Preferential Choice and Coordination Conditions

J. A. Bergstra [a,b], C. A. Middelburg [c,a,*]

[a] *Programming Research Group, University of Amsterdam, P.O. Box 41882, 1009 DB Amsterdam, Netherlands*

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

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

Abstract

We present a process algebra with conditional expressions of which the conditions concern the enabledness of actions in the context in which a process is placed. With those conditions, it becomes easy to model preferential choices. A preferential choice of a process is a choice whereby certain alternatives are excluded if at least one of the other alternatives is permitted by the context in which the process is placed. Preferential choices are often modelled rather indirectly using a priority mechanism.

Key words: coordination conditions, preferential choice, splitting bisimulation, process algebra, Boolean algebra

1 Introduction

In Ref. [1], we started an investigation into the potentialities of conditional expressions in the setting of the algebraic theory about processes known as ACP [2,3]. The primary intention of the investigation is to find basic ways to increase expressiveness. The main extensions of ACP introduced in Ref. [1]

* Corresponding author.
Email addresses: janb@science.uva.nl (J. A. Bergstra), keesm@win.tue.nl (C. A. Middelburg).
URLs: http://www.science.uva.nl/~janb (J. A. Bergstra), http://www.win.tue.nl/~keesm (C. A. Middelburg).

are $\mathrm{ACP}^{\mathrm{c}}$ (ACP with conditional expressions), $\mathrm{ACP}^{\mathrm{cs}}$ ($\mathrm{ACP}^{\mathrm{c}}$ with signals) and $\mathrm{ACP}^{\mathrm{cr}}$ ($\mathrm{ACP}^{\mathrm{c}}$ with retrospective conditions).

In the current paper, we proceed with the investigation started in Ref. [1]. We present $\mathrm{ACP}^{\mathrm{cc}}$, a variant of $\mathrm{ACP}^{\mathrm{c}}$ in which the conditions concern the enabledness of actions in the context in which a process is placed. Such conditions are called coordination conditions in this paper because they are primarily intended for coordination of processes that proceed in parallel. The merit of $\mathrm{ACP}^{\mathrm{cc}}$ is primarily the following. If a process has different alternatives to proceed, some may be preferred in the sense that the others should be excluded if at least one of the preferred alternatives is permitted by the context in which the process is placed. Such preferential choices are often modelled rather indirectly using a priority mechanism, see e.g. Ref. [4], but can easily be modelled in $\mathrm{ACP}^{\mathrm{cc}}$.

In Ref. [5], ACP has been extended with a priority mechanism. However, this priority mechanism, although suitable for dealing with interrupts, is unsuitable for modelling preferential choices, because it does not feature local pre-emption in the sense of Ref. [4]. The priority mechanism used in Ref. [4] to model preferential choices does feature local pre-emption. Such a priority mechanism can be added to ACP as well. However, the solution to model preferential choices with a priority mechanism featuring local pre-emption lacks the property that there is a close connection with a simple explanation of preferential choices, such as the explanation given in the preceding paragraph. By modelling preferential choices with coordination conditions, a close connection with the simple explanation given in the preceding paragraph can be achieved. This forms part of our motivation to develop $\mathrm{ACP}^{\mathrm{cc}}$.

Features for preferential choices are found in programming languages and process algebraic formalisms. To the best of our knowledge, the first programming language with a construct for preferential choices is occam [6]. The occam programming language is based on CSP [7,8]. However, the construct for preferential choices included in occam, which is known as the `PRI ALT` construct, has no counterpart in CSP. A counterpart of the `PRI ALT` construct in an extension of CSP, with a formal semantics in CSP-style, is introduced for the first time in Ref. [9].

A counterpart of the `PRI ALT` construct in an extension of CCS [10,11], with a formal semantics in CCS-style, is introduced for the first time in Ref. [12]. It appears that the counterpart of the `PRI ALT` construct introduced in Ref. [12], which is known as the priority choice operator, has inspired the addition of priority guards to CCS in Ref. [13]. In the latter paper, the basic idea is to take action prefixes of the form $U : a$, where a is an action and U is a finite set of actions, called a priority guard. This is meant to indicate that performing a is blocked if at least one of the actions in U is enabled by the context. In

other words, performing a is made conditional on a condition concerning the enabledness of actions by altering the syntax of action prefixes. An obvious alternative for altering the syntax of action prefixes is adding an operator for making behaviour conditional, together with operators for making conditions out of other conditions. This is more in the spirit of process algebras with general sequential composition instead of action prefixing, like ACP. All this brings us to believe that it is worthwhile to elaborate on the modelling of preferential choices in the setting of $\mathrm{ACP}^{\mathrm{c}}$.

