline{s_3(d)}} \cdot X + \sum_{e \in D} r_1(e) \cdot \underline{\underline{s_3(d)}} \cdot \underline{\underline{c_2(e)}} \cdot \sigma_{\text{rel}} \left(\underline{\underline{c_2(e)}} \right) \cdot X_e \right) \quad (\text{for } d \in D).
\end{aligned}$$

Hiding the internal communications, we obtain

$$\begin{aligned}
\tau_I(X) &= \sum_{d \in D} r_1(d) \cdot \sigma_{\text{rel}}(\tau_I(X_d)), \\
\tau_I(X_d) &= \sum_{e \in D} r_1(e) \cdot \sigma_{\text{rel}} \left(\underline{\underline{s_3(d)}} \right) \cdot \tau_I(X_e) \\
&+ \sigma_{\text{rel}} \left(\underline{\underline{s_3(d)}} \cdot \tau_I(X) + \sum_{e \in D} r_1(e) \cdot \underline{\underline{s_3(d)}} \cdot \underline{\underline{c_2(e)}} \cdot \sigma_{\text{rel}}(\tau_I(X_e)) \right).
\end{aligned}$$

The composition denotes a buffer with capacity two and delay of two time units. Now, we apply the time abstraction operator, and obtain the following specification:

$$\begin{aligned}
\pi_{\text{tf}} \circ \tau_I(X) &= \sum_{d \in D} r_1(d) \cdot \pi_{\text{tf}} \circ \tau_I(X_d), \\
\pi_{\text{tf}} \circ \tau_I(X_d) &= s_3(d) \cdot \pi_{\text{tf}} \circ \tau_I(X) + \sum_{e \in D} r_1(e) \cdot s_3(d) \cdot \pi_{\text{tf}} \circ \tau_I(X_e).
\end{aligned}$$

This is exactly the result of the time free calculation on page 106 of [7] (after abstraction). We find that the following remarkable equation holds for the buffers D^{ij} , but not for the other variants of buffers we discuss:

$$\partial_H(\pi_{\text{tf}}(D^{12}) \parallel \pi_{\text{tf}}(D^{23})) = \pi_{\text{tf}} \circ \partial_H(D^{12} \parallel D^{23}).$$

Finally, we drop the restriction in the previous specification that output must occur before the next input. We obtain the following specification:

$$E^{ij} = \sum_{d \in D} r_i(d) \cdot \sigma_{\text{rel}} \left(\underline{\underline{s_j(d)}} \parallel E^{ij} \right).$$

Now, the composition satisfies the following recursive specification:

$$\begin{aligned} Y^0 &= \sum_{d \in D} r_1(d) \cdot \sigma_{\text{rel}}(Y_d^1), \\ Y_d^1 &= \underline{\underline{c_2(d)}} \cdot Y_d^2 + \sum_{e \in D} r_1(e) \cdot \underline{\underline{c_2(d)}} \cdot \sigma_{\text{rel}}(Y_{de}^3) \quad (\text{for } d \in D), \\ Y_d^2 &= \sigma_{\text{rel}} \left(\underline{\underline{s_3(d)}} \cdot Y^0 + \sum_{e \in D} r_1(e) \cdot \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}}(Y_e^1) \right) + \sum_{e \in D} r_1(e) \cdot \sigma_{\text{rel}}(Y_{de}^3) \quad (\text{for } d \in D), \\ Y_{de}^3 &= \underline{\underline{s_3(d)}} \cdot Y_e^1 + \underline{\underline{c_2(e)}} \cdot \left(\underline{\underline{s_3(d)}} \cdot Y_d^2 + \sum_{f \in D} r_1(f) \cdot \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}}(Y_{ef}^3) \right) \\ &\quad + \sum_{f \in D} r_1(f) \cdot \left(\underline{\underline{c_2(e)}} \parallel \underline{\underline{s_3(d)}} \right) \cdot \sigma_{\text{rel}}(Y_{ef}^3) \quad (\text{for } d, e \in D). \end{aligned}$$

