Dormancy-aware timed branching bisimilarity for timed analysis of communication protocols

Middelburg, C.A.

Link to publication

Dormancy-Aware Timed Branching Bisimilarity
With an Application to Communication Protocol Analysis

C.A. Middelburg

ORCID: https://orcid.org/0000-0002-8725-0197

Informatics Institute, Faculty of Science, University of Amsterdam
Science Park 904, 1098 XH Amsterdam, the Netherlands
C.A.Middelburg@uva.nl

Abstract. A variant of the standard notion of branching bisimilarity for processes with discrete relative timing is proposed which is coarser than the standard notion. Using a version of ACP (Algebra of Communicating Processes) with abstraction for processes with discrete relative timing, it is shown that the proposed variant allows of both the functional correctness and the performance properties of the PAR (Positive Acknowledgement with Retransmission) protocol to be analyzed. In the version of ACP concerned, the difference between the standard notion of branching bisimilarity and its proposed variant is characterized by a single axiom schema.

Keywords: process algebra, discrete relative timing, branching bisimilarity, PAR protocol, functional correctness, performance property.

1998 ACM Computing Classification: C.2.2, D.1.3, D.2.4, F.1.2, F.3.1.

1 Introduction

The axiom systems of versions of ACP (Algebra of Communicating Processes) with abstraction for processes with discrete relative timing are based on the notion of branching bisimilarity from [16] adapted to processes with discrete relative timing (see e.g. [2,3,4]). The experience with analyzing the PAR (Positive Acknowledgement with Retransmission) protocol [26, Section 3.3] during the writing of [4] triggered a quest for a variant of that notion of branching bisimilarity that is coarser. In this paper, such a variant is proposed.

The PAR protocol is a communication protocol which is based on time-outs of positive acknowledgements. Timing is essential for the correct behaviour of the PAR protocol: the protocol behaves only correctly if the time-out time of the sender, i.e. the time after which it retransmits a datum for which it is still awaiting an acknowledgement, is longer than the time that a complete protocol cycle takes. There have been attempts to describe this protocol using process algebra without timing. In those attempts, see e.g. [27], it is excluded that the sender times out too early by inhibiting a time-out so long as it does not lead to deadlock. This boils down to assuming a sender that is somehow able to determine, when it is waiting for an acknowledgement, whether the receiver and

the channels used for communication are at rest — this is nicely illustrated in [9]. Such attempts resort to assumptions of which it is unlikely that they can be fulfilled by actual communication protocols.

In [4], a version of ACP with abstraction for processes with discrete relative timing is used to describe the PAR protocol and to show that the protocol behaves correctly if the time-out time of the sender is longer than the time that a complete protocol cycle takes. In order to achieve this result, it is necessary to take the timing of actions into account in all calculations that must be done before abstraction from the actions that should be regarded as internal can be applied. From the point where abstraction from those actions can be applied, the timing of actions is no longer relevant. Actually, many internal actions can only be removed after abstraction from the timing of actions.

It would facilitate analysis of performance properties if most internal actions could be removed without preceding abstraction from the timing of actions. The axioms used for the removal of internal actions are based on the notion of branching bisimilarity for processes with discrete relative timing introduced in [2,3,4]. What is needed in the case of performance analysis, is a coarser equivalence relation. This is the main motivation for the proposal in this paper of a coarser equivalence relation, with a plausible rationale, which still coincides with the original version of branching bisimilarity from [16] in the case without timing.

In [5], a first attempt has been made to define this coarser version. The attempt was unsatisfactory. The definition in that paper was in fact complicated to such an extent that it led to errors in the paper that were not spotted at the time. Like the definition of the standard version, the definition concerned refers to a two-phase operational semantics of a version of ACP with abstraction for processes with discrete relative timing. In this paper, a definition will be given that refers to a time-stamped operational semantics instead. It turns out that taking a time-stamped operational semantics as the basis results in a definition that is far less complicated than the one from [5].

To give a first indication of the usefulness of the proposed equivalence, it is also shown in this paper that in the case of the PAR protocol all internal actions that hinder performance analysis can be removed without preceding abstraction from the timing of actions if we use the axioms based on the proposed equivalence. Because of the plausible rationale behind this equivalence that is given in this paper, it is likely that the usefulness of the proposed equivalence is not restricted to the PAR protocol.

The contributions of this paper are not only the definition of a variant of the standard notion of branching bisimilarity for processes with discrete relative timing and the devising of an axiom schema characterizing the difference between the standard notion and the variant, but also (a) the definition of both a two-phase structural operational semantics and a time-stamped structural operational semantics of a version of ACP with abstraction for processes with discrete relative timing, (b) two equivalent definitions of the standard notion of branching bisimilarity for processes with discrete relative timing, one referring to the two-phase semantics and one referring to the time-stamped semantics,

and (c) the extension of the version of ACP being considered with unrestricted nested guarded recursion. The version of ACP being considered is arguably the core of the versions presented earlier in [2,3,4]. To the best of my knowledge, time-stamped structural operational semantics of process algebras for processes with relative timing and process algebras with unrestricted nested guarded recursion have not been considered in the computer science literature before.

The structure of this paper is as follows. First, the version of ACP with abstraction for processes with discrete relative timing used in this paper is presented (Section 2). Next, this version is used to formally specify the PAR protocol and to analyze it (Section 3). Then, the standard notion of branching bisimilarity for processes with discrete relative timing is defined in two ways (Section 4). Following this, the variant of the standard notion referred to above is introduced and an axiom schema that characterizes the difference between the two notions is given (Section 5). After that, the analysis is revisited using that axiom schema (Section 6). Finally, some concluding remarks are made (Section 7).

In [4], a coherent collection of four process algebras with timing, each dealing with timing in a different way, is presented. The time scale on which the time is measured is either discrete or continuous, and the timing of actions is either relative or absolute. In the current paper, a minor variant of the process algebra with discrete relative timing from that collection is used. Various constants and operators of that process algebra have counterparts in the other process algebras from the collection. In [4], a notational distinction is made between a constant or operator of one process algebra and its counterparts in other process algebras, by means of different decorations of a common symbol, if they should not be identified in case process algebras are compared. In this paper, the decorations are omitted because they can safely be omitted so long as a single process algebra is used.

2 Process Algebraic Preliminaries

The version of ACP used in this paper can be considered the core of $\mathrm{ACP}_\tau^{\mathrm{drt}}$, the process algebra with abstraction for processes with discrete relative timing from [4], extended with nested guarded recursion. To distinguish it from $\mathrm{ACP}_\tau^{\mathrm{drt}}$, the core of $\mathrm{ACP}_\tau^{\mathrm{drt}}$ is denoted by $\mathrm{ACP}_{\mathrm{drt}}^\tau$ in this paper. In this section, $\mathrm{ACP}_{\mathrm{drt}}^\tau$ and its extension with nested guarded recursion are introduced. It focuses on the signatures and axiom systems of these algebraic theories. The congruence with respect to which closed substitution instances of the equations and conditional equations from the axiom systems are valid is introduced in Section 4.

2.1 $\mathrm{ACP}_{\mathrm{drt}}^\tau$

In the case of $\mathrm{ACP}_{\mathrm{drt}}^\tau$, timing is relative to the time at which the preceding action is performed and time is measured on a discrete time scale. Measuring time on a discrete time scale means that time is divided into time slices and timing of actions is done with respect to the time slices in which they are performed.

Roughly speaking, $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ is ACP^{τ} extended with two operators to deal with timing: one-time-slice delay and current-time-slice time-out. The former operator is a basic one and the latter operator is a useful auxiliary one.

In $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$, it is assumed that a fixed but arbitrary finite set A of *basic actions*, with $\tau, \delta \notin \mathsf{A}$, and a fixed but arbitrary commutative and associative *communication* function $\gamma : (\mathsf{A} \cup \{\tau, \delta\}) \times (\mathsf{A} \cup \{\tau, \delta\}) \to (\mathsf{A} \cup \{\tau, \delta\})$, such that $\gamma(\tau, a) = \delta$ and $\gamma(\delta, a) = \delta$ for all $a \in \mathsf{A} \cup \{\tau, \delta\}$, have been given. Basic actions are taken as atomic processes. The function γ is regarded to give the result of synchronously performing any two basic actions for which this is possible, and to be δ otherwise. Henceforth, we write A_{τ} for $\mathsf{A} \cup \{\tau\}$ and $\mathsf{A}_{\tau\delta}$ for $\mathsf{A} \cup \{\tau, \delta\}$.

The signature of the algebraic theory $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ consists of the following constants and operators:

- for each $a \in \mathsf{A}$, the *undelayable action* constant $\underline{a}$;
- the *undelayable silent step* constant $\underline{\tau}$;
- the *undelayable deadlock* constant $\underline{\delta}$;
- the binary *alternative composition* operator $+$;
- the binary *sequential composition* operator $\cdot$;
- the unary *one-time-slice delay* operator σ;
- the binary *parallel composition* operator $\parallel$;
- the binary *left merge* operator $\lfloor\!\lfloor$;
- the binary *communication merge* operator $\mid$;
- for each $H \subseteq \mathsf{A}$, the unary *encapsulation* operator ∂_H;
- for each $I \subseteq \mathsf{A}$, the unary *abstraction* operator τ_I;
- the unary *current-time-slice time-out* operator ν.

It is assumed that there is a countably infinite set $\mathcal{X}$ of variables, which contains x, y and z. Terms over the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ are built as usual. Infix notation is used for the binary operators. The need to use parentheses is reduced by using the associativity of the operators $+$, $\cdot$ and $\parallel$ and the following precedence conventions: the operator $\cdot$ binds stronger than all other binary operators and the operator $+$ binds weaker than all other binary operators. Henceforth, terms over the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ are also called $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ terms. This terminology will also be used for terms over the signature of restrictions and extensions of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$.

The constants of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ can be explained as follows ($a \in \mathsf{A}$):

- $\underline{a}$ denotes the process that performs the observable action a in the current time slice and after that immediately terminates successfully;
- $\underline{\tau}$ denotes the process that performs an unobservable action in the current time slice and after that immediately terminates successfully;
- $\underline{\delta}$ denotes the process that is neither capable of performing any action in the current time slice nor capable of idling till the next time slice.

Let t and t' be closed $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ terms denoting processes p and p', and let $H, I \subseteq \mathsf{A}$. Then the operators of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ can be explained as follows:

- $t + t'$ denotes the process that behaves either as p or as p', where the choice between the two is resolved at the instant that one of them performs its first action, and not before;
- $t \cdot t'$ denotes the process that first behaves as p and following successful termination of p behaves as p';
- $\sigma(t)$ denotes the process that idles till the next time slice and then behaves as p;
- $t \parallel t'$ denotes the process that behaves as p and p' in parallel, by which is meant that, each time an action is performed, either a next action of p is performed or a next action of p' is performed or a next action of p and a next action of p' are performed synchronously;
- $t \parallel\!\!\!\!\perp t'$ denotes the same process as $t \parallel t'$, except that it starts with performing an action of p;
- $t \mid t'$ denotes the same process as $t \parallel t'$, except that it starts with performing an action of p and an action of p' synchronously;
- $\partial_H(t)$ denotes the process that behaves the same as p, except that it keeps p from performing actions in H;
- $\tau_I(t)$ denotes the process that behaves the same as p, except that actions in I are turned into unobservable actions;
- $\nu(t)$, denotes the process that behaves the same as p, except that it keeps p from idling till the next time slice.

The axiom system of $\mathrm{ACP}^\tau_{\mathrm{drt}}$ consists of the equations given in Table 1. In this table, a, b, and c stand for arbitrary members of $\mathsf{A}_{\tau\delta}$ and H and I stand for arbitrary subsets of A.

The axiom names in Table 1 have been chosen such that it is clear whether an axiom is an axiom from the untimed case without any adaptation, an axiom from the untimed case with a minor adaptation (with suffix DR), or a completely new axiom (with prefix DR).

$\mathrm{ACP}^\tau_{\mathrm{drt}}$ is the core of the process algebra $\mathrm{ACP}^{\mathrm{drt}}_\tau$ presented in [4]. In $\mathrm{ACP}^\tau_{\mathrm{drt}}$, the *deadlocked process* constant $\mathring{\delta}$ and the binary *relative initialization* operator $\overline{v}_{\mathsf{rel}}$ from $\mathrm{ACP}^{\mathrm{drt}}_\tau$ have been omitted, the binary *relative delay* operator σ_{rel} has been replaced by the unary one-time-slice delay operator σ, and the binary *relative time-out* operator v_{rel} has been replaced by the unary current-time-slice time-out operator ν. The omitted constant and operator and the replaced operators have counterparts that are basic in the case of absolute timing, parametric timing or continuous time scale. However, the omitted constant and operator are not basic in the case of relative timing and discrete time scale and the replaced operators can be defined in terms of their replacements in the case of relative timing and discrete time scale. In [4], what is omitted or replaced in $\mathrm{ACP}^\tau_{\mathrm{drt}}$ is relevant to the coherence of the different process algebras presented there.

Two subtheories of $\mathrm{ACP}^\tau_{\mathrm{drt}}$ that will be referred to later are $\mathrm{BPA}^\tau_{\mathrm{drt}}$ and $\mathrm{ACP}_{\mathrm{drt}}$. $\mathrm{BPA}^\tau_{\mathrm{drt}}$ is the restriction of $\mathrm{ACP}^\tau_{\mathrm{drt}}$ obtained by omitting the operators $\parallel$, $\parallel\!\!\!\!\perp$, $\mid$, and ∂_H (for $H \subseteq \mathsf{A}$) and the axioms in which these operators occur. $\mathrm{ACP}_{\mathrm{drt}}$ is the restriction of $\mathrm{ACP}^\tau_{\mathrm{drt}}$ obtained by omitting the constant $\underline{\tau}$, the operators τ_I (for $I \subseteq \mathsf{A}$), and the axioms in which this constant or these operators occur.

5

Table 1. Axiom system of $\text{ACP}^\tau_{\text{drt}}$

$x + y = y + x$	A1	$x \parallel y = (x \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} y + y \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} x) + x \mid y$	CM1
$(x + y) + z = x + (y + z)$	A2	$\underline{a} \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} x = \underline{a} \cdot x$	CM2DR
$x + x = x$	A3	$\underline{a} \cdot x \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} y = \underline{a} \cdot (x \parallel y)$	CM3DR
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$\sigma(x) \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} \nu(y) = \underline{\delta}$	DRCM1
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$\sigma(x) \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} (\nu(y) + \sigma(z)) = \sigma(x \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} z)$	DRCM2
$x + \underline{\delta} = x$	A6DR	$(x + y) \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} z = x \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} z + y \mathbin{\rule[0.4ex]{1.2ex}{0.4pt}\hspace{-1.2ex}\parallel} z$	CM4
$\underline{\delta} \cdot x = \underline{\delta}$	A7DR	$\underline{a} \cdot x \mid \underline{b} = (\underline{a} \mid \underline{b}) \cdot x$	CM5DR
		$\underline{a} \mid \underline{b} \cdot x = (\underline{a} \mid \underline{b}) \cdot x$	CM6DR
$\sigma(x) + \sigma(y) = \sigma(x + y)$	DRT1	$\underline{a} \cdot x \mid \underline{b} \cdot y = (\underline{a} \mid \underline{b}) \cdot (x \parallel y)$	CM7DR
$\sigma(x) \cdot y = \sigma(x \cdot y)$	DRT2	$\nu(x) \mid \sigma(y) = \underline{\delta}$	DRCM3
		$\sigma(x) \mid \nu(y) = \underline{\delta}$	DRCM4
$\partial_H(\underline{a}) = \underline{a}$ if $a \notin H$	D1DR	$\sigma(x) \mid \sigma(y) = \sigma(x \mid y)$	DRCM5
$\partial_H(\underline{a}) = \underline{\delta}$ if $a \in H$	D2DR	$(x + y) \mid z = x \mid z + y \mid z$	CM8
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3	$x \mid (y + z) = x \mid y + x \mid z$	CM9
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4		
$\partial_H(\sigma(x)) = \sigma(\partial_H(x))$	DRD	$\underline{a} \mid \underline{b} = \underline{c}$ if $\gamma(a,b) = c$	CFDR
$\tau_I(\underline{a}) = \underline{a}$ if $a \notin I$	TI1DR	$\nu(\underline{a}) = \underline{a}$	DRTO1
$\tau_I(\underline{a}) = \underline{\tau}$ if $a \in I$	TI2DR	$\nu(x + y) = \nu(x) + \nu(y)$	DRTO2
$\tau_I(x + y) = \tau_I(x) + \tau_I(y)$	TI3	$\nu(x \cdot y) = \nu(x) \cdot y$	DRTO3
$\tau_I(x \cdot y) = \tau_I(x) \cdot \tau_I(y)$	TI4	$\nu(\sigma(x)) = \underline{\delta}$	DRTO4
$\tau_I(\sigma(x)) = \sigma(\tau_I(x))$	DRTI		

$\underline{a} \cdot \underline{\tau} = \underline{a}$	DRB1
$\underline{a} \cdot (\underline{\tau} \cdot (\nu(x) + y) + \nu(x)) = \underline{a} \cdot (\nu(x) + y)$	DRB2
$\underline{a} \cdot (\underline{\tau} \cdot (\nu(x) + y) + y) = \underline{a} \cdot (\nu(x) + y)$	DRB3
$\underline{a} \cdot (\sigma(\underline{\tau} \cdot x) + \nu(y)) = \underline{a} \cdot (\sigma(x) + \nu(y))$	DRB4

The notation $\sigma^n(t)$, where $n \in \mathbb{N}$, is used for the n-fold application of σ to t, i.e. $\sigma^0(t) = t$ and $\sigma^{n+1}(t) = \sigma(\sigma^n(t))$.