The presentation of $\mathrm{ACP}^{\mathrm{cc}}$ includes the axioms of $\mathrm{ACP}^{\mathrm{cc}}$ and the main models of $\mathrm{ACP}^{\mathrm{cc}}$. Those models are based on labelled transition systems of which the labels consist of a condition and an action, called conditional transition systems, and a variant of bisimilarity in which a transition of one of the related transition systems may be simulated by a set of transitions of the other transition system, called splitting bisimilarity. The presented models cover finitely branching processes as well as infinitely branching processes. We also extend $\mathrm{ACP}^{\mathrm{cc}}$ with a preferential choice operator to show that it is easy to give defining equations for a preferential choice operator in the presence of conditional expressions of which the conditions are coordination conditions. This extension of $\mathrm{ACP}^{\mathrm{cc}}$ can be viewed as an application of $\mathrm{ACP}^{\mathrm{cc}}$ that remains entirely within the domain of process algebra.

What is attained is primarily an extension of ACP that provides the possibility to model preferential choices in such a way that a close connection with a simple explanation of preferential choices is achieved. In attaining this result, the introduction of coordination conditions is considered to be crucial. Coordination conditions make it possible to abstract from the details of the mechanism by which a process is brought to proceed at each stage according to the enabledness of actions in the context in which it is placed. In the case of a priority mechanism featuring local pre-emption, for example, the modelling of a process that begins by making a preferential choice depends upon the process of which it happens to be a subprocess. In the case of coordination conditions, the modelling of a process that begins by making a preferential choice concerns that preferential choice only.

The coordination conditions of $\mathrm{ACP}^{\mathrm{cc}}$ are similar to the priority guards added to CCS in Ref. [13]. Priority guards can be regarded as coordination conditions of a simple form which can only be used together with action prefixing. Moreover, unlike the priority guards, the coordination conditions do not assume CCS-style communication. Therefore, we consider coordination conditions an improvement on priority guards.

The preferential choice operator added to $\mathrm{ACP}^{\mathrm{cc}}$ is similar to the priority choice operator added to CCS in Ref. [12]. The preferential choice operator is defined for all possible operands, whereas syntactic restrictions are imposed on

the operands of the priority choice operator of Ref. [12]. Moreover, unlike the priority choice operator, the preferential choice operator does not assume CCS-style communication. Therefore, we consider the preferential choice operator an improvement on the priority choice operator. Note that, because it is not defined for all possible operands, the priority choice operator violates the basics of an algebraic approach. The modelling of preferential choices in Ref. [4] mentioned above is based on Ref. [12] and works only under the same syntactic restrictions.

The structure of this paper is as follows. First of all, we introduce $\mathrm{PA}_\delta^{\mathrm{cc}}$, a simple precursor of $\mathrm{ACP}^{\mathrm{cc}}$ that does not support communication (Section 2). After that, we introduce conditional transition systems and splitting bisimilarity of conditional transition systems (Section 3) and the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$, the main models of $\mathrm{PA}_\delta^{\mathrm{cc}}$ (Section 4). Following this, we have a closer look at splitting bisimilarity based on structural operational semantics (Section 5). Next, we move from $\mathrm{PA}_\delta^{\mathrm{cc}}$ to $\mathrm{ACP}^{\mathrm{cc}}$ (Section 6) and adapt the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$ to full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$ (Section 7). Then, we extend $\mathrm{ACP}^{\mathrm{cc}}$ with guarded recursion (Section 8). Thereupon, we extend $\mathrm{ACP}^{\mathrm{cc}}$ with the preferential choice operator (Section 9). Following this, we give examples of the use of coordination conditions and the preferential choice operator (Section 10). Finally, we make some remarks about related work and mention some options for future work (Section 11).

Some familiarity with the field of Boolean algebra is desirable. The definitions of all notions concerning Boolean algebras that are used in this paper can, for example, be found in Ref. [14].

2 PA_δ with Coordination Conditions

PA_δ is a subtheory of ACP that does not support communication (see e.g. Ref. [3]). In this section, we present an extension of PA_δ with encapsulation, pre-abstraction and guarded commands, called $\mathrm{PA}_\delta^{\mathrm{cc}}$. Encapsulation was originally incorporated in ACP to encapsulate actions of a process from communication with actions from the outside (see Ref. [2]). Pre-abstraction was added to ACP in Ref. [15] as a limited kind of abstraction: the actions from which is abstracted are identified, but they remain observable as internal actions. [1] Guarded commands are conditional expressions of the form $\zeta :\rightarrow p$, where ζ and p are expressions representing a condition and a process, respectively. Guarded commands were added to ACP for the first time in Ref. [17].

[1] This limited kind of abstraction was not given a name in Ref. [15]. The name pre-abstraction originates from Ref. [16].

In $\mathrm{PA}_\delta^{\mathrm{cc}}$, just as in PA_δ or ACP extended with pre-abstraction, it is assumed that a fixed but arbitrary finite set of *actions* A, with $\delta \notin \mathsf{A}$ and $t \in \mathsf{A}$, has been given. Action t is the internal action that replaces all occurrences of actions from which is abstracted by means of pre-abstraction. Henceforth, we write A_δ for $\mathsf{A} \cup \{\delta\}$.