After abstraction, we obtain the following specification for $Z^0 = \tau_I(Y^0)$:

$$\begin{aligned} Z^0 &= \sum_{d \in D} r_1(d) \cdot \sigma_{\text{rel}}(Z_d^1), \\ Z_d^1 &= \sigma_{\text{rel}} \left(\underline{\underline{s_3(d)}} \cdot Z^0 + \sum_{e \in D} r_1(e) \cdot \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}}(Z_e^1) \right) + \sum_{e \in D} r_1(e) \cdot \sigma_{\text{rel}}(Z_{de}^2) \quad (\text{for } d \in D), \\ Z_{de}^2 &= \underline{\underline{s_3(d)}} \cdot Z_e^1 + \sum_{f \in D} r_1(f) \cdot \underline{\underline{s_3(d)}} \cdot \sigma_{\text{rel}}(Z_{ef}^2) \quad (\text{for } d, e \in D). \end{aligned}$$

Again, the delay between input and output is two time units, but now, it is possible that three data elements are present in the system at the same time.

6 Absolute and Parametric Timing

6.1 Discrete Time Process Algebra with Absolute Timing

We present a version of the theory in the previous section using absolute timing, where all timing is related to a global clock.

We start with constants $fts(a)$, denoting a in the first time slice ($a \in A_{\delta\tau}$), followed by immediate termination. Besides, we have operators $+$, \cdot as before. In addition, we have the *absolute discrete time unit delay* σ_{abs} , that increments all timing in its scope. Thus, $\sigma_{\text{abs}}(fts(a))$ denotes a in the second time slice. In the axioms we use the absolute value operator $|\cdot|$. This operator turns out to be the identity for all processes using absolute timing only: it initializes a process in the first time slice. Note that in the term $\sigma_{\text{abs}}(fts(a)) \cdot fts(\delta)$, after execution of the a in slice 2, an immediate deadlock will occur. This term is different from $\sigma_{\text{abs}}(fts(a) \cdot fts(\delta))$, where after the execution of a in slice 2, further activity in this slice can take place (of a parallel process). We conclude that in the absolute time theory, the immediate deadlock constant δ is necessary. The axiomatization of BPA_{dat} adds the axioms of Table 10 to the axioms A1-5, A6ID, A7ID (see Table 1).

The extension with delayable actions is similar to the relative time case. Here, we see a large advantage of the time iteration operator over the unbounded start delay operator.

Again, we can find $\text{BPA}_{\delta}^{\tau}$ as an SRM specification by using the interpretation of a as $atstau(a)$. The extension with parallel composition can be found along the same lines as for the relative time case (see [6]). The principle RSP(DAT) can be used to prove identities for σ_{abs}^* :

$$y = x + \sigma_{\text{abs}}(y) \quad \Longrightarrow \quad y = \sigma_{\text{abs}}^*(x) \quad \text{RSP(DAT)}.$$

$\sigma_{\text{abs}}(x) + \sigma_{\text{abs}}(y) = \sigma_{\text{abs}}(x + y)$	$\nu_{\text{abs}}(\hat{\delta}) = \hat{\delta}$
$\sigma_{\text{abs}}(x) \cdot \hat{\delta} = \sigma_{\text{abs}}(x \cdot \hat{\delta})$	$\nu_{\text{abs}}(fts(a)) = fts(a)$
$\sigma_{\text{abs}}(x) \cdot (\nu_{\text{abs}}(y) + z) = \sigma_{\text{abs}}(x) \cdot z$	$\nu_{\text{abs}}(x + y) = \nu_{\text{abs}}(x) + \nu_{\text{abs}}(y)$
$\sigma_{\text{abs}}(x) \cdot \sigma_{\text{abs}}(y) = \sigma_{\text{abs}}(x \cdot y)$	$\nu_{\text{abs}}(x \cdot y) = \nu_{\text{abs}}(x) \cdot y$
$fts(\delta) \cdot x = fts(\delta)$	$\nu_{\text{abs}}(\sigma_{\text{abs}}(x)) = fts(\delta)$
$\sigma_{\text{abs}}(\hat{\delta}) = fts(\delta)$	$ \hat{\delta} = \hat{\delta}$
$fts(a) + fts(\delta) = fts(a)$	$ fts(a) = fts(a)$
$\sigma_{\text{abs}}(x) + fts(\delta) = \sigma_{\text{abs}}(x)$	$ x + y = x + y $
$\sigma_{\text{abs}}^*(x) = x + \sigma_{\text{abs}}(\sigma_{\text{abs}}^*(x))$	$ x \cdot y = x \cdot y$
$ats(a) = \sigma_{\text{abs}}^*(fts(a))$	$ \sigma_{\text{abs}}(x) = \sigma_{\text{abs}}(x)$

Table 10: Axioms of BPA_{dat} ($a \in A_\tau$).