The commutativity and associativity of the operators $+$ and $\parallel$ permit the use of the notations $\sum_{i \in \mathcal{I}} t_i$ and $\big\|_{i \in \mathcal{I}} t_i$, where $\mathcal{I} = \{i_1, \ldots, i_n\}$, for the alternative composition $t_{i_1} + \ldots + t_{i_n}$ and the parallel composition $t_{i_1} \parallel \ldots \parallel t_{i_n}$, respectively. The convention is used that $\sum_{i \in \mathcal{I}} t_i$ and $\big\|_{i \in \mathcal{I}} t_i$ stand for $\underline{\delta}$ if $\mathcal{I} = \emptyset$. If $\mathcal{I} = \{i \in \mathbb{N} \mid \phi_1(i) \wedge \ldots \wedge \phi_n(i)\}$, we write $\sum_{\phi_1(i), \ldots, \phi_n(i)} t_i$ for $\sum_{i \in \mathcal{I}} t_i$ and $\big\|_{\phi_1(i), \ldots, \phi_n(i)} t_i$ for $\big\|_{i \in \mathcal{I}} t_i$.

An $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term t' is said to be a *summand* of an $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term t if there exists a $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term t'' such that $t = t' + t''$ or $t = t'$ is derivable from axioms A1 and A2.

All processes that can be denoted by a closed $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term can be denoted by a basic term over $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$. The set $\mathcal{B}$ of *basic terms* over $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ is the smallest set satisfying:

 - if $a \in \mathsf{A}_{\tau\delta}$, then $\underline{a} \in \mathcal{B}$;
 - if $a \in \mathsf{A}_{\tau}$ and $t \in \mathcal{B}$, then $\underline{a} \cdot t \in \mathcal{B}$;
 - if $t \in \mathcal{B}$, then $\sigma(t) \in \mathcal{B}$;
 - if $t, t' \in \mathcal{B}$, then $t + t' \in \mathcal{B}$.

Modulo axioms A1 and A2, each basic term from $\mathcal{B}$ is of the following form:

$$\sum_{1 \leq i \leq n} \underline{a_i} \cdot t_i + \sum_{1 \leq j \leq m} \underline{\underline{b_j}} + \sum_{1 \leq k \leq l} \sigma(t'_k) \, ,$$

where $n, m, l \in \mathbb{N}$, $a_i \in \mathsf{A}_{\tau}$ and $t_i \in \mathcal{B}$ for $1 \leq i \leq n$, $b_j \in \mathsf{A}_{\tau}$ for $1 \leq j \leq m$, and $t'_k \in \mathcal{B}$ for $1 \leq k \leq l$. Moreover, modulo axioms A1 and A2, each basic term from $\mathcal{B}$ can be proved equal to a basic term of this form where $l \leq 1$ by applications of axiom DRT1.

The following theorem states that each closed $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term can be reduced to a basic term.

Theorem 1 (Elimination). *For each closed $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term t, there exists a basic term $t' \in \mathcal{B}$ such that $t = t'$ is derivable from the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$.*

Proof. The proof goes by induction on the size of t. Corollary 5.3.2.8 from [29] states that, for each closed term t of $\mathrm{ACP}_{\mathrm{drt}}$, there exists a basic term $t' \in \mathcal{B}$ such that $t = t'$ is derivable from the axioms of $\mathrm{ACP}_{\mathrm{drt}}$.[1] The proof of this corollary given in [29] carries over to the setting with closed terms in which $\underline{\tau}$ may occur. Because the axioms of $\mathrm{ACP}_{\mathrm{drt}}$ are included in the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$, this means that the only case that remains to be covered is the case where t is of the form $\tau_I(t')$. By the induction hypothesis, it is sufficient to prove that, for each closed term of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ of the form $\tau_I(t')$ with $t' \in \mathcal{B}$, there exists a basic term $t'' \in \mathcal{B}$ such that $\tau_I(t') = t''$ is derivable from the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$. This is easily proved by induction on the structure of t'. □

In Table 2, some equations concerning parallel composition are given that are derivable for closed terms from the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ and valid for rooted branching bisimilarity as defined in Section 4. These equations are called the *axioms of standard concurrency.*

Theorem 2 (Standard Concurrency). *All closed substitution instances of the axioms of standard concurrency are derivable from the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$.*

---

[1] In [29], $\mathrm{ACP}_{\mathrm{drt}}$ is called $\mathrm{ACP}^{-}_{\mathrm{drt}}$-ID.

Table 2. Axioms of standard concurrency

$x \parallel y = y \parallel x$	$x \mid y = y \mid x$
$(x \parallel y) \parallel z = x \parallel (y \parallel z)$	$(x \mid y) \mid z = x \mid (y \mid z)$
$(x \mathbin{\underline{\parallel}} y) \mathbin{\underline{\parallel}} z = x \mathbin{\underline{\parallel}} (y \parallel z)$	$x \mid (y \mathbin{\underline{\parallel}} z) = (x \mid y) \mathbin{\underline{\parallel}} z$

Proof. The proof goes very similar to the proof of Theorem 4.3.3 from [6]. Here, by Theorem 1, it is sufficient to prove the theorem only for all closed $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ terms of the form $\sum_{1 \leq i \leq n} \underline{a_i} \cdot t_i + \sum_{1 \leq j \leq m} \underline{b_j} + \sum_{1 \leq k \leq l} \sigma(t'_k)$ where $l \leq 1$. The possibility of a summand of the form $\sigma(t)$ has only a minor effect on the complexity of the proof. This is the case because the left merge and communication merge of many combinations of summands that involve a summand of the form $\sigma(t)$ are derivably equal to $\underline{\delta}$ by one of the axioms DRCM1–DRCM4. □

In many applications, communication is synchronous communication between two processes. In those applications, $\gamma(\gamma(a, b), c) = \delta$ for all $a, b, c \in \mathsf{A}$. This kind of communication is called *handshaking communication*. Under the assumption of handshaking communication, the equation $\nu(x \mid y \mid z) = \underline{\delta}$ is derivable for closed terms. This equation is called the *handshaking axiom*.

An important result is the following expansion theorem, which is useful in the elimination of parallel compositions in terms of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ in the case where all communication is handshaking communication.

Theorem 3 (Expansion). *In* $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ *extended with the axioms of standard concurrency and the handshaking axiom, the following equation is derivable for all $n > 2$:*

$$
x_1 \parallel \ldots \parallel x_n = \sum_{1 \leq i \leq n} x_i \mathbin{\underline{\parallel}} \left(\mathop{\parallel}_{\substack{1 \leq j \leq n, \\ j \neq i}} x_j \right) + \sum_{\substack{1 \leq i \leq n, \\ i < j \leq n}} (x_i \mid x_j) \mathbin{\underline{\parallel}} \left(\mathop{\parallel}_{\substack{1 \leq k \leq n, \\ k \neq i, k \neq j}} x_k \right).
$$

Proof. The proof goes the same as the proof of Theorem 4.3.5 from [6]. □

We will use the standardized terminology and notation for handshaking communication that were introduced for ACP in [8]. Processes send, receive and communicate data at *ports*. If a port is used for communication between two processes, it is called *internal*. Otherwise, it is called *external*. We write:

- $s_i(d)$ for the action of sending datum d at port i;
- $r_i(d)$ for the action of receiving datum d at port i;
- $c_i(d)$ for the action of communicating datum d at port i.

The action $c_i(d)$ is the action that is left when $s_i(d)$ and $r_i(d)$ are performed synchronously.

2.2 $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ with Guarded Recursion

A closed $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ term denotes a process with a finite upper bound to the number of actions that it can perform. Guarded recursion allows the description of processes without a finite upper bound to the number of actions that it can perform. In this section, we extend $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ with guarded recursion by adding constants for solutions of guarded recursive specifications and axioms concerning these additional constants. We write $\mathrm{ACP}^{\tau}_{\mathrm{drt}}{+}\mathrm{REC}$ for the resulting theory.

Let Σ be a signature that includes the signature of $\mathrm{BPA}^{\tau}_{\mathrm{drt}}$, let X be a variable from $\mathcal{X}$, and let t be a term over Σ in which X occurs. Then an occurrence of X in t is *guarded* if no abstraction operator occurs in t and t has a subterm of the form $a \cdot t'$ where $a \in \mathsf{A}$ or the form $\sigma(t')$ and t' contains this occurrence of X. A term t over Σ is a *guarded* term over Σ if all occurrences of variables in t are guarded.

Let Σ be a signature that includes the signature of $\mathrm{BPA}^{\tau}_{\mathrm{drt}}$. Then a *recursive specification* over Σ is a set $\{X_i = t_i \mid i \in I\}$, where I is an index set, each X_i is a variable from $\mathcal{X}$, each t_i is a term over Σ in which only variables from $\{X_i \mid i \in I\}$ occur, and $X_i \neq X_j$ for all $i, j \in I$ with $i \neq j$. We write $\mathrm{V}(E)$, where E is a recursive specification $\{X_i = t_i \mid i \in I\}$ over Σ, for the set $\{X_i \mid i \in I\}$. Let E be a recursive specification over Σ and let $X \in \mathrm{V}(E)$. Then the unique equation $X = t \in E$ is called the *recursion equation for X in E*.

Let Σ be a signature that includes the signature of $\mathrm{BPA}^{\tau}_{\mathrm{drt}}$, and let Φ be a set of equations over the signature Σ that includes the axioms of $\mathrm{BPA}^{\tau}_{\mathrm{drt}}$. A recursive specification $\{X_i = t_i \mid i \in I\}$ over Σ is *guarded modulo Φ* if each t_i is rewritable to a guarded term over Σ by using the equations from Φ in either direction and/or the equations in $\{X_j = t_j \mid j \in I \wedge i \neq j\}$ from left to right.

A solution of a guarded recursive specification $\{X_i = t_i \mid i \in I\}$ in a semantics of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}{+}\mathrm{REC}$ is a set $\{p_i \mid i \in I\}$ of processes in that semantics such that each equation in $\{X_i = t_i \mid i \in I\}$ holds if, for each $i \in I$, X_i is assigned p_i. Here, p_i is said to be the X_i-component of the solution. A guarded recursive specification has a unique solution for rooted branching bisimilarity as defined on an operational semantics of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}{+}\mathrm{REC}$ in Section 4. A recursive specification that is not guarded may not have a unique solution. For example, $\{x = \underline{a} + x\}$ and $\{x = x \cdot \underline{a}\}$ have infinitely many solutions for rooted branching bisimilarity.

Below, for particular signatures Σ that include the signature of $\mathrm{BPA}^{\tau}_{\mathrm{drt}}$ and particular sets Φ of equations over Σ that include the axioms of $\mathrm{BPA}^{\tau}_{\mathrm{drt}}$, for each recursive specification E over Σ that is guarded modulo Φ and $X \in \mathrm{V}(E)$, a constant $\langle X | E \rangle$ that stands for the X-component of the unique solution of E will be introduced. The notation $\langle t | E \rangle$ will be used for t with, for all $X \in \mathrm{V}(E)$, all occurrences of X in t replaced by $\langle X | E \rangle$.

The signature and axiom system of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}{+}\mathrm{REC}$ are $\bigcup_{i \in \mathbb{N}} \Sigma_i$ and $\bigcup_{i \in \mathbb{N}} \Phi_i$, respectively, where:

- Σ_0 is the signature of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$;
- Φ_0 is the axiom system of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$;
- Σ_{i+1} is Σ_i with added, for each recursive specification E over Σ_i that is guarded modulo Φ_i and $X \in \mathrm{V}(E)$, a *recursion* constant $\langle X | E \rangle$;

9

Table 3. Additional axioms for $\text{ACP}^\tau_{\text{drt}}+\text{REC}$

$\langle X|E\rangle = \langle t|E\rangle$	if $X = t \in E$	RDP
$E \Rightarrow X = \langle X|E\rangle$	if $X \in \text{V}(E)$	RSP

– Φ_{i+1} is Φ_i with added, for each recursive specification E over Σ_i that is guarded modulo Φ_i and $X \in \text{V}(E)$, the equation $\langle X|E\rangle = \langle t|E\rangle$ for the unique term t over Σ_i such that $X = t \in E$ and the conditional equation $E \Rightarrow X = \langle X|E\rangle$.

The equations and conditional equations added to the axiom system of $\text{ACP}^\tau_{\text{drt}}$ to obtain the axiom system of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ are the instances of the axiom schemas RDP and RSP, respectively, given in Table 3. In these axiom schemas, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ term, and E stands for an arbitrary recursive specification over the signature of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ that is guarded modulo the axioms of $\text{ACP}^\tau_{\text{drt}}+$ REC. Side conditions restrict what X, t and E stand for.

For a fixed E, the equations $\langle X|E\rangle = \langle t|E\rangle$ and the conditional equations $E \Rightarrow X = \langle X|E\rangle$ express that the constants $\langle X|E\rangle$ make up a solution of E and that this solution is the only one.

Because conditional equations must be dealt with in $\text{ACP}^\tau_{\text{drt}}+\text{REC}$, it is understood that conditional equational logic is used in deriving equations from the axioms of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$. A complete inference system for conditional equational logic can be found in [6,17].

In the setting of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$, we use a, where $a \in \mathsf{A}_{\tau\delta}$, as an abbreviation of $\langle X|X = \underline{a} + \sigma(X)\rangle$. For $a \in \mathsf{A}_\tau$, this means that a denotes the process that performs the action a in the current or any future time slice and after that immediately terminates successfully.

The abbreviation just introduced can be used in guarded recursive specifications because $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ allows nested recursion. This shows that nested recursion can be convenient. However, $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ would not be less expressive with unnested recursion.

Theorem 4. *For each recursion constant $\langle X|E\rangle$ of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$, there exists a recursion constant $\langle X|E'\rangle$ of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ with no recursion constant of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ occurring in E' such that $\langle X|E\rangle = \langle X|E'\rangle$ is derivable from the axioms of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$.*

Proof. In this proof, Σ_n refers to the signature Σ_n from the definition of the signature of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$. The proof goes by induction on the nesting level of $\langle X|E\rangle$, i.e. the smallest $n > 0$ such that $\langle X|E\rangle \in \Sigma_n$. The basis step is trivial because there are no occurrences of recursion constants of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ in E if $\langle X|E\rangle \in \Sigma_1$. The inductive step is proved in the following way. Assume that all recursion constants occurring in E belong to Σ_1. This is allowed by the induction hypothesis. Moreover, assume that, for all recursion constants $\langle X_1|E_1\rangle$ and $\langle X_2|E_2\rangle$ that occur in E, (a) $\text{V}(E_1) \cap \text{V}(E_2) = \emptyset$ if $E_1 \neq E_2$ and

10

(b) $V(E) \cap V(E_1) = \emptyset$. This is allowed by the fact that equality of a recursion constant of $\text{ACP}_{\text{drt}}^\tau + \text{REC}$ and a variable renaming of it is always derivable from RDP and RSP. For E', take the guarded recursive specification obtained from E by first replacing each recursion constant $\langle X''|E''\rangle$ occurring in E by the right-hand side of the recursion equation for X'' in E'' and then, for each $\langle X''|E''\rangle$ occurring in E adding the equations in E'' to E. Be aware that (a) for each right-hand side t of an equation in E, $\langle t|E\rangle$ and $\langle t|E'\rangle$ stand for the same term and (b) for each right-hand side t of an equation in those E'' for which there occurs a recursion constant $\langle X''|E''\rangle$ in E, $\langle t|E''\rangle$ and $\langle t|E'\rangle$ stand for the same term. Now, first apply RDP for each equation in E and each equation in those recursive specifications E'' for which there occurs a recursion constant $\langle X''|E''\rangle$ in E and then apply the instance of RSP for X to the resulting set of equations. This yields $\langle X|E\rangle = \langle X|E'\rangle$. $\square$

To my knowledge, in the setting of ACP-style process algebras, nested guarded recursion has only been considered before in [23]. In that paper, only a restricted form of nested guarded recursion is considered.