If it is permitted by the context in which a process is placed to perform an action a, we say that a is enabled in that context. In $\mathrm{PA}_\delta^{\mathrm{cc}}$, we consider conditions concerning the enabledness of actions. More precisely, conditions are taken from the quotient algebra of the free Boolean algebra over $\{\mathcal{E}_a \mid a \in \mathsf{A}\}$ by the equivalence induced by the equation $\mathcal{E}_t = \top$. Henceforth, we write $\mathcal{C}$ for this algebra and also for the domain of this algebra. The intuition is that $\mathcal{E}_a$ holds if action a is enabled in the context present. Because no context can prevent the internal action t from being performed, t is enabled in any context. The elements of $\mathcal{C}$ are called *coordination* conditions.

The algebraic theory $\mathrm{PA}_\delta^{\mathrm{cc}}$ has two sorts:

- the sort $\mathbf{P}$ of *processes*;
- the sort $\mathbf{C}$ of *conditions*.

The algebraic theory $\mathrm{PA}_\delta^{\mathrm{cc}}$ has the following constants and operators to build terms of sort $\mathbf{P}$:

- the *deadlock* constant $\delta : \mathbf{P}$;
- for each $a \in \mathsf{A}$, the *action* constant $a : \mathbf{P}$;
- the binary *alternative composition* operator $+ : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$;
- the binary *sequential composition* operator $\cdot : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$;
- the binary *guarded command* operator $:\!\to : \mathbf{C} \times \mathbf{P} \to \mathbf{P}$;
- the binary *parallel composition* operator $\parallel : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$;
- the binary *left merge* operator $\between : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$;
- for each $H \subseteq \mathsf{A} \setminus \{t\}$, the unary *encapsulation* operator $\partial_H : \mathbf{P} \to \mathbf{P}$;
- for each $I \subseteq \mathsf{A}$, the unary *pre-abstraction* operator $t_I : \mathbf{P} \to \mathbf{P}$.

The algebraic theory $\mathrm{PA}_\delta^{\mathrm{cc}}$ has the following constants and operators to build terms of sort $\mathbf{C}$:

- the *bottom* constant $\bot : \mathbf{C}$;
- the *top* constant $\top : \mathbf{C}$;
- for each $a \in \mathsf{A}$, the *enabledness* constant $\mathcal{E}_a : \mathbf{C}$;
- the unary *complement* operator $- : \mathbf{C} \to \mathbf{C}$;
- the binary *join* operator $\sqcup : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$;
- the binary *meet* operator $\sqcap : \mathbf{C} \times \mathbf{C} \to \mathbf{C}$;
- for each $H \subseteq \mathsf{A} \setminus \{t\}$, the unary *encapsulation* operator $\partial_H : \mathbf{C} \to \mathbf{C}$;
- for each $I \subseteq \mathsf{A}$, the unary *pre-abstraction* operator $t_I : \mathbf{C} \to \mathbf{C}$.

Terms of sorts $\mathbf{P}$ and $\mathbf{C}$ are built as usual for a many-sorted signature (see e.g. Refs. [18,19]). Throughout the paper, we assume that there are infinitely many variables of sort $\mathbf{P}$, including $x, y, z, x_1, y_1, x_2, y_2, \ldots$, and infinitely many variables of sort $\mathbf{C}$, including $\phi, \psi, \chi, \phi_1, \psi_1, \phi'_1, \psi'_1, \phi_2, \psi_2, \phi'_2, \psi'_2, \ldots$.

We use infix notation for the binary operators. The following precedence conventions are used to reduce the need for parentheses. The operators to build terms of sort $\mathbf{C}$ bind stronger than the operators to build terms of sort $\mathbf{P}$. The operator $+$ binds weaker than all other binary operators to build terms of sort $\mathbf{P}$ and the operators $\cdot$ and $:\!\!\rightarrow$ bind stronger than all other binary operators to build terms of sort $\mathbf{P}$. The operator $\cdot$ binds weaker than $:\!\!\rightarrow$.

Let p and q be closed terms of sort $\mathbf{P}$, ζ and ξ be closed terms of sort $\mathbf{C}$, $a \in \mathsf{A}$, $H \subseteq \mathsf{A} \setminus \{t\}$, and $I \subseteq \mathsf{A}$. Intuitively, the constants and operators to build terms of sort $\mathbf{P}$ can be explained as follows:

- δ can neither perform an action nor terminate successfully;
- a first performs action a unconditionally and then terminates successfully;
- $p + q$ behaves either as p or as q, but not both;
- $p \cdot q$ first behaves as p, but when p terminates successfully it continues by behaving as q;
- $\zeta :\!\!\rightarrow p$ behaves as p under condition ζ;
- $p \parallel q$ behaves as the process that proceeds with p and q in parallel;
- $p \lfloor\!\lfloor q$ behaves the same as $p \parallel q$, except that it starts with performing an action of p;
- $\partial_H(p)$ behaves the same as p, except that actions from H are blocked.
- $t_I(p)$ behaves the same as p, except that it performs the internal action t whenever p would perform an action in I.