The operational rules are more complicated in this case, as we have to keep track of which time slice we are in, we have to keep track of the global clock. The pair $\langle x, n \rangle$ denotes x in the $(n + 1)$ st time slice. The operational rules for the absolute value operator are trivial. For the operational rules for the constants $fts(a)$ ($a \in A_\tau$) and $\hat{\delta}$ and the operators $+$, \cdot , σ_{abs} we refer to [6]. Operational rules for ν_{abs} , σ_{abs}^* , $ats(a)$ ($a \in A_\tau$) are presented in Table 11.

$\frac{\langle x, 0 \rangle \xrightarrow{a} \langle x', 0 \rangle}{\langle \nu_{\text{abs}}(x), 0 \rangle \xrightarrow{a} \langle x', 0 \rangle}$	$\frac{\langle x, 0 \rangle \xrightarrow{a} \langle \surd, 0 \rangle}{\langle \nu_{\text{abs}}(x), 0 \rangle \xrightarrow{a} \langle \surd, 0 \rangle}$	$\frac{\text{ID}(\langle x, 0 \rangle)}{\text{ID}(\langle \nu_{\text{abs}}(x), 0 \rangle)}$	
$\text{ID}(\langle \nu_{\text{abs}}(x), n+1 \rangle)$	$\langle ats(a), n \rangle \xrightarrow{a} \langle \surd, n \rangle$	$\langle ats(a), n \rangle \xrightarrow{\sigma} \langle ats(a), n+1 \rangle$	$\langle ats(\delta), n \rangle \xrightarrow{\sigma} \langle ats(\delta), n+1 \rangle$
$\frac{\langle x, n \rangle \xrightarrow{a} \langle x', n' \rangle}{\langle \sigma_{\text{abs}}^*(x), n+k \rangle \xrightarrow{a} \langle x', n+k \rangle}$	$\frac{\langle x, n \rangle \xrightarrow{a} \langle \surd, n' \rangle}{\langle \sigma_{\text{abs}}^*(x), n+k \rangle \xrightarrow{a} \langle \surd, n+k \rangle}$	$\langle \sigma_{\text{abs}}^*(x), n \rangle \xrightarrow{\sigma} \langle \sigma_{\text{abs}}^*(x), n+1 \rangle$	

Table 11: Operational rules for BPA_{dat} ($a \in A_\tau$).

We also have to adapt the definition of bisimulation. A *strong bisimulation relation* is a symmetric binary relation R on $P \times IN$ such that ($u \in A_{\tau\sigma}$, $a \in A_\tau$):

1. if $R(\langle s, n \rangle, \langle s', n' \rangle)$, then $n = n'$ and if $\text{ID}(\langle s, n \rangle)$, then $\text{ID}(\langle s', n' \rangle)$;
2. if $R(\langle s, n \rangle, \langle t, n \rangle)$ and $\langle s, n \rangle \xrightarrow{u} \langle s', n' \rangle$, then there is a term t' such that $\langle t, n \rangle \xrightarrow{u} \langle t', n' \rangle$ and $R(\langle s', n' \rangle, \langle t', n' \rangle)$;
3. if $R(\langle s, n \rangle, \langle t, n \rangle)$ and $\langle s, n \rangle \xrightarrow{a} \langle \surd, n \rangle$, then $\langle t, n \rangle \xrightarrow{a} \langle \surd, n \rangle$.