A system is often described by a term of the form $\partial_H(\langle X_1|E_1\rangle \| \ldots \| \langle X_n|E_n\rangle)$, i.e. as the encapsulated parallel composition of a number of processes that are described by guarded recursive specifications. The first step in analyzing such a system is usually the extraction of a guarded recursive specification of the system from the term describing it. This involves mere algebraic calculations, in which the expansion theorem plays an important part, and finally application of RSP. In case the functional correctness is analyzed, we can proceed by extracting a guarded recursive specification from a term of the form $\tau_I(\langle X|E\rangle)$ or the form $\pi_{\text{tf}}(\tau_I(\langle X|E\rangle))$, where E is the result of the first step and π_{tf} is the time-free projection operator described below. This involves again application of RSP.

Suppose that E is a guarded recursive specification that includes the equation

$$X = \sum_{i \in \mathcal{I}} \underline{\underline{a_i}} \cdot t_i + \sum_{j \in \mathcal{J}} \underline{\underline{b_j}} + \sigma(X) \, .$$

In that case, we can derive the equation

$$\langle X|E\rangle = \sum_{i \in \mathcal{I}} a_i \cdot \langle t_i|E\rangle + \sum_{j \in \mathcal{J}} b_j \, .$$

We make use of this fact in each of the guarded recursive specifications given in Section 3.

We usually write X for $\langle X|E\rangle$ if E is clear from the context. In those cases, it should also be clear from the context that we use X as a constant.

We write $\text{BPA}_{\text{drt}}^\tau + \text{REC}$ for the theory that is obtained by replacing in the definition of the signature and axiom system of $\text{ACP}_{\text{drt}}^\tau + \text{REC}$ given above all occurrences of "$\text{ACP}_{\text{drt}}^\tau$" by "$\text{BPA}_{\text{drt}}^\tau$". $\text{BPA}_{\text{drt}}^\tau + \text{REC}$ plays a role in Section 5.

In Sections 4 and 5, we write $\mathcal{P}_{\text{rec}}$ for the set of all closed $\text{ACP}_{\text{drt}}^\tau + \text{REC}$ terms.

11

Table 4. Defining axioms of the time-free projection operator

$\pi_{\sf tf}(\underline{a}) = \langle X \vert X = \underline{a} + \sigma(X) \rangle$	DRTFP1
$\pi_{\sf tf}(x + y) = \pi_{\sf tf}(x) + \pi_{\sf tf}(y)$	DRTFP2
$\pi_{\sf tf}(x \cdot y) = \pi_{\sf tf}(x) \cdot \pi_{\sf tf}(y)$	DRTFP3
$\pi_{\sf tf}(\sigma(x)) = \pi_{\sf tf}(x)$	DRTFP4

2.3 Time-free projection

$\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC can be extended with the unary *time-free projection* operator $\pi_{\sf tf}$. This operator makes the process denoted by its operand time free. For any time-free process, the following holds: it is always capable of idling till a future time slice and it is never bound to idle till a future time slice. Thus, time-free projection amounts to abstraction from the timing of actions. The defining axioms for the time-free projection operator are given in Table 4. In this table, a stands for an arbitrary member of $\mathsf{A}_{\tau\delta}$.

The time-free projection operator relates the timed theory $\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC to the untimed theory ACP^τ+REC (ACP^τ from [6] with guarded recursion as in Section 2.2). For each ACP^τ+REC term t, let $\epsilon(t)$ be the $\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC term obtained by replacing, for each $a \in \mathsf{A}_{\tau\delta}$, all occurrences of a in t by $\langle X \vert X = \underline{a} + \sigma(X) \rangle$. Then, for each equation $t = t'$ that can be derived from the axioms of ACP^τ+REC, the equation $\epsilon(t) = \epsilon(t')$ can be derived from the axioms of $\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC. Moreover, for each $\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC term t, there exists an ACP^τ+REC term t' such that $\pi_{\sf tf}(t) = \epsilon(t')$ is derivable from the axioms of $\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC.

3 The PAR Protocol

In this section, $\mathrm{ACP}^\tau_{\mathrm{drt}}$+REC is used to formally specify a simple version of the communication protocol known as the PAR protocol and to analyze it. Timing is essential for the correct behaviour of this protocol. The treatment of the PAR protocol in this section is largely a slightly adapted shortened version of its treatment in Sections 2.2.1, 2.2.4, and 6.2.3 of [4].[2]

3.1 Informal Description of the PAR Protocol

As its name suggests, the PAR protocol is a communication protocol based on time-outs of positive acknowledgements. The sender waits for a positive acknowledgement before a new datum is transmitted. If an acknowledgement is not received within a complete protocol cycle, the old datum is retransmitted. In order to avoid duplicates due to retransmission, data are labeled with an alternating bit from $B = \{0, 1\}$. The configuration of the PAR protocol is shown in Fig. 1 by means of a connection diagram.

---

[2] The treatment of the PAR protocol in [4] have been copied almost verbatim without mentioning its origin in at least one other publication.

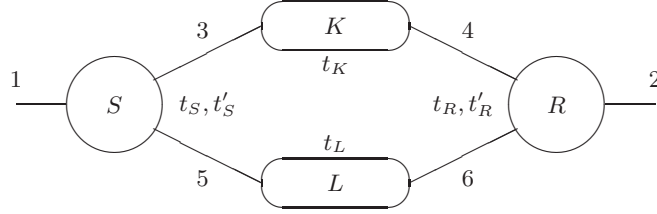**Fig. 1.** Connection diagram for PAR protocol

We have a sender process S, a receiver process R, and two channels K and L. The process S waits until a datum d is offered at an external port (port 1). When a datum is offered at this port, S consumes it, packs it with an alternating bit b in a frame (d, b), and then delivers the frame at an internal port used for sending (port 3). Next, S waits until an acknowledgement ack is offered at an internal port used for receiving (port 5). When the acknowledgement does not arrive within a certain time period, S delivers the same frame again and goes back to waiting for an acknowledgement. When the acknowledgement arrives within that time period, S goes back to waiting for a datum, but the alternating bit changes to $(1 - b)$. The process R waits until a frame (d, b) is offered at an internal port used for receiving (port 4). When a frame is offered at this port, R consumes it, unpacks it, and then delivers the datum d at an external port (port 2) if the alternating bit b is the right one and in any case an acknowledgement ack at an internal port used for sending (port 6). After that, R goes back to waiting for a frame, but the right bit changes to $(1 - b)$ if the alternating bit was the right one. The processes K and L pass on frames from an internal port of S to an internal port of R and acknowledgements from an internal port of R to an internal port of S, respectively. Because the channels are supposed to be unreliable, they may produce an error instead of passing on frames or acknowledgements. The times t_S, t_R, t_K, t_L are the times that it takes the different processes to pack and deliver, to unpack and deliver or simply to deliver what they consume. The time t'_S is the time-out time of the sender, i.e., the time after which it retransmits a datum in the case where it is still waiting for an acknowledgement. The time t'_R is the time that it takes the receiver to produce and deliver an acknowledgement.

3.2   Formal Specification of the PAR Protocol

We assume that the times $t_S, t_R, t_K, t_L, t'_S, t'_R$ are non-zero. We also assume a finite set of data D. Let $F = D \times B$ be the set of frames. For $d \in D$ and $b \in B$, we write d, b for the frame (d, b). We use the standardized notation for handshaking communication introduced in Section 2. The recursive specification of the sender consists of the following equations:

$$S \quad = S_0 \ ,$$

$$S_b \quad = \sum_{d \in D} \underline{r_1(d)} \cdot \sigma^{t_S}(SF_{d,b}) + \sigma(S_b)$$

(for every $b \in B$),

$$SF_{d,b} = \underline{s_3(d,b)} \cdot \left(\sum_{k < t'_S} \sigma^k(\underline{r_5(ack)}) \cdot S_{1-b} + \sigma^{t'_S}(SF_{d,b}) \right)$$

(for every $d \in D$ and $b \in B$).

The recursive specification of the receiver consists of the following equations:

$$R \ = R_0 \ ,$$

$$R_b = \sum_{d \in D} \underline{r_4(d,b)} \cdot \sigma^{t_R}(\underline{s_2(d)}) \cdot \sigma^{t'_R}(\underline{s_6(ack)}) \cdot R_{1-b}$$
$$+ \sum_{d \in D} \underline{r_4(d,1-b)} \cdot \sigma^{t'_R}(\underline{s_6(ack)}) \cdot R_b + \sigma(R_b)$$

(for every $b \in B$).

Each of the channels is recursively defined by a single equation:

$$K = \sum_{f \in F} \underline{r_3(f)} \cdot \left(\sigma^{t_K}(\underline{s_4(f)}) + \sum_{k \le t_K} \sigma^k(\underline{error}) \right) \cdot K + \sigma(K) \ ,$$

$$L = \underline{r_6(ack)} \cdot \left(\sigma^{t_L}(\underline{s_5(ack)}) + \sum_{k \le t_L} \sigma^k(\underline{error}) \right) \cdot L + \sigma(L) \ .$$

The whole system is described by the following term:

$$\partial_H(S \parallel K \parallel L \parallel R) \ ,$$

where

$$H = \{s_i(f), r_i(f) \mid i \in \{3,4\}, f \in F\} \cup \{s_i(ack), r_i(ack) \mid i \in \{5,6\}\} \ .$$

This protocol is only correct if the time-out time t'_S is longer than a complete protocol cycle, i.e., if $t'_S > t_K + t_R + t'_R + t_L$. If the time-out time is shorter than a complete protocol cycle, the time-out is called premature. In that case, while an acknowledgement is still on the way, the sender will retransmit the current frame. When the acknowledgement finally arrives, the sender will treat this acknowledgement as an acknowledgement of the retransmitted frame. However, an acknowledgement of the retransmitted frame may be on the way. If the next frame transmitted gets lost and the latter acknowledgement arrives, no retransmission of that frame will follow and the protocol will fail.

3.3 Analysis of the PAR Protocol: Expansion

We can analyze the PAR protocol described in Section 3.2 using a technique similar to the basic one used in the case without timing.

In Section 3.2, we first gave guarded recursive specifications of the sender process S, the receiver process R and the channel processes K and L, and then described the whole PAR protocol by the term $\partial_H(S \parallel K \parallel L \parallel R)$. Because all communication is handshaking communication, the expansion theorem for $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ (see Section 2) is applicable. By using this expansion theorem and RSP, we are able to give a guarded recursive specification of the whole PAR protocol.

First, we rewrite the recursive specifications of S, R, K and L, using their equations and the axioms of $\mathrm{ACP}_{\mathrm{drt}}$, to ones in a form that is better suited to expansion.

Secondly, we expand the term $\partial_H(S \parallel K \parallel L \parallel R)$ by repeated application of the expansion theorem. Except for the first step, we expand a subterm of the right-hand side of a previous equation that has not been expanded before. After a number of applications of the expansion theorem, the terms on the left-hand sides of the equations obtained in this way will include all unexpanded terms on the right-hand sides iff $t'_S > t_K + t_R + t'_R + t_L$. At that point, we have that the terms on the left-hand sides of the equations make up a solution of the guarded recursive specification obtained by replacing all occurrences of these terms in the equations by occurrences of corresponding variables. Let X be the corresponding variable for $\partial_H(S \parallel K \parallel L \parallel R)$. Then we derive from RSP that $\partial_H(S \parallel K \parallel L \parallel R)$ is the X-component of the solution of this guarded recursive specification. The guarded recursive specification concerned can easily be rewritten, using its equations and the axioms of $\mathrm{ACP}_{\mathrm{drt}}$, to the following guarded recursive specification:

$$X = X_0 \,,$$

$$X_b = \sum_{d \in D} \underline{r_1(d)} \cdot \sigma^{t_S}(Y_{d,b}) + \sigma(X_b) \,,$$

$$Y_{d,b} = \underline{c_3(d,b)} \cdot \left(\sigma^{t_K}(\underline{c_4(d,b)}) \cdot \sigma^{t_R}(\underline{s_2(d)}) \cdot \sigma^{t'_R}(\underline{c_6(ack)}) \cdot Z_{d,b} \right.$$
$$\left. + \sum_{k \le t_K} \sigma^k(\underline{error}) \cdot \sigma^{t'_S - k}(Y_{d,b}) \right) \,,$$

$$Z_{d,b} = \sigma^{t_L}(\underline{c_5(ack)}) \cdot X_{1-b} + \sum_{k \le t_L} \sigma^k(\underline{error}) \cdot \sigma^{t'_S - (t_K + t_R + t'_R + k)}(U_{d,b}) \,,$$

$$U_{d,b} = \underline{c_3(d,b)} \cdot \left(\sigma^{t_K}(\underline{c_4(d,b)}) \cdot \sigma^{t'_R}(\underline{c_6(ack)}) \cdot V_{d,b} \right.$$
$$\left. + \sum_{k \le t_K} \sigma^k(\underline{error}) \cdot \sigma^{t'_S - k}(U_{d,b}) \right) \,,$$

$$V_{d,b} = \sigma^{t_L}(\underline{c_5(ack)}) \cdot X_{1-b} + \sum_{k \le t_L} \sigma^k(\underline{error}) \cdot \sigma^{t'_S - (t_K + t'_R + k)}(U_{d,b}) \,.$$

From this recursive specification we can conclude informally that, if we abstract from all actions other than the send and receive actions at the external ports 1 and 2 and in addition from the timing of actions, the whole PAR protocol behaves functionally correct, i.e. as a buffer with capacity one, provided $t'_S > t_K + t_R + t'_R + t_L$. Section 3.4 starts with an outline of how this conclusion can be drawn in a formal way.

Details of the analysis outlined in this section can be found in Section 2.2.4 of [4].

3.4 Analysis of the PAR Protocol: Abstraction

We want to abstract from all actions other than the send and receive actions at the external ports 1 and 2, i.e. from the actions in the set

$$I = \{c_i(d,b) \mid i \in \{3,4\}, d \in D, b \in B\} \cup \{c_5(ack), c_6(ack), error\} \ .$$

We can proceed in different ways. First of all, we can focus on functional correctness. This means that we do not only abstract from the actions in the set I, but also from all timing of actions (using the time free projection operator from Section 2.3). After the abstraction from all timing of actions, we can proceed in a theory without timing, to wit $ACP^\tau + REC$ (see Section 2.3). Starting from the specification of $\partial_H(S \parallel K \parallel L \parallel R)$ at the end of Section 3.3, we can easily calculate that $\pi_{\mathsf{tf}}(\partial_H(S \parallel K \parallel L \parallel R))$ is the X'-component of the solution of the guarded recursive specification that consists of the following equations:

$$X' \ = X'_0 \ ,$$

$$X'_b \ = \sum_{d \in D} r_1(d) \cdot Y'_{d,b} \ ,$$

$$Y'_{d,b} = c_3(d,b) \cdot (c_4(d,b) \cdot s_2(d) \cdot c_6(ack) \cdot Z'_{d,b} + error \cdot Y'_{d,b}) \ ,$$

$$Z'_{d,b} = c_5(ack) \cdot X'_{1-b} + error \cdot U'_{d,b} \ ,$$

$$U'_{d,b} = c_3(d,b) \cdot (c_4(d,b) \cdot c_6(ack) \cdot V'_{d,b} + error \cdot U'_{d,b}) \ ,$$

$$V'_{d,b} = c_5(ack) \cdot X'_{1-b} + error \cdot U'_{d,b} \ .$$

Starting from this specification, we can calculate, using axiom $x \cdot (\tau \cdot (y+z)+z) = x \cdot (y+z)$ (B2) from ACP^τ, together with CFAR (Cluster Fair Abstraction Rule) to remove cycles of silent steps, that $\tau_I(\pi_{\mathsf{tf}}(\partial_H(S \parallel K \parallel L \parallel R)))$, which equals $\pi_{\mathsf{tf}}(\tau_I(\partial_H(S \parallel K \parallel L \parallel R)))$, is the solution of the guarded recursive specification that consists of the following equation:

$$B = \sum_{d \in D} r_1(d) \cdot s_2(d) \cdot B \ .$$

For more information on the silent step in ACP^τ, including details about CFAR, see [6]. We have obtained a guarded recursive specification of a buffer with

capacity one. Thus, we see that the PAR protocol is functionally correct. We want to stress that, in order to achieve this result, it was necessary to calculate first the time-dependent behavior of the whole protocol, because the PAR protocol is only correct if the timing parameters are set correctly. A complete verification in process algebra without timing is not possible without resorting to artificial tricks such as excluding the premature time-out of an acknowledgement by inhibiting a time-out so long as it does not lead to deadlock (see e.g. [27]).