Intuitively, the constants and operators to build terms of sort $\mathbf{C}$ can be explained as follows:

- $\mathcal{E}_a$ is a condition that holds if action a is enabled in the context present;
- $\bot$ is a condition that never holds;
- $\top$ is a condition that always holds;
- $-\zeta$ is the opposite of ζ;
- $\zeta \sqcup \xi$ is ζ or ξ;
- $\zeta \sqcap \xi$ is both ζ and ξ;
- $\partial_H(\zeta)$ is ζ with all enabledness conditions $\mathcal{E}_a$ with $a \in H$ replaced by $\bot$;
- $t_I(\zeta)$ is ζ with all enabledness conditions $\mathcal{E}_a$ with $a \in I$ replaced by $\top$.

In $\mathrm{PA}_\delta^{\mathrm{cc}}$, the enabledness of actions is affected only by encapsulation and pre-abstraction. Encapsulation places a process in a context in which the encapsulated actions are disabled and pre-abstraction places a process in a context in which the pre-abstracted actions are enabled.

Table 1
Axioms of Boolean algebras

$\phi \sqcup \bot = \phi$	BA1	$\phi \sqcap \top = \phi$	BA5
$\phi \sqcup -\phi = \top$	BA2	$\phi \sqcap -\phi = \bot$	BA6
$\phi \sqcup \psi = \psi \sqcup \phi$	BA3	$\phi \sqcap \psi = \psi \sqcap \phi$	BA7
$\phi \sqcup (\psi \sqcap \chi) = (\phi \sqcup \psi) \sqcap (\phi \sqcup \chi)$	BA4	$\phi \sqcap (\psi \sqcup \chi) = (\phi \sqcap \psi) \sqcup (\phi \sqcap \chi)$	BA8

Some earlier extensions of ACP include conditional expressions of the form $p \lhd \zeta \rhd q$; see e.g. Ref. [20]. This notation with triangles originates from Ref. [21]. We treat conditional expressions of the form $p \lhd \zeta \rhd q$, where p and q are terms of sort $\mathbf{P}$ and ζ is a term of sort $\mathbf{C}$, as abbreviations. That is, we write $p \lhd \zeta \rhd q$ for $\zeta :\rightarrow p + -\zeta :\rightarrow q$.

The axioms of $\mathrm{PA}_\delta^{\mathrm{cc}}$ are the axioms of Boolean Algebras (BA) given in Table 1 and the additional axioms given in Table 2. Several axioms given in Table 2 are actually axiom schemas: a stands for an arbitrary element of A_δ, c stands for an arbitrary element of A, H stands for an arbitrary subset of $\mathsf{A} \setminus \{t\}$, and I stands for an arbitrary subset of A.

The axioms of BA given in Table 1 have been taken from Ref. [22]. Several alternatives for this axiomatization can be found in the literature. If we use basic laws of BA other than axioms BA1–BA8 in a step of a derivation, we will refer to them as applications of BA and not give their derivation from axioms BA1–BA8.

The axioms of $\mathrm{PA}_\delta^{\mathrm{cc}}$ include the axioms of PA_δ (A1–A7 and M1–M4), the usual axioms for encapsulation (D1–D4) and the usual axioms for pre-abstraction (I1–I4), see e.g. Ref. [16]. Axioms GC1–GC8 have been taken from Ref. [20], but in the case of GC5 and GC8 with adaptation to conditional expressions of the form $\zeta :\rightarrow p$. Axioms ET, GC11E, GC12E, ED1–ED7 and EI1–EI7 are new. Axiom ET expresses that the internal action t is enabled in any context. GC11E is not, like GC1–GC8, merely the adaptation of an axiom taken from Ref. [20] to conditional expressions of the form $\zeta :\rightarrow p$. That adaptation alone would yield ϕ instead of $\partial_H(\phi)$ in the right-hand side of GC11E. The reason for the additional adaptation is that encapsulation places a process in a context in which the encapsulated actions are disabled. Similarly, the reason for the appearance of $t_H(\phi)$ in the right-hand side of GC12E is that pre-abstraction places a process in a context in which the pre-abstracted actions are enabled. The defining axioms of encapsulation and pre-abstraction on conditions are ED1–ED7 and EI1–EI7, respectively.