We say process expressions x and y are *strongly (tail) bisimilar*, denoted $x \Leftrightarrow y$, if there exists a strong bisimulation relation with $R(\langle x, 0 \rangle, \langle y, 0 \rangle)$. A *branching tail bisimulation relation* is a binary relation R on $P \times IN$ such that ($u \in A_{\tau\sigma}$, $a \in A_\tau$):

1. if $R(\langle s, n \rangle, \langle s', n' \rangle)$, then $n = n'$ and if $\text{ID}(\langle s, n \rangle)$, then $\text{ID}(\langle s', n' \rangle)$;
2. if $R(\langle s, n \rangle, \langle t, n \rangle)$ and $\langle s, n \rangle \xrightarrow{u} \langle s', n' \rangle$, then there are terms t^* , t' such that $\langle t, n \rangle \xRightarrow{(u)} \langle t^*, n \rangle \xrightarrow{u} \langle t', n' \rangle$ and $R(\langle s, n \rangle, \langle t^*, n \rangle)$, $R(\langle s', n' \rangle, \langle t', n' \rangle)$;
3. if $R(\langle s, n \rangle, \langle t, n \rangle)$ and $\langle s, n \rangle \xrightarrow{a} \langle \surd, n \rangle$, then there is a term t' such that $\langle t, n \rangle \xRightarrow{(a)} \langle t', n \rangle \xrightarrow{a} \langle \surd, n \rangle$ and $R(\langle s, n \rangle, \langle t', n \rangle)$.

We say process expressions x and y are *branching tail bisimilar*, denoted $x \approx_{\text{bt}} y$, if there exists a branching tail bisimulation relation with $R(\langle x, 0 \rangle, \langle y, 0 \rangle)$. Process expressions x and y are *rooted branching tail bisimilar*, denoted $x \approx_{\text{rbt}} y$, if there exists a branching tail bisimulation relation with $R(\langle x, 0 \rangle, \langle y, 0 \rangle)$, that is strong for all pairs that can be reached from $\langle x, 0 \rangle, \langle y, 0 \rangle$ by just performing time steps.

Axioms for the silent step are comparable to the ones for relative time. The model of closed process expressions modulo rooted branching tail bisimilarity satisfies the axioms in Table 12.

$$\begin{aligned}
x \cdot (fts(\tau) \cdot (v_{\text{abs}}(y) + z + fts(\delta)) + v_{\text{abs}}(y)) &= x \cdot (v_{\text{abs}}(y) + z + fts(\delta)) \\
x \cdot (fts(\tau) \cdot (v_{\text{abs}}(y) + z + fts(\delta)) + z) &= x \cdot (v_{\text{abs}}(y) + z + fts(\delta)) \\
x \cdot (\sigma_{\text{abs}}(fts(\tau) \cdot (y + fts(\delta))) + v_{\text{abs}}(z)) &= x \cdot (\sigma_{\text{abs}}(y + fts(\delta)) + v_{\text{abs}}(z)) \\
x \cdot \sigma_{\text{abs}}^*(fts(\tau) \cdot \sigma_{\text{abs}}^*(v_{\text{abs}}(y) + v_{\text{abs}}(z) + fts(\delta)) + v_{\text{abs}}(y)) &= x \cdot \sigma_{\text{abs}}^*(v_{\text{abs}}(y) + v_{\text{abs}}(z) + fts(\delta))
\end{aligned}$$

Table 12: Axioms for $\text{BPA}_{\text{dt}}\tau$.

6.2 Parametric Time

In this section we integrate the absolute time and the relative time approach. All axioms presented in the previous sections are still valid for all parametric time processes. We obtain a finite axiomatization, that allows an elimination theorem. As a consequence, we can expand expressions like $cts(a) \parallel fts(b)$, $cts(a) \parallel (fts(b) + ats(\delta))$.

We follow [6], where we introduced the operators \odot , the (relative) *time spectrum combinator*, and μ , the *spectrum tail operator*. The absolute value operator can also be called the spectrum head operator. $P \odot Q$ is a process that, when initialized in the first time slice, behaves as $|P|$; when initialized in slice $n + 1$ its behaviour is determined by Q as follows: initialize in slice n thereafter apply σ_{abs} . The process $\mu(X)$ computes a process such that $X = |X| \odot \mu(X)$. For a parametric discrete time process we have the time spectrum sequence $|X|, |\mu(X)|, |\mu^2(X)|, \dots$. For each infinite sequence $(P_n)_{n \in \mathbb{N}}$ one may imagine a process P with $|\mu^n(P)| = P_n$ though not all such P can be finitely expressed.