Next, we can have a look at the performance properties. Starting from the specification of $\partial_H(S \,\|\, K \,\|\, L \,\|\, R)$ at the end of Section 3.3, we can easily calculate that $\tau_I(\partial_H(S \,\|\, K \,\|\, L \,\|\, R))$ is the X''-component of the solution of the guarded recursive specification that consists of the following equations:

$$X'' = \sum_{d \in D} \underline{r_1(d)} \cdot \sigma^{t_S}(Y_d'') + \sigma(X'') \,,$$

$$Y_d'' = \sigma^{t_K}(\underline{\tau} \cdot \sigma^{t_R}(\underline{s_2(d)} \cdot \sigma^{t_R'}(Z''))) + \sum_{k \le t_K} \sigma^k(\underline{\tau} \cdot \sigma^{t_S'-k}(Y_d'')) \,,$$

$$Z'' = \sigma^{t_L}(\underline{\tau} \cdot X'') + \sum_{k \le t_L} \sigma^k(\underline{\tau} \cdot \sigma^{t_S'-(t_K+t_R+t_R'+k)}(U'')) \,,$$

$$U'' = \sigma^{t_K}(\underline{\tau} \cdot \sigma^{t_R'}(V'')) + \sum_{k \le t_K} \sigma^k(\underline{\tau} \cdot \sigma^{t_S'-k}(U'')) \,,$$

$$V'' = \sigma^{t_L}(\underline{\tau} \cdot X'') + \sum_{k \le t_L} \sigma^k(\underline{\tau} \cdot \sigma^{t_S'-(t_K+t_R'+k)}(U'')) \,.$$

Not many simplifications have been achieved. This is mainly because we cannot leave out silent steps that occur in between delays. In effect, all internal choices that may be made, e.g. whether or not a channel forwards a datum correctly, remain visible. Some initial observations concerning this matter, as well as an analysis of a slightly different version of the PAR protocol, were made in an unpublished paper of Dragan Bošnački (1997). In any case, from this specification, we can conclude informally that the protocol takes at least $t_S + t_K + t_R$ time slices between consumption and delivery of a datum, and in general, between consumption and delivery we have $t_S + t_K + t_R + i \cdot t_S'$ time slices, where $i \ge 0$. After delivery, at least $t_R' + t_L$ time slices must pass before the next datum can be consumed, and in general, between delivery and readiness for the next consumption, we have $t_R' + t_L$ or $t_R' + t_L + j \cdot t_S' - t_R$ time slices, where $j > 0$.

Details of the analysis outlined in this section can be found in Section 6.2.3 of [4].

4 The Standard Version of Branching Bisimilarity

For the relevant versions of ACP, the accepted definition of the standard version of branching bisimilarity for processes with discrete relative timing refers to a two-phase operational semantics of the version concerned, i.e. an operational semantics where processes can make transitions by performing an action in the current time slice or by idling till the next time slice. In Section 5, a definition

of a variant of the standard version of branching bisimilarity for processes with discrete relative timing will be given. That definition refers to a time-stamped operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$, i.e. an operational semantics where processes can make transitions by idling till a time slice in which an action can be performed and subsequently performing that action.

In this section, a two-phase operational semantics and a time-stamped operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ is presented and the standard version of branching bisimilarity for processes with discrete relative timing is defined once referring to the two-phase operational semantics and once referring to the time-stamped operational semantics. This way, it becomes easier to see the essential difference between the standard version of branching bisimilarity for processes with discrete relative timing and the variant introduced in Section 5.

4.1 A Definition Referring to a Two-Phase Semantics

The definition of the standard version of branching bisimilarity given below refers to a two-phase structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ that consists of:

a binary *transition* relation $\_ \xrightarrow{\alpha} \_ \subseteq \mathcal{P}_{\mathsf{rec}} \times (\mathcal{P}_{\mathsf{rec}} \cup \{\surd\})$ for each $\alpha \in \mathsf{A}_\tau \cup \{\sigma\}$, where, for each $t \in \mathcal{P}_{\mathsf{rec}}$, not $t \xrightarrow{\sigma} \surd$.

The relations from this structural operational semantics describe what the processes denoted by terms from $\mathcal{P}_{\mathsf{rec}}$ are capable of doing as follows:

if $\alpha \in \mathsf{A}_\tau$, then $t \xrightarrow{\alpha} t'$ indicates that the process denoted by t has the potential to make a transition to the process denoted by t' by performing action α in the current time slice;

if $\alpha = \sigma$, then $t \xrightarrow{\alpha} t'$ indicates that the process denoted by t has the potential to make a transition to the process denoted by t' by idling till the next time slice.

Moreover, $\surd$ denotes the process that is only capable of terminating successfully in the current time slice. There exists no $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ term that denotes this process.

The relations from the two-phase structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ are defined below by means of rules of which some have negative premises, i.e. premises of the form $\neg (t \xrightarrow{\alpha} t')$. Because of this, it is not so obvious what relations are defined by the rules concerned. An informal explanation is given here. The relations defined by these rules are the unique ones that constitute an operational semantics such that $t \xrightarrow{\alpha} t'$ holds in that operational semantics iff $t \xrightarrow{\alpha} t'$ can be deduced from the rules under some set of negative assumptions that hold in that operational semantics. The uniqueness follows from the fact that the rules are such that no closed substitution instance of the conclusion of a rule depends negatively on itself. Formal details and further explanation can for example be found in [14].

We write $t \not\xrightarrow{\sigma}$ for the set of all premises $\neg (t \xrightarrow{\sigma} t')$ where $t' \in \mathcal{P}_{\mathsf{rec}}$.

18

The relations from the two-phase structural operational semantics of $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ are the relations defined by the rules given in Table 5 as explained above. In this table, a, b, and c stand for arbitrary actions from A_τ, H and I stand for arbitrary subsets of A, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary $\text{ACP}^\tau_{\text{drt}}+\text{REC}$ term, and E stands for an arbitrary guarded recursive specification over $\text{ACP}^\tau_{\text{drt}}+\text{REC}$.

In Figure 2, the processes that are denoted by the closed terms $\sigma(\underline{a}) + \sigma(\underline{b})$ and $\underline{\tau} \cdot \sigma(\underline{a}) + \sigma(\underline{b})$ according to the two-phase operational semantics are presented graphically. These graphical presentations show the effect of the peculiar transition rules for alternative composition with respect to idling till the next time slice: in the case of an alternative composition of two processes, the choice between the two is resolved at the instant that one of them performs its first action, and not before.
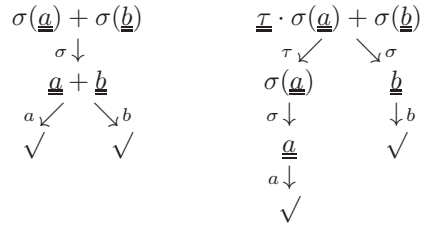

Fig. 2. Graphical two-phase presentation of two processes

Two processes are considered equal if they can simulate each other insofar as their observable potentials to make transitions are concerned. This can be dealt with by means of the notion of branching bisimilarity introduced in [16] adapted to processes with discrete relative timing in the style of [2].

In the definition of rooted branching bisimilarity below, we denote the reflexive and transitive closure of $\xrightarrow{\sigma}$ by $\xrightarrow{\sigma}\!\!\!\twoheadrightarrow$. So $t \xrightarrow{\sigma}\!\!\!\twoheadrightarrow t'$ indicates that the process denoted by t' is reachable from the process denoted by t by idling only.

A *branching bisimulation* is a binary relation R on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ such that, for all $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ with $R(t_1, t_2)$, the following transfer conditions hold:

- if $t_1 \xrightarrow{\alpha} t_1'$, then
    - either $\alpha \equiv \tau$ and $R(t_1', t_2)$
    - or, for some $n \in \mathbb{N}$, there exist $t_0^*, \ldots, t_n^* \in \mathcal{P}_{\mathsf{rec}}$ and $t_2' \in \mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ with $t_0^* \equiv t_2$ such that
        - $*$ for all $i \in \mathbb{N}$ with $i < n$, $t_i^* \xrightarrow{\tau} t_{i+1}^*$ and $R(t_1, t_{i+1}^*)$,
        - $*$ $t_n^* \xrightarrow{\alpha} t_2'$, and $R(t_1', t_2')$;
- if $t_2 \xrightarrow{\alpha} t_2'$, then
    - either $\alpha \equiv \tau$ and $R(t_1, t_2')$
    - or, for some $n \in \mathbb{N}$, there exist $t_0^*, \ldots, t_n^* \in \mathcal{P}_{\mathsf{rec}}$ and $t_1' \in \mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ with $t_0^* \equiv t_1$ such that
        - $*$ for all $i \in \mathbb{N}$ with $i < n$, $t_i^* \xrightarrow{\tau} t_{i+1}^*$ and $R(t_{i+1}^*, t_2)$,
        - $*$ $t_n^* \xrightarrow{\alpha} t_1'$, and $R(t_1', t_2')$.

$$\underline{a} \xrightarrow{a} \surd$$

$$\frac{x \xrightarrow{a} x'}{x+y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x+y \xrightarrow{a} y'} \qquad \frac{x \xrightarrow{a} \surd}{x+y \xrightarrow{a} \surd} \qquad \frac{y \xrightarrow{a} \surd}{x+y \xrightarrow{a} \surd}$$

$$\frac{x \xrightarrow{\sigma} x',\ y \xnrightarrow{\sigma}}{x+y \xrightarrow{\sigma} x'} \qquad \frac{x \xnrightarrow{\sigma},\ y \xrightarrow{\sigma} y'}{x+y \xrightarrow{\sigma} y'} \qquad \frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x+y \xrightarrow{\sigma} x'+y'}$$

$$\frac{x \xrightarrow{a} x'}{x \cdot y \xrightarrow{a} x' \cdot y} \qquad \frac{x \xrightarrow{a} \surd}{x \cdot y \xrightarrow{a} y} \qquad \frac{x \xrightarrow{\sigma} x'}{x \cdot y \xrightarrow{\sigma} x' \cdot y}$$

$$\overline{\sigma(x) \xrightarrow{\sigma} x}$$

$$\frac{x \xrightarrow{a} x'}{x \parallel y \xrightarrow{a} x' \parallel y} \qquad \frac{y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'} \qquad \frac{x \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} y} \qquad \frac{y \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} x}$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y'}{x \parallel y \xrightarrow{c} x' \parallel y'}\ \gamma(a,b)=c \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd}{x \parallel y \xrightarrow{c} x'}\ \gamma(a,b)=c$$

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} y'}{x \parallel y \xrightarrow{c} y'}\ \gamma(a,b)=c \qquad \frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd}{x \parallel y \xrightarrow{c} \surd}\ \gamma(a,b)=c \qquad \frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x \parallel y \xrightarrow{\sigma} x' \parallel y'}$$

$$\frac{x \xrightarrow{a} x'}{x \mathbin{\underline{\parallel}} y \xrightarrow{a} x' \parallel y} \qquad \frac{x \xrightarrow{a} \surd}{x \mathbin{\underline{\parallel}} y \xrightarrow{a} y} \qquad \frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x \mathbin{\underline{\parallel}} y \xrightarrow{\sigma} x' \mathbin{\underline{\parallel}} y'}$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y'}{x \mid y \xrightarrow{c} x' \parallel y'}\ \gamma(a,b)=c \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd}{x \mid y \xrightarrow{c} x'}\ \gamma(a,b)=c$$

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} y'}{x \mid y \xrightarrow{c} y'}\ \gamma(a,b)=c \qquad \frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd}{x \mid y \xrightarrow{c} \surd}\ \gamma(a,b)=c \qquad \frac{x \xrightarrow{\sigma} x',\ y \xrightarrow{\sigma} y'}{x \mid y \xrightarrow{\sigma} x' \mid y'}$$

$$\frac{x \xrightarrow{a} x'}{\partial_H(x) \xrightarrow{a} \partial_H(x')}\ a \notin H \qquad \frac{x \xrightarrow{a} \surd}{\partial_H(x) \xrightarrow{a} \surd}\ a \notin H \qquad \frac{x \xrightarrow{\sigma} x'}{\partial_H(x) \xrightarrow{\sigma} \partial_H(x')}$$

$$\frac{x \xrightarrow{a} x'}{\tau_I(x) \xrightarrow{a} \tau_I(x')}\ a \notin I \qquad \frac{x \xrightarrow{a} \surd}{\tau_I(x) \xrightarrow{a} \surd}\ a \notin I$$

$$\frac{x \xrightarrow{a} x'}{\tau_I(x) \xrightarrow{\tau} \tau_I(x')}\ a \in I \qquad \frac{x \xrightarrow{a} \surd}{\tau_I(x) \xrightarrow{\tau} \surd}\ a \in I \qquad \frac{x \xrightarrow{\sigma} x'}{\tau_I(x) \xrightarrow{\sigma} \tau_I(x')}$$

$$\frac{x \xrightarrow{a} x'}{\nu(x) \xrightarrow{a} x'} \qquad \frac{x \xrightarrow{a} \surd}{\nu(x) \xrightarrow{a} \surd}$$

$$\frac{\langle t|E \rangle \xrightarrow{a} x'}{\langle X|E \rangle \xrightarrow{a} x'}\ X{=}t \in E \qquad \frac{\langle t|E \rangle \xrightarrow{a} \surd}{\langle X|E \rangle \xrightarrow{a} \surd}\ X{=}t \in E \qquad \frac{\langle t|E \rangle \xrightarrow{\sigma} x'}{\langle X|E \rangle \xrightarrow{\sigma} x'}\ X{=}t \in E$$

Two terms $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ are *branching bisimilar*, written $t_1 \underline{\leftrightarrow}_{\mathsf{b}} t_2$, if there exists a branching bisimulation R with $R(t_1, t_2)$. Branching bisimilarity is also called *branching bisimulation equivalence*.

Branching bisimilarity is not a congruence with respect to the operators $+$, $\parallel$, and $|$. This can be remedied by means of a root condition.

If R is a binary relation on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$, then a pair $(t_1, t_2) \in \mathcal{P}_{\mathsf{rec}} \times \mathcal{P}_{\mathsf{rec}}$ is said to satisfy the *root condition* in R if the following condition holds:

- if $t_1 \xrightarrow{\alpha} t_1'$, then there exists a $t_2' \in \mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ such that $t_2 \xrightarrow{\alpha} t_2'$ and $R(t_1', t_2')$;
- if $t_2 \xrightarrow{\alpha} t_2'$, then there exists a $t_1' \in \mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ such that $t_1 \xrightarrow{\alpha} t_1'$ and $R(t_1', t_2')$.

Two terms $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ are *rooted branching bisimilar*, written $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$, if there exists a branching bisimulation R with $R(t_1, t_2)$ such that, for all terms $t_1', t_2' \in \mathcal{P}_{\mathsf{rec}}$ with $t_1 \xrightarrow{\sigma} t_1'$, $t_2 \xrightarrow{\sigma} t_2'$, and $R(t_1', t_2')$, the pair (t_1', t_2') satisfies the root condition in R. Rooted branching bisimilarity is also called *rooted branching bisimulation equivalence*.

Let R be a branching bisimulation such that $R(t_1, t_2)$ and, for all terms $t_1', t_2' \in \mathcal{P}_{\mathsf{rec}}$ with $t_1 \xrightarrow{\sigma} t_1'$, $t_2 \xrightarrow{\sigma} t_2'$, and $R(t_1', t_2')$, the pair (t_1', t_2') satisfies the root condition in R. Then we say that R is a branching bisimulation *witnessing* $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$.

If R is a branching bisimulation witnessing $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$, then not only the pair (t_1, t_2) must satisfy the root condition in R, but also all other pairs (t_1', t_2') for which $t_1 \xrightarrow{\sigma} t_1'$, $t_2 \xrightarrow{\sigma} t_2'$, and $R(t_1', t_2')$. Without this additional requirement (relative to branching bisimilarity as defined in [16]), branching bisimilarity is doomed not to be a congruence with respect to alternative composition, left merge and communication merge in the current setting (see Theorem 6 below).

4.2 Properties of Rooted Branching Bisimilarity

Branching bisimilarity and rooted branching bisimilarity are equivalence relations.

Theorem 5 (Equivalence). *The binary relations $\underline{\leftrightarrow}_{\mathsf{b}}$ and $\underline{\leftrightarrow}_{\mathsf{rb}}$ on $\mathcal{P}_{\mathsf{rec}}$ are equivalence relations.*

Proof. By Lemma 2.5 from [16], known as the stuttering lemma, $\underline{\leftrightarrow}_{\mathsf{b}}$ is an instance of the standard notion of branching bisimilarity. By Proposition 2.11 from [16], $\underline{\leftrightarrow}_{\mathsf{b}}$ is therefore an equivalence relation. The proofs of reflexivity and symmetry of $\underline{\leftrightarrow}_{\mathsf{rb}}$ go the same as the proofs of reflexivity and symmetry of $\underline{\leftrightarrow}_{\mathsf{b}}$. To prove that $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$ and $t' \underline{\leftrightarrow}_{\mathsf{rb}} t''$ implies $t \underline{\leftrightarrow}_{\mathsf{rb}} t''$, consider the relation composition of witnessing branching bisimulations R and R' for $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$ and $t' \underline{\leftrightarrow}_{\mathsf{rb}} t''$, respectively. Take arbitrary $t_1, t_2, t_3 \in \mathcal{P}_{\mathsf{rec}}$ such that $R(t_1, t_2)$ and $R'(t_2, t_3)$. Two cases have to be distinguished: (a) the case where (t_1, t_2) has to satisfy the root condition in R and (b) the case where (t_1, t_2) does not have to satisfy the root condition in R. If (t_1, t_2) has to satisfy the root condition in R, then (t_2, t_3) has to satisfy the root condition in R' as well. Case (a) follows immediately from the satisfaction of the the root condition in R and R' by (t_1, t_2)

and (t_2, t_3), respectively. The proof of case (b) goes the same as in the proof of Proposition 2.11 from [16] (for the details, look at the proof of Proposition 7 from [7]). □

Rooted branching bisimilarity is a congruence relation with respect to all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$.