We will use the sum notation $\sum_{i \in \mathcal{I}} p_i$, where $\mathcal{I} = \{i_1, \ldots, i_n\}$ and $p_{i_1}, \ldots, p_{i_n}$ are terms of sort $\mathbf{P}$, for $p_{i_1} + \ldots + p_{i_n}$. The convention is that $\sum_{i \in \mathcal{I}} p_i$ stands for δ if $\mathcal{I} = \emptyset$. Note that the sum notation is not used for alternative composition

Table 2
Axioms of $\mathrm{PA}_\delta^{\mathrm{cc}}$

$x + y = y + x$		A1	$\top :\rightarrow x = x$		GC1
$(x + y) + z = x + (y + z)$		A2	$\bot :\rightarrow x = \delta$		GC2
$x + x = x$		A3	$\phi :\rightarrow \delta = \delta$		GC3
$(x + y) \cdot z = x \cdot z + y \cdot z$		A4	$\phi :\rightarrow (x + y) = \phi :\rightarrow x + \phi :\rightarrow y$		GC4
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$		A5	$\phi :\rightarrow (x \cdot y) = \phi :\rightarrow x \cdot y$		GC5
$x + \delta = x$		A6	$\phi :\rightarrow (\psi :\rightarrow x) = \phi \sqcap \psi :\rightarrow x$		GC6
$\delta \cdot x = \delta$		A7	$\phi \sqcup \psi :\rightarrow x = \phi :\rightarrow x + \psi :\rightarrow x$		GC7
			$\phi :\rightarrow x \between y = \phi :\rightarrow (x \between y)$		GC8
			$\partial_H(\phi :\rightarrow x) = \partial_H(\phi) :\rightarrow \partial_H(x)$		GC11E
$x \parallel y = x \between y + y \between x$		M1	$t_I(\phi :\rightarrow x) = t_I(\phi) :\rightarrow t_I(x)$		GC12E
$a \between x = a \cdot x$		M2			
$a \cdot x \between y = a \cdot (x \parallel y)$		M3	$\partial_H(\bot) = \bot$		ED1
$(x + y) \between z = x \between z + y \between z$		M4	$\partial_H(\top) = \top$		ED2
			$\partial_H(\mathcal{E}_c) = \mathcal{E}_c$	if $c \notin H$	ED3
$\partial_H(a) = a$	if $a \notin H$	D1	$\partial_H(\mathcal{E}_c) = \bot$	if $c \in H$	ED4
$\partial_H(a) = \delta$	if $a \in H$	D2	$\partial_H(-\phi) = -\partial_H(\phi)$		ED5
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$		D3	$\partial_H(\phi \sqcup \psi) = \partial_H(\phi) \sqcup \partial_H(\psi)$		ED6
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$		D4	$\partial_H(\phi \sqcap \psi) = \partial_H(\phi) \sqcap \partial_H(\psi)$		ED7
$t_I(a) = a$	if $a \notin I$	I1	$t_I(\bot) = \bot$		EI1
$t_I(a) = t$	if $a \in I$	I2	$t_I(\top) = \top$		EI2
$t_I(x + y) = t_I(x) + t_I(y)$		I3	$t_I(\mathcal{E}_c) = \mathcal{E}_c$	if $c \notin I$	EI3
$t_I(x \cdot y) = t_I(x) \cdot t_I(y)$		I4	$t_I(\mathcal{E}_c) = \top$	if $c \in I$	EI4
			$t_I(-\phi) = -t_I(\phi)$		EI5
			$t_I(\phi \sqcup \psi) = t_I(\phi) \sqcup t_I(\psi)$		EI6
$\mathcal{E}_t = \top$		ET	$t_I(\phi \sqcap \psi) = t_I(\phi) \sqcap t_I(\psi)$		EI7

over an infinite set of alternatives.

An interesting subtheory of $\mathrm{PA}_\delta^{\mathrm{cc}}$ is $\mathrm{BPA}_\delta^{\mathrm{cc}}$. This subtheory is obtained by removing the parallel composition operator, the left merge operator, the encapsulation operators and the pre-abstraction operators from the signature of $\mathrm{PA}_\delta^{\mathrm{cc}}$ and removing all axioms in which these operators occur from the ax-

ioms of $\mathrm{PA}_\delta^{\mathrm{cc}}$ – in other words, the axioms of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ are BA1–BA8, A1–A7, GC1–GC7, ET.

To prove a statement for all closed terms of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$, it is sufficient to prove it for all basic terms. The set $\mathcal{B}$ of *basic terms* is inductively defined by the following rules:

- $\delta \in \mathcal{B}$;
- if ζ is a closed term of sort $\mathbf{C}$ and $a \in \mathsf{A}$, then $\zeta :\rightarrow a \in \mathcal{B}$;
- if ζ is a closed term of sort $\mathbf{C}$, $a \in \mathsf{A}$ and $p \in \mathcal{B}$, then $\zeta :\rightarrow a \cdot p \in \mathcal{B}$;
- if $p, q \in \mathcal{B}$, then $p + q \in \mathcal{B}$.

The basic terms are exactly the closed terms of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ of the form

$$\sum_{i<n} \zeta_i :\rightarrow a_i \cdot p_i + \sum_{i<m} \xi_i :\rightarrow b_i \ ,$$

where $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{m-1} \in \mathsf{A}$, $\zeta_0, \ldots, \zeta_{n-1}, \xi_0, \ldots, \xi_{m-1}$ are closed terms of sort $\mathbf{C}$ and $p_0, \ldots, p_{n-1}$ are basics terms $(n, m \geq 0)$. We can prove that all closed terms of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ are derivably equal to a basic term.