The theory BPA_{dpt} unites the theories BPA_{drt} and BPA_{dat} together with the additional axioms in Table 13. We extend with delayable actions, to BPA_{dpt} , as indicated in [6]. We can define:

$ cts(a) = fts(a)$	$\mu(\delta) = \delta$
$ \sigma_{\text{rel}}(x) = \sigma_{\text{abs}}(\mu(x))$	$\mu(cts(a)) = cts(a)$
$ v_{\text{rel}}(x) = v_{\text{abs}}(x)$	$\mu(x + y) = \mu(x) + \mu(y)$
$\sigma_{\text{abs}}(x) = \sigma_{\text{abs}}(x)$	$\mu(x \cdot y) = \mu(x) \cdot \mu(y)$
$v_{\text{abs}}(x) = v_{\text{abs}}(x)$	$\mu(\sigma_{\text{rel}}(x)) = \sigma_{\text{rel}}(\mu(x))$
	$\mu(v_{\text{rel}}(x)) = v_{\text{rel}}(\mu(x))$
	$\mu(fts(a)) = \delta$
$ x \odot y = x $	$\mu(\sigma_{\text{abs}}(x)) = x $
$\mu(x \odot y) = y$	$\mu(v_{\text{abs}}(x)) = \delta$
$x = x \odot \mu(x)$	$\sigma_{\text{abs}}(x) \cdot y = \sigma_{\text{abs}}(x \cdot \mu(y))$

Table 13: Additional axioms of BPA_{dpt} ($a \in A_\tau$).

- x is an absolute time process iff $\text{BPA}_{\text{dpt}} \vdash x = |x|$;
- x is an relative time process iff $\text{BPA}_{\text{dpt}} \vdash x = \mu(x)$.

Each BPA_{dpt} process expression can be written in the form

$$X = |X| \odot |\mu(X)| \odot \dots \odot |\mu^n(X)| \odot \mu^{n+1}(X).$$

One can reduce each $|\mu^n(X)|$ to a BPA_{dat} -term and if n is sufficiently large, we can write $\mu^{n+1}(X)$ without any σ_{abs} or $\text{fts}(a)$, so it will be in the relative time signature. We call $(|X|, |\mu(X)|, |\mu^2(X)|, \dots)$ the time spectrum expansion sequence (TSS) of X . Note that we obtain the following spectrum expansion for the ν_{rel} operator:

$$\nu_{\text{rel}}(x) = \nu_{\text{abs}}(x) \odot \nu_{\text{rel}}(\mu(x)).$$

For further details, we refer to [6]. Now the axioms introduced for silent steps in relative and absolute time are valid in parametric time as well.

7 Concluding Remarks

We presented axioms for discrete time process algebra with silent step in branching bisimulation semantics. A small difference with previously published work is that we use time iteration instead of unbounded start delay and that the conditional axiom for silent step is replaced by an unconditional one. For the first version, relative discrete time process algebra, we have given soundness and completeness results with respect to the term model and the graph model (exploiting the isomorphy of those two structures). From this completeness result we can conclude that the principle RSP(DRT) that is used for deriving equalities concerning time iteration can be replaced by axioms.

We have given several embeddings of the time free theories BPA_δ and BPA_δ^τ in the discrete time theories BPA_{drt} and $\text{BPA}_{\text{drt}}^\tau$ respectively.

The extension of the relative discrete time theories with additional operators such as time abstraction, merge with communication, encapsulation and abstraction follows along the same lines as in the time free theory. Some calculations regarding communicating buffers illustrate the use of the relative discrete time theory.

Finally, an outline is given for defining absolute and parametric time versions of discrete time process algebra.

An interesting topic for future research is obtaining soundness and completeness results for the absolute and parametric time process algebras as presented in this paper. The authors expect that, at least for the absolute time version, a similar approach should give these results. With respect to parametric time process algebra a lot of work has to be done. We have reasons to believe that the theory is sound, but a major open question is whether the theory is complete. Also issues relating to the applicability of the theory have to be investigated.