Theorem 6 (Congruence). *The binary relation $\underline{\leftrightarrow}_{\mathsf{rb}}$ on $\mathcal{P}_{\mathsf{rec}}$ is a congruence relation with respect to all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$.*

Proof. In the proof, the auxiliary notions of weakly rooted branching bisimilarity and strong root condition are used. Two terms $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ are *weakly rooted branching bisimilar*, written $t_1 \underline{\leftrightarrow}_{\mathsf{wrb}} t_2$, if there exists a branching bisimulation R with $R(t_1, t_2)$ such that the pair (t_1, t_2) satisfies the root condition in R; if R is a binary relation on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$, then a pair $(t_1, t_2) \in \mathcal{P}_{\mathsf{rec}} \times \mathcal{P}_{\mathsf{rec}}$ is said to satisfy the *strong root condition* in R if, for all terms $t'_1, t'_2 \in \mathcal{P}_{\mathsf{rec}}$ with $t_1 \xrightarrow{\sigma} t'_1$, $t_2 \xrightarrow{\sigma} t'_2$, and $R(t'_1, t'_2)$, the pair (t'_1, t'_2) satisfies the root condition in R.

It follows directly from the above definitions that $\underline{\leftrightarrow}_{\mathsf{rb}}$ is a congruence with respect to all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ iff $\underline{\leftrightarrow}_{\mathsf{wrb}}$ is a congruence with respect to all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ and satisfaction of the strong root condition is preserved by all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$. It is not hard to see that satisfaction of the strong root condition is preserved by all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$.

The RBB safe congruence format from [12] is applicable to $\underline{\leftrightarrow}_{\mathsf{wrb}}$. Except for the rules for alternative composition, left merge, and communication merge concerning the relation $\xrightarrow{\sigma}$, all rules given in Table 5 satisfy the restrictions of the RBB safe format. However, the preservation of the strong root condition by alternative composition, left merge, and communication merge exclude the possibility that these rules are the cause of non-congruence. □

Below, the soundness of the axiom system of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ will be established.

In this paper, the following terminology will be used in soundness proofs: (a) an equation *eq* over a signature Σ is said to be *valid with respect to* a congruence relation $\simeq$ on the set of all closed terms over a signature Σ if, for each closed substitution instance $t = t'$ of *eq*, $t \simeq t'$ and (b) a conditional equation *ceq* over a signature Σ is said to be *valid with respect to* a congruence relation $\simeq$ on the set of all closed terms over a signature Σ if, for each closed substitution instance $\{t_i = t'_i \mid i \in I\} \Rightarrow t = t'$ of *ceq*, $t \simeq t'$ if $t_i \simeq t'_i$ for each $i \in I$.

Theorem 7 (Soundness). *For all terms $t, t' \in \mathcal{P}_{\mathsf{rec}}$, $t = t'$ is derivable from the axioms of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ only if $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$.*

Proof. Because $\underline{\leftrightarrow}_{\mathsf{rb}}$ is a congruence with respect to all operators from the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$, only the validity of each axiom of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ has to be proved. The proofs of the validity of the axioms of $\mathrm{ACP}_{\mathrm{drt}}$ with respect to $\underline{\leftrightarrow}$ given in the proof of Theorem 5.2.3.10 from [29] carry over to the setting with closed terms in which $\underline{\tau}$ may occur. Because $\underline{\leftrightarrow} \subseteq \underline{\leftrightarrow}_{\mathsf{rb}}$, validity with respect

to $\underline{\leftrightarrow}$ implies validity with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$. Therefore, it is sufficient to prove the validity of axioms TI1DR, TI2DR, TI3, TI4, DRTI, DRB1–DRB4, RDP, and RSP with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$.

Below, we write $csi(eq)$, where eq is an equation between two $\mathrm{ACP}^{\tau}_{\mathrm{drt}}+\mathrm{REC}$ terms, for the set of all closed substitution instances of eq. Moreover, we write R_{id} for the identity relation on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$.

For each axiom ax from the above-mentioned axioms, a branching bisimulation R_{ax} witnessing the validity of ax can be constructed as follows:

– if ax is an instance of one of the axiom schemas TI1DR, TI2DR, TI3, TI4, DRTI, RDP or axiom DRB1:

$$R_{ax} = \{(t,t') \mid t = t' \in csi(ax)\} \cup R_{id} \ ;$$

– if ax is an instance of axiom schema DRB2:

$$R_{ax} = \{(t,t') \mid t = t' \in csi(ax)\}$$
$$\cup \{(t,t') \mid t = t' \in csi(\underline{\tau} \cdot (\nu(x) + y) + \nu(x) = \nu(x) + y)\} \cup R_{id} \ ;$$

– if ax is an instance of axiom schema DRB3: similar;
– if ax is an instance of axiom schema DRB4:

$$R_{ax} = \{(t,t') \mid t = t' \in csi(ax)\}$$
$$\cup \{(t,t') \mid t = t' \in csi(\sigma(\underline{\tau} \cdot x) + \nu(y) = \sigma(x) + \nu(y))\}$$
$$\cup \{(t,t') \mid t = t' \in csi(\underline{\tau} \cdot x = x)\} \cup R_{id} \ ;$$

– if ax is an instance $\{X_i = t_i \mid i \in I\} \Rightarrow X_j = \langle X_j | \{X_i = t_i \mid i \in I\}\rangle$ $(j \in I)$ of RSP:

$$R_{ax}$$
$$= \{(\theta(X_j), \langle X_j | \{X_i = t_i \mid i \in I\}\rangle) \mid j \in I \wedge \theta \in \Theta \wedge \bigwedge_{i \in I} \theta(X_i) \underline{\leftrightarrow} \theta(t_i)\}$$
$$\cup R_{id} \ ,$$

where Θ is the set of all functions from $\mathcal{X}$ to $\mathcal{P}_{\mathsf{rec}}$ and $\theta(t)$, where $\theta \in \Theta$ and $t \in \mathcal{P}_{\mathsf{rec}}$, stands for t with, for all $X \in \mathcal{X}$, all occurrences of X replaced by $\theta(X)$.

For each equational axiom ax, it is straightforward to check that the constructed relation R_{ax} is a branching bisimulation witnessing, for each closed substitution instance $t = t'$ of ax, $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$. For each conditional equational axiom ax, i.e. for each instance of RSP, it is straightforward to check that the constructed relation R_{ax} is a branching bisimulation witnessing, for each closed substitution instance $\{t_i = t'_i \mid i \in I\} \Rightarrow t = t'$ of ax, $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$ if $t_i \underline{\leftrightarrow}_{\mathsf{rb}} t'_i$ for each $i \in I$. $\square$

The axiom system of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}+\mathrm{REC}$ is complete with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ in which only the constants and operators of $\mathrm{ACP}_{\mathrm{drt}}$ occur.

Theorem 8 (Completeness). *For all terms $t, t' \in \mathcal{P}_{\mathsf{rec}}$ that are closed terms of $\mathrm{ACP}_{\mathrm{drt}}$, $t = t'$ is derivable from the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ if $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$.*

Proof. This is Theorem 5.2.3.13 from [29]. $\qquad\qquad\qquad\qquad\qquad\qquad\square$

It is an open problem whether the axiom system of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ is complete with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ in which only the constants and operators of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ occur.

Even if the axiom system of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ is complete with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ in which only the constants and operators of $\mathrm{ACP}^{\tau}_{\mathrm{drt}}$ occur, it is not complete with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$. Consider, for example, the $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ term $\tau_{\{a,b\}}(\langle X | E \rangle)$, where $E = \{X = \underline{a} \cdot Y, Y = \underline{b} \cdot X + \underline{c}\}$. Then $\tau_{\{a,b\}}(\langle X | E \rangle) \underline{\leftrightarrow}_{\mathsf{rb}} \underline{\tau} \cdot \underline{c}$, but the equation $\tau_{\{a,b\}}(\langle X | E \rangle) = \underline{\tau} \cdot \underline{c}$ is not derivable from the axioms of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$. It is an open problem which additional axioms and/or restrictions on the form of guarded recursive specifications are needed to yield completeness with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$.

4.3 A Definition Referring to a Time-Stamped Semantics

The definition of the standard version of branching bisimilarity given below refers to a time-stamped structural operational semantics of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ that consists of:

– a binary *transition* relation $\_ \xrightarrow{a[n]} \_ \subseteq \mathcal{P}_{\mathsf{rec}} \times (\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\})$ for each $a \in \mathsf{A}_{\tau}$ and $n \in \mathbb{N}$;
– a unary *idling* relation $\_ \xmapsto{n} \subseteq \mathcal{P}_{\mathsf{rec}}$ for each $n \in \mathbb{N}$.

The relations from this structural operational semantics describe what the processes denoted by terms from $\mathcal{P}_{\mathsf{rec}}$ are capable of doing as follows:

– $t \xrightarrow{a[n]} t'$ indicates that the process denoted by t has the potential to make a transition to the process denoted by t' by performing action a in the nth-next time slice;
– $t \xmapsto{n}$ indicates that the process denoted by t is capable of idling till the nth-next time slice.

To define the time-stamped structural operational semantics of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$, there is the need to introduce the auxiliary unary *one-time-slice shift* operator $\overline{\sigma}$. This operator can be explained as follows: $\overline{\sigma}(t)$, where t is a closed $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ term, denotes the process that remains after the process denoted by t has idled till the next time slice. The defining axioms for the one-time-slice shift operator are given in Table 6. In this table, a stands for an arbitrary member of $\mathsf{A}_{\tau\delta}$.

We use the notation $\overline{\sigma}^n(t)$, where $n \in \mathbb{N}$, for the n-fold application of $\overline{\sigma}$ to t, i.e. $\overline{\sigma}^0(t) = t$ and $\overline{\sigma}^{n+1}(t) = \overline{\sigma}(\overline{\sigma}^n(t))$.

The relations from the time-stamped structural operational semantics of $\mathrm{ACP}^{\tau}_{\mathrm{drt}} + \mathrm{REC}$ are the smallest relations satisfying the rules given in Table 7.

24

Table 6. Defining axioms of the one-time-slice shift operator

$\overline{\sigma}(\underline{a}) = \underline{\delta}$	DRSH1
$\overline{\sigma}(x + y) = \overline{\sigma}(x) + \overline{\sigma}(y)$	DRSH2
$\overline{\sigma}(x \cdot y) = \overline{\sigma}(x) \cdot y$	DRSH3
$\overline{\sigma}(\sigma(x)) = x$	DRSH4

In this table, a, b, and c stand for arbitrary actions from A_τ, n stands for an arbitrary natural number from $\mathbb{N}$, H and I stand for arbitrary subsets of A, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary $\text{ACP}_{\text{drt}}^\tau +$ REC term, and E stands for an arbitrary guarded recursive specification over $\text{ACP}_{\text{drt}}^\tau + \text{REC}$.

In Figure 3, the processes that are denoted by the closed terms $\sigma(\underline{a}) + \sigma(\underline{b})$ and $\underline{\tau} \cdot \sigma(\underline{a}) + \sigma(\underline{b})$ according to the time-stamped operational semantics are presented graphically as far as the transitions are concerned. Noteworthy is that, different from the processes denoted by these terms according to the time-stamped operational semantics (see Figure 2), both processes now have a choice.
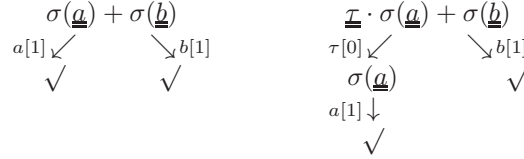

Fig. 3. Graphical time-stamped presentation of two processes

A *ts-branching bisimulation* is a binary relation R on $\mathcal{P}_{\text{rec}} \cup \{\sqrt{}\}$ such that, for all $t_1, t_2 \in \mathcal{P}_{\text{rec}}$ with $R(t_1, t_2)$, the following transfer conditions hold:

- if $t_1 \xrightarrow{a[n]} t_1'$, then
  - either $a \equiv \tau$ and $R(t_1', \overline{\sigma}^n(t_2))$
  - or, for some $m \in \mathbb{N}$, there exist $t_0^*, \ldots, t_m^* \in \mathcal{P}_{\text{rec}}$, $t_2' \in \mathcal{P}_{\text{rec}} \cup \{\sqrt{}\}$, and $n_0, \ldots, n_m \in \mathbb{N}$ with $t_0^* \equiv t_2$ and $\sum_{i=0}^m n_i = n$ such that
    * for all $k \in \mathbb{N}$ with $k < m$,
      $$t_k^* \xrightarrow{\tau[n_k]} t_{k+1}^* \text{ and } R(\overline{\sigma}^{\sum_{i=0}^k n_i}(t_1), t_{k+1}^*);$$
    * $t_m^* \xrightarrow{a[n_m]} t_2'$, and $R(t_1', t_2')$;
- if $t_2 \xrightarrow{a[n]} t_2'$, then
  - either $a \equiv \tau$ and if $R(\overline{\sigma}^n(t_1), t_2')$
  - or, for some $m \in \mathbb{N}$, there exist $t_0^*, \ldots, t_m^* \in \mathcal{P}_{\text{rec}}$, $t_1' \in \mathcal{P}_{\text{rec}} \cup \{\sqrt{}\}$, and $n_0, \ldots, n_m \in \mathbb{N}$ with $t_0^* \equiv t_1$ and $\sum_{i=0}^m n_i = n$ such that
    * for all $k \in \mathbb{N}$ with $k < m$,
      $$t_k^* \xrightarrow{\tau[n_k]} t_{k+1}^* \text{ and } R(t_{k+1}^*, \overline{\sigma}^{\sum_{i=0}^k n_i}(t_2));$$

Table 7. Transition rules for $\text{ACP}^{\tau}_{\text{drt}}+\text{REC}$

$$\overline{x \stackrel{0}{\longmapsto}}$$

$$\overline{\underline{a} \stackrel{a[0]}{\longrightarrow} \surd}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x'}{x+y \stackrel{a[n]}{\longrightarrow} x'} \qquad \frac{y \stackrel{a[n]}{\longrightarrow} y'}{x+y \stackrel{a[n]}{\longrightarrow} y'} \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd}{x+y \stackrel{a[n]}{\longrightarrow} \surd} \qquad \frac{y \stackrel{a[n]}{\longrightarrow} \surd}{x+y \stackrel{a[n]}{\longrightarrow} \surd} \qquad \frac{x \stackrel{n}{\longmapsto}}{x+y \stackrel{n}{\longmapsto}} \qquad \frac{y \stackrel{n}{\longmapsto}}{x+y \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x'}{x \cdot y \stackrel{a[n]}{\longrightarrow} x' \cdot y} \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd}{x \cdot y \stackrel{a[n]}{\longrightarrow} y} \qquad \frac{x \stackrel{n}{\longmapsto}}{x \cdot y \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x'}{\sigma(x) \stackrel{a[n+1]}{\longrightarrow} x'} \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd}{\sigma(x) \stackrel{a[n+1]}{\longrightarrow} \surd} \qquad \frac{x \stackrel{n}{\longmapsto}}{\sigma(x) \stackrel{n+1}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x', \ y \stackrel{n}{\longmapsto}}{x \parallel y \stackrel{a[n]}{\longrightarrow} x' \parallel \overline{\sigma}^n(y)} \qquad \frac{x \stackrel{n}{\longmapsto}, \ y \stackrel{a[n]}{\longrightarrow} y'}{x \parallel y \stackrel{a[n]}{\longrightarrow} \overline{\sigma}^n(x) \parallel y'} \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd, \ y \stackrel{n}{\longmapsto}}{x \parallel y \stackrel{a[n]}{\longrightarrow} \overline{\sigma}^n(y)} \qquad \frac{x \stackrel{n}{\longmapsto}, \ y \stackrel{a[n]}{\longrightarrow} \surd}{x \parallel y \stackrel{a[n]}{\longrightarrow} \overline{\sigma}^n(x)}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x', \ y \stackrel{b[n]}{\longrightarrow} y'}{x \parallel y \stackrel{c[n]}{\longrightarrow} x' \parallel y'} \gamma(a,b)=c \qquad \frac{x \stackrel{a[n]}{\longrightarrow} x', \ y \stackrel{b[n]}{\longrightarrow} \surd}{x \parallel y \stackrel{c[n]}{\longrightarrow} x'} \gamma(a,b)=c$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} \surd, \ y \stackrel{b[n]}{\longrightarrow} y'}{x \parallel y \stackrel{c[n]}{\longrightarrow} y'} \gamma(a,b)=c \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd, \ y \stackrel{b[n]}{\longrightarrow} \surd}{x \parallel y \stackrel{c[n]}{\longrightarrow} \surd} \gamma(a,b)=c \qquad \frac{x \stackrel{n}{\longmapsto}, \ y \stackrel{n}{\longmapsto}}{x \parallel y \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x', \ y \stackrel{n}{\longmapsto}}{x \,\underline{\parallel}\, y \stackrel{a[n]}{\longrightarrow} x' \parallel \overline{\sigma}^n(y)} \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd, \ y \stackrel{n}{\longmapsto}}{x \,\underline{\parallel}\, y \stackrel{a[n]}{\longrightarrow} \overline{\sigma}^n(y)} \qquad \frac{x \stackrel{n}{\longmapsto}, \ y \stackrel{n}{\longmapsto}}{x \,\underline{\parallel}\, y \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x', \ y \stackrel{b[n]}{\longrightarrow} y'}{x \mid y \stackrel{c[n]}{\longrightarrow} x' \parallel y'} \gamma(a,b)=c \qquad \frac{x \stackrel{a[n]}{\longrightarrow} x', \ y \stackrel{b[n]}{\longrightarrow} \surd}{x \mid y \stackrel{c[n]}{\longrightarrow} x'} \gamma(a,b)=c$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} \surd, \ y \stackrel{b[n]}{\longrightarrow} y'}{x \mid y \stackrel{c[n]}{\longrightarrow} y'} \gamma(a,b)=c \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd, \ y \stackrel{b[n]}{\longrightarrow} \surd}{x \mid y \stackrel{c[n]}{\longrightarrow} \surd} \gamma(a,b)=c \qquad \frac{x \stackrel{n}{\longmapsto}, \ y \stackrel{n}{\longmapsto}}{x \mid y \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x'}{\partial_H(x) \stackrel{a[n]}{\longrightarrow} \partial_H(x')} a \notin H \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd}{\partial_H(x) \stackrel{a[n]}{\longrightarrow} \surd} a \notin H \qquad \frac{x \stackrel{n}{\longmapsto}}{\partial_H(x) \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x'}{\tau_I(x) \stackrel{a[n]}{\longrightarrow} \tau_I(x')} a \notin I \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd}{\tau_I(x) \stackrel{a[n]}{\longrightarrow} \surd} a \notin I$$