Lemma 1 (Elimination for $\mathrm{BPA}_\delta^{\mathrm{cc}}$) *For all closed terms p of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$, there exists a basic term $q \in \mathcal{B}$ such that $\mathrm{BPA}_\delta^{\mathrm{cc}} \vdash p = q$.*

PROOF. The term rewriting system consisting of axioms A4–A7 and GC1–GC7 oriented from left to right is strongly normalizing. This can be proved by using the method of lexicographical path ordering of Kamin and Lévy (see e.g. Ref. [23]), making the signature one-sorted, taking the ordering $:\rightarrow > \cdot > +$, $:\rightarrow > \delta$, $:\rightarrow > \sqcap$, and giving the lexicographical status for the first argument to $\cdot$ and the lexicographical status for the second argument to $:\rightarrow$. Moreover, it is easy to see that each normal form is a basic term. $\square$

We can also prove that all closed terms of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$ are derivably equal to a basic term.

Lemma 2 (Elimination for $\mathrm{PA}_\delta^{\mathrm{cc}}$) *For all closed terms p of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$, there exists a basic term $q \in \mathcal{B}$ such that $\mathrm{PA}_\delta^{\mathrm{cc}} \vdash p = q$.*

PROOF. In the proof use is made of the weight of a term p, written $|p|$. It is inductively defined as follows: $|\diamond_0| = 1$ for constants $\diamond_0$, $|\diamond_1(p)| = |p|$ for

unary operators $\diamond_1$, $|p \diamond_2 q| = |p| + |q|$ for binary operators $\diamond_2$ not in $\{+, \sqcup\}$, and $|p \diamond_2 q| = \max\{|p|, |q|\}$ for $\diamond_2$ in $\{+, \sqcup\}$.

The term rewriting system consisting of all axioms of $\mathrm{PA}_\delta^{\mathrm{cc}}$, except BA1–BA8 and A1–A3, oriented from left to right is strongly normalizing. This can be proved by using the method of lexicographical path ordering of Kamin and Lévy, making the signature one-sorted, ranking the operators $\parallel$ and $\between$ by the sum of the weights of their arguments as in Ref. [24], taking the ordering $\ldots > \parallel_3 > \between_3 > \parallel_2 > \between_2 > :\!\rightarrow > \cdot > +$, $:\!\rightarrow > \delta$, $:\!\rightarrow > \sqcap$, $\partial_H > :\!\rightarrow$, $\partial_H > \bot$, $\partial_H > -$, $\partial_H > \sqcup$, $t_I > :\!\rightarrow$, $t_I > t$, $t_I > \top$, $t_I > -$, $t_I > \sqcup$, for all $H \subseteq \mathsf{A} \setminus \{t\}$ and $I \subseteq \mathsf{A}$, and giving the lexicographical status for the first argument to $\cdot$ and the lexicographical status for the second argument to $:\!\rightarrow$. Moreover, it is easy to see that each normal form is a basic term. $\Box$

Moreover, we can prove that $\mathrm{PA}_\delta^{\mathrm{cc}}$ is a conservative extension of $\mathrm{BPA}_\delta^{\mathrm{cc}}$.

Lemma 3 (Conservative extension) *If p and q are closed terms of sort* $\mathbf{P}$ *from the language of* $\mathrm{BPA}_\delta^{\mathrm{cc}}$*, then* $\mathrm{BPA}_\delta^{\mathrm{cc}} \vdash p = q$ *iff* $\mathrm{PA}_\delta^{\mathrm{cc}} \vdash p = q$.

PROOF. The implication from left to right follows immediately from the fact that the axioms of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ are included in the axioms of $\mathrm{PA}_\delta^{\mathrm{cc}}$. The implication from right to left is proved as follows. Let p and q be closed terms of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ such that $\mathrm{PA}_\delta^{\mathrm{cc}} \vdash p = q$. Because it is left-linear and non-overlapping (see e.g. Ref. [23]), the term rewriting system used in the proof of Lemma 2 is confluent modulo axioms A1–A3 and BA1–BA8. Consequently, the reductions of p and q by means of this term rewriting system yields the same normal form modulo axioms A1–A3 and BA1–BA8. Moreover, the reductions of p and q only use axioms A4–A7 and GC1–GC7, oriented from left to right, because the additional operators of $\mathrm{PA}_\delta^{\mathrm{cc}}$ do not occur in p and q, and no rewrite rule introduces occurrences of those operators that were not already there in its left-hand side. Hence, the reduction of p into its normal form followed by the reverse of the reduction of q into its normal form is a proof of $p = q$ in $\mathrm{BPA}_\delta^{\mathrm{cc}}$. $\Box$

The preceding three lemmas will be useful in the completeness proof of $\mathrm{PA}_\delta^{\mathrm{cc}}$ for the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$ that will be introduced in Section 4.