References

- [1] J. C. M. Baeten and J. A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3(2):142–188, 1991.
- [2] J. C. M. Baeten and J. A. Bergstra. Discrete time process algebra (extended abstract). In W.R. Cleaveland, editor, *CONCUR'92, Third International Conference on Concurrency Theory*, volume 630 of *Lecture Notes in Computer Science*, pages 401–420, Stony Brook, 1992. Springer-Verlag.
- [3] J. C. M. Baeten and J. A. Bergstra. On sequential composition, action prefixes and process prefix. *Formal Aspects of Computing*, 6(3):250–268, 1994.
- [4] J. C. M. Baeten and J. A. Bergstra. Discrete time process algebra with abstraction. In H. Reichel, editor, *FCT'95, International Conference on Fundamentals of Computation Theory*, volume 965 of *Lecture Notes in Computer Science*, pages 1–15, Dresden, 1995. Springer-Verlag.
- [5] J. C. M. Baeten and J. A. Bergstra. Some simple calculations in relative discrete time process algebra. In E. H. L. Aarts, H. M. M. ten Eikelder, C. Hemerik, and M. Rem, editors, *Simplex Sigillum Veri*, pages 67–74. Eindhoven University of Technology, 1995. Liber Amicorum dedicated to prof.dr. F. E. J. Kruseman Aretz.

- [6] J. C. M. Baeten and J. A. Bergstra. Discrete time process algebra. *Formal Aspects of Computing*, 8(2):188–208, 1996.
- [7] J. C. M. Baeten and W. P. Weijland. *Process Algebra*, volume 18 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1990.
- [8] T. Basten. Branching bisimilarity is an equivalence indeed! *Information Processing Letters*, 58(1):141–147, 1996.
- [9] J. A. Bergstra and P. Klint. The discrete time Toolbus. In M. Wirsing and M. Nivat, editors, *Algebraic Methodology and Software Technology (AMAST'96)*, volume 1101 of *Lecture Notes in Computer Science*, pages 286–305, Munich, 1996. Springer-Verlag.
- [10] J. A. Bergstra and J. W. Klop. Process algebra for synchronous communication. *Information and Control*, 60(1/3):109–137, 1984.
- [11] J. A. Bergstra and J. W. Klop. Algebra of communicating processes with abstraction. *Theoretical Computer Science*, 37(1):77–121, 1985.
- [12] J. A. Bergstra and C. A. Middelburg. A process algebra semantics of ϕ SDL. Technical Report LGPS 129, Utrecht University, Department of Philosophy, 1995.
- [13] S. H. J. Bos and M. A. Reniers. The $\hat{P}C$ -bus in discrete-time process algebra. *Science of Computer Programming*, 1996. To appear.
- [14] S. D. Brookes, C. A. R. Hoare, and A. W. Roscoe. A theory of communicating sequential processes. *Journal of the ACM*, 31(3):560–599, 1984.
- [15] R. J. van Glabbeek and W.P. Weijland. Branching time and abstraction in bisimulation semantics. *Journal of the ACM*, 43(3):555–600, 1996.
- [16] J. F. Groote. Specification and verification of real time systems in ACP. In L. Logrippo, R. L. Probert, and H. Ural, editors, *Protocol Specification, Testing and Verification*, volume 10 of *Proc. IFIP WG 6.1 Tenth International Symposium*, pages 261–274, Ottawa, 1990. North-Holland.
- [17] M. Hennessy and T. Regan. A temporal process algebra. Technical Report 2/90, University of Sussex, 1990.
- [18] A. S. Klusener. *Models and Axioms for a Fragment of Real Time Process Algebra*. PhD thesis, Eindhoven University of Technology, 1993.
- [19] R. Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, 1980.
- [20] R. Milner. *Communication and Concurrency*. International Series in Computer Science. Prentice Hall International, 1989.
- [21] F. Moller and C. Tofts. A temporal calculus of communicating systems. In J. C. M. Baeten and J. W. Klop, editors, *CONCUR'90 – Theories of Concurrency: Unification and Extension*, volume 458 of *Lecture Notes in Computer Science*, pages 401–415, Amsterdam, 1990. Springer-Verlag.
- [22] X. Nicollin and J. Sifakis. The algebra of timed processes, ATP: Theory and application. *Information and Computation*, 114(1):131–178, 1994.
- [23] G. M. Reed and A. W. Roscoe. A timed model for communicating sequential processes. *Theoretical Computer Science*, 58:249–261, 1988.
- [24] M. A. Reniers and J. J. Vereijken. Completeness in discrete-time process algebra. Technical Report CSR 96/15, Eindhoven University of Technology, Department of Computing Science, 1996.
- [25] C. Verhoef. A congruence theorem for structured operational semantics with predicates and negative premises. *Nordic Journal of Computing*, 2(2):274–302, 1995.