$$\frac{x \stackrel{a[n]}{\longrightarrow} x'}{\tau_I(x) \stackrel{\tau[n]}{\longrightarrow} \tau_I(x')} a \in I \qquad \frac{x \stackrel{a[n]}{\longrightarrow} \surd}{\tau_I(x) \stackrel{\tau[n]}{\longrightarrow} \surd} a \in I \qquad \frac{x \stackrel{n}{\longmapsto}}{\tau_I(x) \stackrel{n}{\longmapsto}}$$

$$\frac{x \stackrel{a[0]}{\longrightarrow} x'}{\nu(x) \stackrel{a[0]}{\longrightarrow} x'} \qquad \frac{x \stackrel{a[0]}{\longrightarrow} \surd}{\nu(x) \stackrel{a[0]}{\longrightarrow} \surd}$$

$$\frac{\langle t|E \rangle \stackrel{a[n]}{\longrightarrow} x'}{\langle X|E \rangle \stackrel{a[n]}{\longrightarrow} x'} X=t \in E \qquad \frac{\langle t|E \rangle \stackrel{a[n]}{\longrightarrow} \surd}{\langle X|E \rangle \stackrel{a[n]}{\longrightarrow} \surd} X=t \in E \qquad \frac{\langle t|E \rangle \stackrel{n}{\longmapsto}}{\langle X|E \rangle \stackrel{n}{\longmapsto}} X=t \in E$$

$$\frac{x \stackrel{a[n+1]}{\longrightarrow} x'}{\overline{\sigma}(x) \stackrel{a[n]}{\longrightarrow} x'} \qquad \frac{x \stackrel{a[n+1]}{\longrightarrow} \surd}{\overline{\sigma}(x) \stackrel{a[n]}{\longrightarrow} \surd} \qquad \frac{x \stackrel{n+1}{\longmapsto}}{\overline{\sigma}(x) \stackrel{n}{\longmapsto}}$$

$$* \ t_m^* \xrightarrow{a[n_m]} t_1', \text{ and } R(t_1', t_2');$$
$$- \ t_1 \xmapsto{n} \text{ iff } t_2 \xmapsto{n}.$$

Two terms $t_1, t_2 \in \mathcal{P}_{\text{rec}}$ are *ts-branching bisimilar*, written $t_1 \underline{\leftrightarrow}_b' t_2$, if there exists a ts-branching bisimulation R with $R(t_1, t_2)$.

If R is a binary relation on $\mathcal{P}_{\text{rec}} \cup \{\sqrt\}$, then a pair $(t_1, t_2) \in \mathcal{P}_{\text{rec}} \times \mathcal{P}_{\text{rec}}$ is said to satisfy the *ts-root condition* in R if the following condition holds:

- if $t_1 \xrightarrow{a[n]} t_1'$, then there exists a $t_2' \in \mathcal{P}_{\text{rec}} \cup \{\sqrt\}$ such that $t_2 \xrightarrow{a[n]} t_2'$ and $R(t_1', t_2')$;
- if $t_2 \xrightarrow{a[n]} t_2'$, then there exists a $t_1' \in \mathcal{P}_{\text{rec}} \cup \{\sqrt\}$ such that $t_1 \xrightarrow{a[n]} t_1'$ and $R(t_1', t_2')$.

Two terms $t_1, t_2 \in \mathcal{P}_{\text{rec}}$ are *rooted ts-branching bisimilar*, written $t_1 \underline{\leftrightarrow}_{rb}' t_2$, if there exists a ts-branching bisimulation R such that $R(t_1, t_2)$ and (t_1, t_2) satisfies the ts-root condition in R.

Rooted ts-branching bisimilarity is an equivalence relation.

Theorem 9 (Equivalence). *The binary relation $\underline{\leftrightarrow}_{rb}'$ on $\mathcal{P}_{\text{rec}}$ is an equivalence relation.*

Proof. The proofs of reflexivity and symmetry of $\underline{\leftrightarrow}_{rb}'$ go the same as the proofs of reflexivity and symmetry of $\underline{\leftrightarrow}_{rb}$. To prove that $t \underline{\leftrightarrow}_{rb}' t'$ and $t' \underline{\leftrightarrow}_{rb}' t''$ implies $t \underline{\leftrightarrow}_{rb}' t''$, consider the relation composition of witnessing branching bisimulations R and R' for $t \underline{\leftrightarrow}_{rb}' t'$ and $t' \underline{\leftrightarrow}_{rb}' t''$, respectively. Take arbitrary $t_1, t_2, t_3 \in \mathcal{P}_{\text{rec}}$ such that $R(t_1, t_2)$ and $R'(t_2, t_3)$. Two cases have to be distinguished: (a) the case where (t_1, t_2) has to satisfy the root condition in R and (b) the case where (t_1, t_2) does not have to satisfy the root condition in R. If (t_1, t_2) has to satisfy the root condition in R, then (t_2, t_3) has to satisfy the root condition in R' as well. Case (a) follows immediately from the satisfaction of the the root condition in R and R' by (t_1, t_2) and (t_2, t_3), respectively. The proof of case (b) goes along the same lines as the proof of Proposition 2.11 from [16] given in [7], but taking into account the number of time slices involved in sequences of silent steps. Such a similar proof is possible because the replacement of the conditions

- for all $k \in \mathbb{N}$ with $k < m$,
  $$t_k^* \xrightarrow{\tau[n_k]} t_{k+1}^* \text{ and } R(\overline{\sigma}^{\sum_{i=0}^k n_i}(t_1), t_{k+1}^*);$$
- for all $k \in \mathbb{N}$ with $k < m$,
  $$t_k^* \xrightarrow{\tau[n_k]} t_{k+1}^* \text{ and } R(t_{k+1}^*, \overline{\sigma}^{\sum_{i=0}^k n_i}(t_2))$$

in the definition of ts-branching bisimulation by

- for all $k \in \mathbb{N}$ with $k < m$,
  $$t_k^* \xrightarrow{\tau[n_k]} t_{k+1}^* \text{ and if } k+1 = m \text{ then } R(\overline{\sigma}^{\sum_{i=0}^k n_i}(t_1), t_{k+1}^*);$$
- for all $k \in \mathbb{N}$ with $k < m$,
  $$t_k^* \xrightarrow{\tau[n_k]} t_{k+1}^* \text{ and if } k+1 = m \text{ then } R(t_{k+1}^*, \overline{\sigma}^{\sum_{i=0}^k n_i}(t_2))$$

yields a bisimilarity relation that coincides with ts-branching bisimilarity as defined above. This is due to a stuttering property that can be proved in the same way as Lemma 2.5 from [16], but taking into account the number of time slices involved in sequences of silent steps. □

The fact that rooted ts-branching bisimilarity is an equivalence relation also follows from Theorem 10 (see below). The proof of Theorem 9 given above shows that this fact does not depend on the extension of the transition relations and idling relations involved.

In Figure 4, the process that is denoted by the closed term $\sigma(\sigma(\underline{a})) + \sigma(\underline{b})$ is presented graphically as far as the transitions are concerned, both according to the two-phase operational semantics and according to the time-stamped operational semantics. In this example, the two presentations convey in some sense the same behaviour. This raises the question how the two-phase structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ and the time-stamped structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ are exactly related.
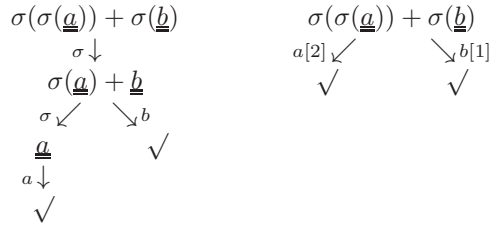
Fig. 4. Graphical two-phase and time-stamped presentation of one process

Below, we write $t \xrightarrow{\sigma^{n+1}} t'$, where $t, t' \in \mathcal{P}_{\mathsf{rec}}$ and $n \in \mathbb{N}$, to indicate that there exist $t_0, \ldots, t_{n+1} \in \mathcal{P}_{\mathsf{rec}}$ with $t_0 \equiv t$ and $t_{n+1} \equiv t'$ such that, for all $i \in \mathbb{N}$ with $i \leq n$, $t_i \xrightarrow{\sigma} t_{i+1}$.

The following proposition relates the two-phase structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ and the time-stamped structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$.

Proposition 1. *For all $t \in \mathcal{P}_{\mathsf{rec}}$, $t' \in \mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$, $a \in \mathsf{A}_\tau$, and $n \in \mathbb{N}$, the following holds:*

(1) $t \xrightarrow{a[0]} t'$ iff $t \xrightarrow{a} t'$;

(2) $t \xrightarrow{a[n+1]} t'$ iff there exists a $t'' \in \mathcal{P}_{\mathsf{rec}}$ such that $t \xrightarrow{\sigma^{n+1}} t''$ and $t'' \xrightarrow{a} t'$;

(3) $t \xrightarrow{n+1}$ iff there exists a $t'' \in \mathcal{P}_{\mathsf{rec}}$ such that $t \xrightarrow{\sigma^{n+1}} t''$.

Proof. Both the 'if' part and the 'only if' part of (1)–(3) are easily proved by induction on the structure of t. □

Rooted branching bisimilarity and rooted ts-branching bisimilarity coincide.

Theorem 10 (Equivalent definitions). *For all terms $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$, $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$ iff $t_1 \underline{\leftrightarrow}'_{\mathsf{rb}} t_2$.*

Proof. The 'only if' part. Assume that $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ and $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$. Then there is a branching bisimulation R witnessing $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$. Construct a binary relation R' on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ from R as follows:

$$R' = R \setminus \bigcup_{n \in \mathbb{N}} \{(t_1'', t_2'') \mid \exists t_1', t_2' \in \mathcal{P}_{\mathsf{rec}} \bullet R(t_1', t_2') \wedge t_1' \xrightarrow{\sigma^{n+1}} t_1'' \wedge t_2' \xrightarrow{\sigma^{n+1}} t_2''\} \;.$$

Using Proposition 1, it is straightforward to check that the constructed relation R' is a ts-branching bisimulation witnessing $t_1 \underline{\leftrightarrow}'_{\mathsf{rb}} t_2$.

The 'if' part. Assume that $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ and $t_1 \underline{\leftrightarrow}'_{\mathsf{rb}} t_2$. Then there is a ts-branching bisimulation S witnessing $t_1 \underline{\leftrightarrow}'_{\mathsf{rb}} t_2$. Construct a binary relation S' on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ from S as follows:

$$S' = S \cup \bigcup_{n \in \mathbb{N}} \{(t_1'', t_2'') \mid \exists t_1', t_2' \in \mathcal{P}_{\mathsf{rec}} \bullet S(t_1', t_2') \wedge t_1' \xrightarrow{\sigma^{n+1}} t_1'' \wedge t_2' \xrightarrow{\sigma^{n+1}} t_2''\} \;.$$

Using Proposition 1, it is straightforward to check that the constructed relation S' is a branching bisimulation witnessing $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$. $\qquad\square$

It may come across as surprising that not too much is removed from R in the proof of the 'only if' part of Theorem 10. However, that not too much is removed is self-evident when realising that the following is a direct corollary of Proposition 1: there exist $s'', t'' \in \mathcal{P}_{\mathsf{rec}}$ such that $t \xrightarrow{\sigma^{m+1}} s''$, $s'' \xrightarrow{\sigma^{n-m}} t''$, $t'' \xrightarrow{a} t'$, and $s'' \xrightarrow{b} s'$ iff $t \xrightarrow{a[n+1]} t'$, $t \xrightarrow{b[m+1]} s'$, and $m < n$. Figure 4 above gives an example where $n = 2$ and $m = 1$.

The following soundness result is a corollary of Theorems 7 and 10.

Corollary 1. *For all terms $t, t' \in \mathcal{P}_{\mathsf{rec}}$, $t = t'$ is derivable from the axioms of $\mathrm{ACP}_{\mathrm{drt}}^{\tau} + \mathrm{REC}$ only if $t \underline{\leftrightarrow}'_{\mathsf{rb}} t'$.*

Basic terms as defined in Section 2.1 reflect the two-phase semantics of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$. A variant of basic terms, called ts-basic terms, that reflect the time-stamped semantics of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ can also be defined. The set $\mathcal{B}'$ of *ts-basic terms* over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ is the smallest set satisfying:

- if $a \in \mathsf{A}_{\tau\delta}$ and $n \in \mathbb{N}$, then $\sigma^n(\underline{a}) \in \mathcal{B}'$;
- if $a \in \mathsf{A}_{\tau}$, $n \in \mathbb{N}$, and $t \in \mathcal{B}'$, then $\sigma^n(\underline{a}) \cdot t \in \mathcal{B}'$;
- if $t, t' \in \mathcal{B}'$, then $t + t' \in \mathcal{B}'$.

Modulo axioms A1 and A2, each ts-basic term from $\mathcal{B}'$ is of the following form:

$$\sum_{1 \leq i \leq n} \sigma^{n_i}(\underline{a_i}) \cdot t_i + \sum_{1 \leq j \leq m} \sigma^{m_j}(\underline{b_j}) \;,$$

where $n, m \in \mathbb{N}$, $a_i \in \mathsf{A}_{\tau}$, $n_i \in \mathbb{N}$, and $t_i \in \mathcal{B}'$ for $1 \leq i \leq n$, and $b_j \in \mathsf{A}_{\tau}$ and $m_i \in \mathbb{N}$ for $1 \leq j \leq m$.

The following proposition states that each closed $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$ term can be reduced to a ts-basic term.

Proposition 2. *For each closed term t of $\mathrm{ACP}^\tau_{\mathrm{drt}}$, there exists a ts-basic term $t' \in \mathcal{B}'$ such that $t = t'$ is derivable from the axioms of $\mathrm{ACP}^\tau_{\mathrm{drt}}$.*

Proof. By Theorem 1, it is sufficient to prove the following:

> *for each basic term $t \in \mathcal{B}$, there exists a ts-basic term $t' \in \mathcal{B}'$ such that $t = t'$ is derivable from the axioms of $\mathrm{ACP}^\tau_{\mathrm{drt}}$.*

This is easily proved by induction on the structure of t. $\qquad\qquad\square$

5   A Coarser Version of Branching Bisimilarity

The axioms of $\mathrm{ACP}^\tau_{\mathrm{drt}}$ concerning silent steps (DRB1–DRB4) are based on the standard notion of branching bisimilarity for processes with discrete relative timing, i.e. the one first introduced in [2]. The simple idea behind this notion of branching bisimilarity is that an internal action can be forgotten wherever no options are lost by performing it. The experience with analyzing the PAR protocol (and other protocols) triggered a quest for a version of branching bisimilarity for processes with discrete relative timing that is coarser than the standard version. In this section, a variant of the standard version will be defined that is coarser than the standard version by treating an internal action always as redundant if it is followed by a process that is only capable of idling till the next time slice.