Terms of sort $\mathbf{C}$ are interpreted in $\mathcal{C}$ as usual (see e.g. Ref. [14]), the association of operations with the extra-Boolean operators ∂_H and t_I excepted. With the operator ∂_H is associated the unique endomorphism of $\mathcal{C}$ extending the function ∂_H on $\{\mathcal{E}_a \mid a \in \mathsf{A}\}$ defined by $\partial_H(\mathcal{E}_a) = \bot$ if $a \in H$ and $\partial_H(\mathcal{E}_a) = \mathcal{E}_a$ if $a \notin H$; and with the operator t_I is associated the unique endomorphism of

$\mathcal{C}$ extending the function t_I on $\{\mathcal{E}_a \mid a \in \mathsf{A}\}$ defined by $t_I(\mathcal{E}_a) = \top$ if $a \in I$ and $t_I(\mathcal{E}_a) = \mathcal{E}_a$ if $a \notin I$. It follows automatically that axioms BA1–BA8, ED1–ED7 and EI1–EI7 constitute a sound and complete axiomatization of the expansion of $\mathcal{C}$ with the operations ∂_H and t_I defined above. Henceforth, we loosely write $\mathcal{C}$ for this expansion of $\mathcal{C}$. In the above, and in subsequent sections, the constants and operators to build terms of sort $\mathbf{C}$ are also used at the meta level as names for the elements of $\mathcal{C}$ and operations on $\mathcal{C}$ associated with them.

We proceed to the presentation of the structural operational semantics of $\mathrm{PA}_\delta^{\mathrm{cc}}$. The following relations on closed terms of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$ are used:

- for each $\ell \in (\mathcal{C} \setminus \{\bot\}) \times \mathsf{A}$, a binary relation $\xrightarrow{\ell}$;
- for each $\ell \in (\mathcal{C} \setminus \{\bot\}) \times \mathsf{A}$, a unary relation $\xrightarrow{\ell} \surd$.

We write $p \xrightarrow{[\alpha]\,a} q$ instead of $(p, q) \in \xrightarrow{(\alpha,a)}$ and $p \xrightarrow{[\alpha]\,a} \surd$ instead of $p \in \xrightarrow{(\alpha,a)} \surd$. The relations $\xrightarrow{\ell} \surd$ and $\xrightarrow{\ell}$ can be explained as follows:

- $p \xrightarrow{[\alpha]\,a} \surd$: p is capable of performing action a under condition α and then terminating successfully;
- $p \xrightarrow{[\alpha]\,a} q$: p is capable of performing action a under condition α and then proceeding as q.

The structural operational semantics of $\mathrm{PA}_\delta^{\mathrm{cc}}$ is described by the transition rules given in Table 3. We will return to this structural operational semantics in Section 5.

3 Transition Systems and Splitting Bisimilarity for $\mathrm{PA}_\delta^{\mathrm{cc}}$

In this section, we introduce conditional transition systems and splitting bisimilarity of conditional transition systems. In Section 4, we will make use of conditional transition systems and splitting bisimilarity of conditional transition systems to construct models of $\mathrm{PA}_\delta^{\mathrm{cc}}$. In Section 5, we will show that the structural operational semantics presented in Section 2 induces a conditional transition system for each closed term of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$.

Conditional transition systems are labelled transition systems of which the labels consist of a condition different from $\bot$ and an action. Labels of this kind are sometimes called *guarded actions*. Henceforth, we write $\mathcal{C}^-$ for $\mathcal{C} \setminus \{\bot\}$.

A *conditional transition system* T consists of the following:

Table 3
Transition rules for $\mathrm{PA}_\delta^{\mathrm{cc}}$

$$\frac{}{a \xrightarrow{[\top]\,a} \surd}$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{x + y \xrightarrow{[\phi]\,a} \surd} \qquad \frac{y \xrightarrow{[\phi]\,a} \surd}{x + y \xrightarrow{[\phi]\,a} \surd} \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{x + y \xrightarrow{[\phi]\,a} x'} \qquad \frac{y \xrightarrow{[\phi]\,a} y'}{x + y \xrightarrow{[\phi]\,a} y'}$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{x \cdot y \xrightarrow{[\phi]\,a} y} \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{x \cdot y \xrightarrow{[\phi]\,a} x' \cdot y}$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{\psi :\rightarrow x \xrightarrow{[\phi \sqcap \psi]\,a} \surd}\ \phi \sqcap \psi \neq \bot \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{\psi :\rightarrow x \xrightarrow{[\phi \sqcap \psi]\,a} x'}\ \phi \sqcap \psi \neq \bot$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{x \parallel y \xrightarrow{[\phi]\,a} y} \qquad \frac{y \xrightarrow{[\phi]\,a} \surd}{x \parallel y \xrightarrow{[\phi]\,a} x} \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{x \parallel y \xrightarrow{[\phi]\,a} x' \parallel y} \qquad \frac{y \xrightarrow{[\phi]\,a} y'}{x \parallel y \xrightarrow{[\phi]\,a} x \parallel y'}$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{x \between y \xrightarrow{[\phi]\,a} y} \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{x \between y \xrightarrow{[\phi]\,a} x' \parallel y}$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{\partial_H(x) \xrightarrow{[\partial_H(\phi)]\,a} \surd}\ a \notin H, \partial_H(\phi) \neq \bot \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{\partial_H(x) \xrightarrow{[\partial_H(\phi)]\,a} \partial_H(x')}\ a \notin H, \partial_H(\phi) \neq \bot$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{t_I(x) \xrightarrow{[t_I(\phi)]\,a} \surd}\ a \notin I, t_I(\phi) \neq \bot \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{t_I(x) \xrightarrow{[t_I(\phi)]\,a} t_I(x')}\ a \notin I, t_I(\phi) \neq \bot$$