In [5], a first attempt has been made to define the same variant. However, the attempt was unsatisfactory. The definition of the variant in that paper was in fact complicated to such an extent that it led to errors in the paper that were not spotted at the time. Like the definition of the standard version, the definition concerned refers to a two-phase operational semantics of a version of ACP with abstraction for processes with discrete relative timing. In this section, a definition of the variant will be given that refers to the time-stamped operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}$ presented in Section 4.3. It turns out that taking a time-stamped operational semantics as the basis results in a definition that is far less complicated than the one from [5].

5.1   Dormancy-Aware Branching Bisimilarity

In this section, a version of branching bisimilarity for processes with discrete relative timing is introduced that is coarser than the one introduced in Section 4. The version introduced is coarser as a result of treating an internal action always as redundant if it is followed by a process that is only capable of idling till the next time slice.

With regard to the motivation for this coarser version, consider (a) a process p with the option of idling till the next time slice and then to behave as a process q and (b) a process p' that behaves as p except that in the above-mentioned option the idling till the next time slice is preceded by performing an internal action in the current time slice. The capabilities of p change by idling till the next time slice and the capabilities of p' change by performing an internal action

in the current time slice. However, since it requires unlikely means to perceive differences between spending time in performing an action that is considered to be unobservable and spending time in not performing any action, the latter change is considered to be observable only after the subsequent idling till the next time slice. In either case, the capabilities after the idling till the next time slice are the same. In other words, it is hard to imagine a realistic experiment that can distinguish between the processes p and p'.

To enforce that the passage of time cannot introduce non-determinism, the two-phase structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ is such that, for all $t, t' \in \mathcal{P}_{\mathsf{rec}}$, $\sigma(t) + \sigma(t')$ can only make the transition $\sigma(t) + \sigma(t') \xrightarrow{\sigma} t + t'$. This way to enforce that the passage of time cannot introduce non-determinism causes a problem with the treatment of an internal action as redundant if it is followed by a process that is only capable of idling till the next time slice: even if a version of branching bisimilarity allows of forgetting about the internal action involved, $\tau \cdot \sigma(t) + \sigma(t')$ can still make two transitions (see also Figure 2). This problematic difference in the number of possible transitions does not occur with the time-stamped structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ (see also Figure 3). Therefore, the definition of the coarser version of branching bisimilarity will refer to the time-stamped structural operational semantics of $\mathrm{ACP}^\tau_{\mathrm{drt}}+\mathrm{REC}$ instead.

If a process is capable of performing an action in the current time slice, then it is regarded as being active. We inductively define a unary *activeness* relation $\mathcal{A}$ on $\mathcal{P}_{\mathsf{rec}} \cup \{\surd\}$ as follows:

- if $t \xrightarrow{a[0]} t'$ for some $a \in \mathsf{A}$ and $t' \in \mathcal{P}_{\mathsf{rec}}$, then $\mathcal{A}(t)$;
- if $t \xrightarrow{a[0]} \surd$ for some $a \in \mathsf{A}_\tau$, then $\mathcal{A}(t)$;
- if $t \xrightarrow{\tau[0]} t'$ and $\mathcal{A}(t')$, then $\mathcal{A}(t)$.

If not $\mathcal{A}(t)$, then the process denoted by t is said to be dormant.

A *dormancy-aware* branching bisimulation is a binary relation R on $\mathcal{P}_{\mathsf{rec}} \cup \{\surd\}$ such that, for all $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ with $R(t_1, t_2)$, the following transfer conditions hold:

- if $t_1 \xrightarrow{a[n]} t'_1$, then
  - either $a \equiv \tau$ and if $\mathcal{A}(t'_1)$ then $R(t'_1, \overline{\sigma}^n(t_2))$
  - or, for some $m \in \mathbb{N}$, there exist $t^*_0, \ldots, t^*_m \in \mathcal{P}_{\mathsf{rec}}$, $t'_2 \in \mathcal{P}_{\mathsf{rec}} \cup \{\surd\}$, and $n_0, \ldots, n_m \in \mathbb{N}$ with $t^*_0 \equiv t_2$ and $\sum_{i=0}^m n_i = n$ such that
    * for all $k \in \mathbb{N}$ with $k < m$,
      $$t^*_k \xrightarrow{\tau[n_k]} t^*_{k+1} \text{ and if } \mathcal{A}(t^*_{k+1}) \text{ then } R(\overline{\sigma}^{\sum_{i=0}^k n_i}(t_1), t^*_{k+1});$$
    * $t^*_m \xrightarrow{a[n_m]} t'_2$, and $R(t'_1, t'_2)$;
- if $t_2 \xrightarrow{a[n]} t'_2$, then
  - either $a \equiv \tau$ and if $\mathcal{A}(t'_2)$ then $R(\overline{\sigma}^n(t_1), t'_2)$
  - or, for some $m \in \mathbb{N}$, there exist $t^*_0, \ldots, t^*_m \in \mathcal{P}_{\mathsf{rec}}$, $t'_1 \in \mathcal{P}_{\mathsf{rec}} \cup \{\surd\}$, and $n_0, \ldots, n_m \in \mathbb{N}$ with $t^*_0 \equiv t_1$ and $\sum_{i=0}^m n_i = n$ such that
    * for all $k \in \mathbb{N}$ with $k < m$,
      $$t^*_k \xrightarrow{\tau[n_k]} t^*_{k+1} \text{ and if } \mathcal{A}(t^*_{k+1}) \text{ then } R(t^*_{k+1}, \overline{\sigma}^{\sum_{i=0}^k n_i}(t_2));$$

Table 8. Additional axiom for dormancy-aware branching bisimilarity

$$\underline{a} \cdot (\sigma^n(\underline{\tau} \cdot \sigma(x)) + y) = \underline{a} \cdot (\sigma^n(\sigma(x)) + y) \quad \text{DRB5}$$

$$* \ t_m^* \xrightarrow{a[n_m]} t_1', \text{ and } R(t_1', t_2');$$
$$- \ t_1 \stackrel{n}{\longmapsto} \text{ iff } t_2 \stackrel{n}{\longmapsto}.$$

The two processes presented graphically in Figure 3 are a simple example of processes that are dormancy-aware branching bisimilar but not ts-branching bisimilar.

Two terms $t_1, t_2 \in \mathcal{P}_{\mathsf{rec}}$ are rooted *dormancy-aware* branching bisimilar, written $t_1 \underline{\leftrightarrow}_{\mathsf{rb}} t_2$, if there exists a dormancy-aware branching bisimulation R such that $R(t_1, t_2)$ and (t_1, t_2) satisfies the ts-root condition in R.

Adoption of rooted dormancy-aware branching bisimilarity leads to the addition of the axiom given in Table 8 to the axiom system of $\mathrm{BPA}_\tau^{\mathrm{drt}} + \mathrm{REC}$. In this table, a stands for an arbitrary actions from A_τ and n stands for an arbitrary natural number from $\mathbb{N}$.

Axiom DRB5 is the axiom that characterizes the additional identifications made by rooted dormancy-aware branching bisimilarity. In Section 5.2, we will come back to the point that axiom DRB5 is valid with respect to rooted dormancy-aware branching bisimilarity.

5.2 Properties of Rooted Dormancy-Aware Branching Bisimilarity

Rooted dormancy-aware branching bisimilarity is coarser than rooted branching bisimilarity.

Theorem 11 (Inclusion). *Let t and t' be closed terms of* $\mathrm{ACP}_{\mathrm{drt}}^\tau + \mathrm{REC}$. *Then* $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$ *only if* $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$.

Proof. This follows directly from the definitions of $\underline{\leftrightarrow}_{\mathsf{rb}}$ and $\underline{\leftrightarrow}_{\mathsf{rb}}$. □

Rooted dormancy-aware branching bisimilarity is an equivalence relation.

Theorem 12 (Equivalence). *The relation $\underline{\leftrightarrow}_{\mathsf{rb}}$ is an equivalence relation.*

Proof. Define *generalized silent step* relations $\_ \xrightarrow{\tau[n]}\!\!\!\!\twoheadrightarrow \_ \subseteq \mathcal{P}_{\mathsf{rec}} \times (\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\})$ for $n \in \mathbb{N}$ as the smallest relations satisfying:

- if $t \xrightarrow{\tau[n]} t'$ and $\mathcal{A}(t')$, then $t \xrightarrow{\tau[n]}\!\!\!\!\twoheadrightarrow t'$;
- if $t \xrightarrow{\tau[n]} t'$, $t' \xrightarrow{\tau[n']}\!\!\!\!\twoheadrightarrow t''$, and not $\mathcal{A}(t')$, then $t \xrightarrow{\tau[n+n']}\!\!\!\!\twoheadrightarrow t''$.

Reformulate the definition of rooted dormancy-aware branching bisimilarity that refers to the time-stamped operational semantics defined in Section 4.3 using the generalized silent step relations. Then the reformulated definition is also a definition of rooted ts-branching bisimilarity that refers to the adapted time-stamped operational semantics of $\mathrm{ACP}_{\mathrm{drt}}^\tau + \mathrm{REC}$ that consists of:

32

– the binary transition relation $\_ \xrightarrow{a[n]} \_ \subseteq \mathcal{P}_{\mathsf{rec}} \times (\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\})$ defined in Section 4.3 for each $a \in \mathsf{A}$ and $n \in \mathbb{N}$;
– the binary generalized silent step relation $\_ \xrightarrow{\tau[n]} \_ \subseteq \mathcal{P}_{\mathsf{rec}} \times (\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\})$ defined above for each $n \in \mathbb{N}$;
– the unary idling relation $\_ \xmapsto{n} \subseteq \mathcal{P}_{\mathsf{rec}}$ defined in Section 4.3 for each $n \in \mathbb{N}$.

The proof of Theorem 9 given in Section 4.3 shows that the fact that rooted ts-branching bisimilarity is an equivalence relation does not depend on the extension of the transition relations and idling relations involved. From this and the reformulated definition, it follows immediately that rooted dormancy-aware branching bisimilarity is an equivalence relation. □

Rooted dormancy-aware branching bisimilarity is not a congruence with respect to the operators $\|$, $\lfloor$, and $|$. For example, $\underline{a} \cdot \sigma(\underline{b}) \underleftrightarrow{}_{\mathsf{rb}} \underline{a} \cdot \underline{\tau} \cdot \sigma(\underline{b})$ and $\underline{c} \underleftrightarrow{}_{\mathsf{rb}} \underline{c}$, but $\underline{a} \cdot \sigma(\underline{b}) \| \underline{c} \underleftrightarrow{\not}_{\mathsf{rb}} \underline{a} \cdot \underline{\tau} \cdot \sigma(\underline{b}) \| \underline{c}$ ($\underline{c}$ may be performed after $\underline{\tau}$). However, rooted dormancy-aware branching bisimilarity is preserved by all operators of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+$ REC.

Theorem 13 (Congruence). *The relation* $\underleftrightarrow{}_{\mathsf{rb}}$ *is a congruence relation with respect to all operators from the signature of* $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$.

Proof. Let $t_1, t_1', t_2, t_2' \in \mathcal{P}_{\mathsf{rec}}$ be such that $t_1 \underleftrightarrow{}_{\mathsf{rb}} t_1'$ and $t_2 \underleftrightarrow{}_{\mathsf{rb}} t_2'$, and let R_1 and R_2 be dormancy-aware branching bisimulations witnessing $t_1 \underleftrightarrow{}_{\mathsf{rb}} t_1'$ and $t_2 \underleftrightarrow{}_{\mathsf{rb}} t_2'$, respectively. Define, for each operator $\diamond$ of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$, a binary relation $R_\diamond$ on $\mathcal{P}_{\mathsf{rec}} \cup \{\sqrt{}\}$ as follows:

$$R_+ = \{(t_1 + t_2, t_1' + t_2')\} \cup R_1 \cup R_2 \ ,$$
$$R_. = \{(s_1 \cdot t_2, s_1' \cdot t_2') \mid R_1(s_1, s_1')\} \cup R_2 \ ,$$
$$R_\sigma = \{(\sigma(t_1), \sigma(t_1'))\} \cup R_1 \ ,$$
$$R_\nu = \{(\nu(t_1), \nu(t_1'))\} \cup R_1 \ .$$

It is a routine matter to check that these relations satisfy the transfer and root conditions required to be dormancy-aware branching bisimulations witnessing $t_1 + t_2 \underleftrightarrow{}_{\mathsf{rb}} t_1' + t_2'$, $t_1 \cdot t_2 \underleftrightarrow{}_{\mathsf{rb}} t_1' \cdot t_2'$, $\sigma(t_1) \underleftrightarrow{}_{\mathsf{rb}} \sigma(t_1')$, and $\nu(t_1) \underleftrightarrow{}_{\mathsf{rb}} \nu(t_1')$, respectively. □

The axiom system of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ extended with axiom DRB5 is sound with respect to $\underleftrightarrow{}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ in which only constants and operators of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ occur.

Theorem 14 (Soundness). *For all terms* $t, t' \in \mathcal{P}_{\mathsf{rec}}$ *that are closed terms of* $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$, $t = t'$ *is derivable from the axioms of* $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ *and/or axiom DRB5 only if* $t \underleftrightarrow{}_{\mathsf{rb}} t'$.

Proof. Because $\underleftrightarrow{}_{\mathsf{rb}}$ is a congruence with respect to all operators from the signature of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$, only the validity of each axiom of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ has to be proved. By Theorem 7, the axioms of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ are valid with respect

to $\underline{\leftrightarrow}_{\mathsf{rb}}$. By Theorem 11, validity with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ implies validity with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$. Therefore, it is sufficient to prove the validity of the instances of axiom schema DRB5 with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$. The notation introduced in the proof of Theorem 7 is used here as well.

For each instance ax of axiom schema DRB5, a dormancy-aware branching bisimulation R_{ax} witnessing the validity of ax can be constructed as follows:

$$
\begin{aligned}
R_{ax} = \{(t, t') \mid t = t' \in csi(ax)\} \\
\cup \{(t, t') \mid t = t' \in csi(\sigma^n(\underline{\tau} \cdot \sigma(x)) + y = \sigma^n(\sigma(x)) + y)\} \\
\cup \{(t, t') \mid t = t' \in csi(\underline{\tau} \cdot \sigma(x) = \sigma(x))\} \cup R_{id} .
\end{aligned}
$$

It is straightforward to check that the constructed relation R_{ax} is a dormancy-aware branching bisimulation witnessing, for each closed substitution instance $t = t'$ of ax, $t \underline{\leftrightarrow}_{\mathsf{rb}} t'$. □

As in the case of the axiom system of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ and $\underline{\leftrightarrow}_{\mathsf{rb}}$, the axiom system of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ extended with axiom DRB5 is not complete with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ in which only the constants and operators of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ occur. It is an open problem whether the axiom system of $\mathrm{BPA}_{\mathrm{drt}}^\tau$ extended with axiom DRB5 is complete with respect to $\underline{\leftrightarrow}_{\mathsf{rb}}$ for equations between terms from $\mathcal{P}_{\mathsf{rec}}$ in which only the constants and operators of $\mathrm{BPA}_{\mathrm{drt}}^\tau$ occur.

5.3 Verification in Two Phases

If axiom DRB5 is possibly needed, a natural way to verify an equation of the form $\tau_I(t) = t'$, for $t, t' \in \mathcal{P}_{\mathsf{rec}}$ where no abstraction operator occurs in t and only constants and operators of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ occur in t', is the following two-phase way:

– first, derive from the axioms of $\mathrm{ACP}_{\mathrm{drt}}^\tau + \mathrm{REC}$ an equation $t = t''$ for some $t'' \in \mathcal{P}_{\mathsf{rec}}$ in which only constants and operators of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ occur;
– then, derive from the axioms of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ and DRB5 the equation $\tau_I(t'') = t'$.

This raises the question whether the first phase is always possible. This question can be answered in the affirmative.

Theorem 15. *For all terms $t \in \mathcal{P}_{\mathsf{rec}}$ in which no abstraction operator occurs, there exists a term $t' \in \mathcal{P}_{\mathsf{rec}}$ that is a closed term of $\mathrm{BPA}_{\mathrm{drt}}^\tau + \mathrm{REC}$ such that $t = t'$ is derivable from the axioms of $\mathrm{ACP}_{\mathrm{drt}}^\tau + \mathrm{REC}$.*

Proof. Define the set $\mathcal{L}$ of *linear terms* over $\mathrm{ACP}_{\mathrm{drt}}^\tau$ to be the smallest set satisfying:

– if $a \in \mathsf{A}_{\tau\delta}$, then $\underline{a} \in \mathcal{L}$;
– if $a \in \mathsf{A}_\tau$ and $X \in \mathcal{X}$, then $\underline{a} \cdot X \in \mathcal{L}$;

34

– if $X \in \mathcal{X}$, then $\sigma(X) \in \mathcal{L}$;
– if $t, t' \in \mathcal{L}$, then $t + t' \in \mathcal{L}$.

Define a *linear recursive specification* over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC as a guarded recursive specification E over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC where, for each equation $X = t \in E$, $t \in \mathcal{L}$.

Claim that, for each guarded recursive specification E over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC and each $X \in \mathrm{V}(E)$, there exists a linear recursive specification E' over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC such that $\langle X|E \rangle = \langle X|E' \rangle$ is derivable from the axioms of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC. The proof of this claim goes essentially the same as the proof of Theorem 1 from [15].

Claim that, for all terms $t \in \mathcal{P}_{\mathsf{rec}}$ in which no abstraction operator occurs, there exists a linear recursive specification E over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC and an $X \in \mathrm{V}(E)$ such that $t = \langle X|E \rangle$ is derivable from the axioms of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC. The proof of this claim goes essentially the same as the proof of Lemma 1 from [21] except that the case where t is a constant $\langle X|E \rangle$ now requires the use of the previous claim. The proof involves constructions of linear recursive specifications from linear recursive specifications for the operators of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC other than the abstraction operators. For the greater part, the constructions are reminiscent of operations on process graphs defined in Sections 2.7 and 4.5.5 from [6].

The theorem follows immediately from the last claim because a constant $\langle X|E \rangle$ of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC where E is linear recursive specification over $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC is also a constant of $\mathrm{BPA}_{\mathrm{drt}}^{\tau}$+REC. □

6 Analysis of the PAR Protocol Revisited

We take up the analysis of the PAR protocol again, but now using the additional axiom DRB5.

We abstract from the actions in I, but not from the timing of actions. Starting from the specification of $\partial_H(S \parallel K \parallel L \parallel R)$ in Section 3.2, we can now calculate that $\tau_I(\partial_H(S \parallel K \parallel L \parallel R))$ is the X'''-component of the solution of the guarded recursive specification that consists of the following equations:

$$X''' = \sum_{d \in D} \underline{r_1(d)} \cdot \sigma^{t_S}(Y_d''') + \sigma(X''') ,$$

$$Y_d''' = \sigma^{t_K + t_R}(\underline{s_2(d)}) \cdot (\sigma^{t'_R + t_L}(\underline{\tau} \cdot X''') + \sigma^{t'_S - (t_K + t_R)}(Z''')) + \sigma^{t'_S}(Y_d''') ,$$

$$Z''' = \sigma^{t_K + t'_R + t_L}(\underline{\tau} \cdot X''') + \sigma^{t'_S}(Z''') .$$

Many simplifications have been achieved by using axiom DRB5. In particular, all internal actions that hinder performance analysis could be removed. The ones that could not be removed originate from the communications of acknowledgements at port 5 and these may be followed by the reception of a datum in the same time slice.

It is also possible to show that the PAR protocol is functionally correct by abstracting from the timing of actions next. That is, we can now calculate,

starting from $\pi_{\mathsf{tf}}(X''')$, that $\pi_{\mathsf{tf}}(\tau_I(\partial_H(S \parallel K \parallel L \parallel R)))$ is the solution of the guarded recursive specification of a buffer with capacity one.

A more intelligible specification of $\tau_I(\partial_H(S \parallel K \parallel L \parallel R))$ can be obtained if we add to $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$, for each $n \in \mathbb{N}$, a unary operator σ^{*n} to the signature of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$ and the equation $\sigma^{*n}(x) = x + \sigma^n(\sigma^{*n}(x))$ and the conditional equation $y = x + \sigma^n(y) \Rightarrow y = \sigma^{*n}(x)$ to the axiom system of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$. By using this extension of $\mathrm{ACP}_{\mathrm{drt}}^{\tau}+\mathrm{REC}$, the specification of $\tau_I(\partial_H(S \parallel K \parallel L \parallel R))$ given above can easily be rewritten to the following one:

$$X''' = \sum_{d \in D} r_1(d) \cdot \sigma^{*t'_S}(\sigma^{t_S + t_K + t_R}(\underline{\underline{s_2(d)}}))$$
$$\cdot (\sigma^{t'_R + t_L}(\underline{\tau} \cdot X''') + \sigma^{*t'_S}(\sigma^{t'_R + t_L + t'_S - t_R}(\underline{\tau} \cdot X'''))) \ .$$

This specification clearly exhibits both the functional behaviour and the performance properties of the PAR protocol. It is manifest that the protocol behaves as a buffer with capacity one, that a datum is delivered $t_S + t_K + t_R + i \cdot t'_S$ time slices after its consumption (for some $i \geq 0$), and that the next datum can be consumed $t'_R + t_L$ or $t'_R + t_L + t'_S - t_R + j \cdot t'_S$ time slices after the delivery (for some $j \geq 0$).

7 Concluding Remarks

The PAR protocol has been described and analyzed using a version of ACP with abstraction for processes with discrete relative timing that can be considered the core of the versions presented earlier in [2,3,4]. The protocol could be described adequately. In addition, its functional correctness could be analyzed thoroughly. That is, it could be proved that the protocol behaves correctly if the time-out time of the sender is longer than the time that a complete protocol cycle takes. Its performance properties could not be analyzed as thoroughly. To remedy this, a variant of the standard notion of branching bisimilarity for processes with discrete relative timing, called dormancy-aware branching bisimilarity, has been proposed. This variant is coarser than the standard notion, but it still coincides with the original notion of branching bisimilarity from [16] in the case of processes without timing. A plausible motivation for the variant has been given. It has also been shown that an additional axiom schema that is valid with respect to the variant permits thorough performance analysis of the PAR protocol.

In [20], the PAR protocol is described and analyzed using a version of ACP with abstraction for processes with continuous absolute timing. The description originates from an earlier description in [1]. In the case of the analysis in [20], handwavings are needed to come as far as in the case of our analysis. Other protocols that are described and analyzed in versions of ACP with abstraction for processes with timing include the ABP (Alternating Bit Protocol) and the CABP (Concurrent Alternating Bit Protocol), both in [19], and Fischer's mutual exclusion protocol, in [28]. In virtually all cases, there is a need for a coarser

equivalence. The equivalences suggested in [19,28] are in the case of processes without timing also coarser than the original version of branching bisimilarity from [16]. I claim that, in the case of the above-mentioned protocols, there is no need for an equivalence that is coarser than dormancy-aware branching bisimilarity.

Dormancy-aware branching bisimilarity is most closely related to the equivalences that abstract from silent steps introduced for other versions of ACP with abstraction for processes with timing. The equivalences in question are all versions of branching bisimilarity for processes with timing. Dormancy-aware branching bisimilarity is coarser than the version of branching bisimilarity for processes with discrete relative timing introduced in [2,3,4]. The latter version is in line with the version of branching bisimilarity for processes with continuous relative timing introduced in [20] and both versions are based on the version of branching bisimilarity for processes without timing from [16]. In [20], an outline of an unnamed coarser equivalence, denoted by $\underleftrightarrow{}^*_{\text{rb}}$, is given as well. That equivalence can be regarded as the first step towards dormancy-aware branching bisimilarity: it looks as if its outline in [20] is the point of departure of the unsatisfactory definition of dormancy-aware branching bisimilarity in [5].

It is difficult to compare dormancy-aware branching bisimilarity with the equivalences that abstract from silent steps introduced for other process algebras for processes with timing, such as the different versions of CCS with timing [10,22,30], Timed CSP [11], TIC [24] and TPL [18]. The reason for it is that they are based on equivalences other than branching bisimilarity. For example, the equivalences for the different versions of CCS with timing and TIC are all based on weak bisimilarity, the equivalence for Timed CSP is based on failure equivalence, and the different equivalences for TPL are based on the different testing equivalences. For a comparison of those equivalences for processes without timing, and many others that abstract from silent steps, the reader is referred to [13].

In [25], an alternative for dormancy-aware branching bisimilarity is proposed. The underlying idea of that proposal is that abstraction from internal actions involves full abstraction from their timing. This approach leads to rather counterintuitive situations. For example, with this approach applied to the setting of the current paper, we would have that $\tau_{\{b\}}(\sigma(\underline{a}) \parallel \underline{b}) = \delta$ and $\sigma(\underline{a}) \parallel \tau_{\{b\}}(\underline{b}) \neq \delta$ whereas one intuitively expects to have that $\tau_{\{b\}}(\sigma(\underline{a}) \parallel \underline{b}) = \sigma(\underline{a}) \parallel \tau_{\{b\}}(\underline{b})$. Actually, the setting of [25] is one in which timing is handled by time stamping of actions, timing is absolute, and important operators such as the parallel composition operator are not covered.

As explained in Section 2, $\text{ACP}^\tau_{\text{drt}}$ is the core of the process algebra $\text{ACP}^{\text{drt}}_\tau$ presented in [4]. Another ACP-style process algebra closely related to $\text{ACP}^\tau_{\text{drt}}$ is $\text{ACP}_{\text{drt}}\tau$ from [2,3]. $\text{ACP}_{\text{drt}}\tau$ has, in addition to the constants of $\text{ACP}^\tau_{\text{drt}}$, the deadlocked process constant $\dot{\delta}$ from $\text{ACP}^{\text{drt}}_\tau$, the delayable action constant a for each $a \in \mathsf{A}$, the delayable silent step constant τ, and the delayable deadlock constant δ. Delayable action constants denote processes that perform an observable action in any time slice and after that immediately terminate succesfully.

The delayable silent step and delayable deadlock constants can be explained analogously. The axioms defining the delayable action, silent step, and deadlock constants make use of an additional unbounded start delay operator in [2] and a time iteration operator in [3]. With the exception of the process denoted by the deadlocked process constant, the processes denoted by these constants are definable in $\mathrm{ACP}_{\mathrm{drt}}^{\tau}$+REC.

References

1. Baeten, J.C.M., Bergstra, J.A.: Real time process algebra. Formal Aspects of Computing 3(2), 142–188 (1991). doi:10.1007/BF01898401
2. Baeten, J.C.M., Bergstra, J.A.: Discrete time process algebra with abstraction. In: Reichel, H. (ed.) Fundamentals of Computation Theory. Lecture Notes in Computer Science, vol. 965, pp. 1–15. Springer-Verlag (1995). doi:10.1007/3-540-60249-6_38
3. Baeten, J.C.M., Bergstra, J.A., Reniers, M.A.: Discrete time process algebra with silent step. In: Plotkin, G.D., Stirling, C., Tofte, M. (eds.) Proof, Language and Interaction: Essays in Honour of Robin Milner, pp. 535–569. MIT Press, Cambridge, MA (2000)
4. Baeten, J.C.M., Middelburg, C.A.: Process Algebra with Timing. Monographs in Theoretical Computer Science, An EATCS Series, Springer-Verlag, Berlin (2002). doi:10.1007/978-3-662-04995-2
5. Baeten, J.C.M., Middelburg, C.A., Reniers, M.A.: A new equivalence for processes with timing – with an application to protocol verification. Computer Science Report 02-10, Department of Mathematics and Computer Science, Eindhoven University of Technology (2002). Electronic version
6. Baeten, J.C.M., Weijland, W.P.: Process Algebra, Cambridge Tracts in Theoretical Computer Science, vol. 18. Cambridge University Press, Cambridge (1990). doi:10.1017/CB09780511624193
7. Basten, A.A.: Branching bisimulation is an equivalence indeed! Information Processing Letters 58(3), 141–147 (1996). doi:10.1016/0020-0190(96)00034-8
8. Bergstra, J.A., Klop, J.W.: Verification of an alternating bit protocol by means of process algebra. In: Bibel, W., Jantke, K.P. (eds.) Mathematical Methods of Specification and Synthesis of Software Systems. Lecture Notes in Computer Science, vol. 215, pp. 9–23. Springer-Verlag (1986). doi:10.1007/3-540-16444-8_1
9. Brunekreef, J.J.: Process specification in a UNITY format. In: Ponse, A., Verhoef, C., van Vlijmen, S.F.M. (eds.) Algebra of Communicating Processes 1994. pp. 319–337. Workshops in Computing Series, Springer-Verlag (1995). doi:10.1007/978-1-4471-2120-6_14
10. Chen, L.: An interleaving model for real-time systems. In: Nerode, A., Taitslin, M. (eds.) Symposium on Logical Foundations of Computer Science. Lecture Notes in Computer Science, vol. 620, pp. 81–92. Springer-Verlag (1992). doi:10.1007/BFb0023865
11. Davies, J., et al.: Timed CSP: Theory and practice. In: de Bakker, J.W., Huizing, C., de Roever, W.P., Rozenberg, G. (eds.) Real Time: Theory and Practice. Lecture Notes in Computer Science, vol. 600, pp. 640–675. Springer-Verlag (1992). doi:10.1007/BFb0032011
12. Fokkink, W.J.: Rooted branching bisimulation as a congruence. Journal of Computer and System Sciences 60(1), 13–37 (2000). doi:10.1006/jcss.1999.1663

13. van Glabbeek, R.J.: The linear time – branching time spectrum II. In: Best, E. (ed.) CONCUR'93. Lecture Notes in Computer Science, vol. 715, pp. 66–81. Springer-Verlag (1993). doi:10.1007/3-540-57208-2_6

14. van Glabbeek, R.J.: The meaning of negative premises in transition system specifications II. Journal of Logic and Algebraic Programming 60–61, 229–258 (2004). doi:10.1016/j.jlap.2004.03.007

15. van Glabbeek, R.J., Middelburg, C.A.: On infinite guarded recursive specifications in process algebra. arXiv:2005.00746 [cs.LO] (2020)

16. van Glabbeek, R.J., Weijland, W.P.: Branching time and abstraction in bisimulation semantics. Journal of the ACM 43(3), 555–600 (1996). doi:10.1145/233551.233556

17. Goguen, J.A.: Theorem proving and algebra. arXiv:2101.02690 [cs.LO] (2021)

18. Hennessy, M., Regan, T.: A process algebra for timed systems. Information and Computation 117, 221–239 (1995). doi:10.1006/inco.1995.1041

19. Hillebrand, J.A.: The ABP and CABP – a comparison of performances in real time process algebra. In: Ponse, A., Verhoef, C., van Vlijmen, S.F.M. (eds.) Algebra of Communicating Processes 1994. pp. 124–147. Workshops in Computing Series, Springer-Verlag (1995). doi:10.1007/978-1-4471-2120-6_6

20. Klusener, A.S.: Models and Axioms for a Fragment of Real Time Process Algebra. Ph.D. thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven (1993). doi:10.6100/IR408754

21. Middelburg, C.A.: Imperative process algebra with abstraction. Scientific Annals of Computer Science 32(1), 137–179 (2022). doi:10.7561/SACS.2022.1.137

22. Moller, F., Tofts, C.: Behavioural abstraction in TCCS. In: Kuich, W. (ed.) Proceedings 19th ICALP. Lecture Notes in Computer Science, vol. 623, pp. 559–570. Springer-Verlag (1992). doi:10.1007/3-540-55719-9_104

23. Ponse, A.: Process expressions and Hoare's logic: Showing an irreconcilability of context-free recursion with Scott's induction rule. Information and Computation 95(2), 192–217 (1991). doi:10.1016/0890-5401(91)90044-3

24. Quemada, J., de Frutos, D., Azcorra, A.: TIC: A timed calculus. Formal Aspects of Computing 5(3), 224–252 (1993). doi:10.1007/BF01211556

25. Reniers, M.A., van Weerdenburg, M.: Action abstraction in timed process algebra. In: Arbab, F., Sirjani, M. (eds.) FSEN 2007. Lecture Notes in Computer Science, vol. 4767, pp. 287–301. Springer-Verlag (2007). doi:10.1007/978-3-540-75698-9_19

26. Tanenbaum, A.S., Feamster, N., Wetherall, D.J.: Computer Networks, Sixth Edition. Pearson, Boston (2021)

27. Vaandrager, F.W.: Two simple protocols. In: Baeten, J.C.M. (ed.) Applications of Process Algebra, Cambridge Tracts in Theoretical Computer Science, vol. 17, pp. 23–44. Cambridge University Press, Cambridge (1990). doi:10.1017/CBO9780511608841.003

28. Vereijken, J.J.: Fischer's protocol in timed process algebra. In: Ponse, A., Verhoef, C., van Vlijmen, S.F.M. (eds.) Algebra of Communicating Processes 1995. pp. 245–284. No. 95-14 in Computer Science Report, Department of Mathematics and Computer Science, Eindhoven University of Technology (1995). Electronic version

29. Vereijken, J.J.: Discrete Time Process Algebra. Ph.D. thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven (1997). doi:10.6100/IR498558

30. Wang Yi: Real-time behaviour of asynchronous agents. In: Baeten, J.C.M., Klop, J.W. (eds.) CONCUR'90. Lecture Notes in Computer Science, vol. 458, pp. 502–520. Springer-Verlag (1990). doi:10.1007/BFb0039080