$$\frac{x \xrightarrow{[\phi]\,a} \surd}{t_I(x) \xrightarrow{[t_I(\phi)]\,t} \surd}\ a \in I, t_I(\phi) \neq \bot \qquad \frac{x \xrightarrow{[\phi]\,a} x'}{t_I(x) \xrightarrow{[t_I(\phi)]\,t} t_I(x')}\ a \in I, t_I(\phi) \neq \bot$$

- a set S of *states*;
- a set $\xrightarrow{\ell} \subseteq S \times S$, for each $\ell \in \mathcal{C}^- \times \mathsf{A}$;
- a set $\xrightarrow{\ell} \surd \subseteq S$, for each $\ell \in \mathcal{C}^- \times \mathsf{A}$;
- an *initial state* $s^0 \in S$.

If $(s, s') \in \xrightarrow{\ell}$ for some $\ell \in \mathcal{C}^- \times \mathsf{A}$, then we say that there is a *transition* from s to s'. We usually write $s \xrightarrow{[\alpha]\,a} s'$ instead of $(s, s') \in \xrightarrow{(\alpha,a)}$ and $s \xrightarrow{[\alpha]\,a} \surd$ instead of $s \in \xrightarrow{(\alpha,a)} \surd$. Furthermore, we write $\rightarrow$ for the family of sets $(\xrightarrow{\ell})_{\ell \in \mathcal{C}^- \times \mathsf{A}}$ and $\rightarrow \surd$ for the family of sets $(\xrightarrow{\ell} \surd)_{\ell \in \mathcal{C}^- \times \mathsf{A}}$.

The relations $\xrightarrow{\ell} \surd$ and $\xrightarrow{\ell}$ can be explained as follows:

- $s \xrightarrow{[\alpha]\,a} \surd$: in state s, it is possible to perform action a under condition α, and by doing so to terminate successfully;
- $s \xrightarrow{[\alpha]\,a} s'$: in state s, it is possible to perform action a under condition α,

and by doing so to make a transition to state s'.

A conditional transition system may have states that are not reachable from its initial state by a number of transitions. Connected conditional transition systems are transition systems without unreachable states.

Let $T = (S, \rightarrow, \rightarrow\surd, s^0)$ be a conditional transition system. Then the *reachability* relation of T is the smallest relation $\twoheadrightarrow \subseteq S \times S$ such that:

- $s \twoheadrightarrow s$;
- if $s \xrightarrow{\ell} s'$ and $s' \twoheadrightarrow s''$, then $s \twoheadrightarrow s''$.

We write $\mathrm{RS}(T)$ for $\{s \in S \mid s^0 \twoheadrightarrow s\}$. T is called a *connected* conditional transition system if $S = \mathrm{RS}(T)$. Henceforth, we only consider connected conditional transition systems. This often calls for extraction of the connected part of a conditional transition system that is composed of connected conditional transition systems.

Let $T = (S, \rightarrow, \rightarrow\surd, s^0)$ be a conditional transition system that is not necessarily connected. Then the *connected part* of T, written $\Gamma(T)$, is defined as follows:

$$\Gamma(T) = (S', \rightarrow', \rightarrow\surd', s^0) \ ,$$

where

$$S' = \mathrm{RS}(T) \ ,$$

and for every $\ell \in \mathcal{C}^- \times \mathsf{A}$:

$$\xrightarrow{\ell}{}' \ = \xrightarrow{\ell} \cap (S' \times S') \ ,$$

$$\xrightarrow{\ell}\surd' = \xrightarrow{\ell}\surd \cap S' \ .$$

It is assumed that for each infinite cardinal κ a fixed but arbitrary set $\mathcal{S}_\kappa$ with the following properties has been given:

- the cardinality of $\mathcal{S}_\kappa$ is greater than or equal to κ;
- if $S_1, S_2 \subseteq \mathcal{S}_\kappa$, then $S_1 \uplus S_2 \subseteq \mathcal{S}_\kappa$ and $S_1 \times S_2 \subseteq \mathcal{S}_\kappa$. [2]

[2] We write $A \uplus B$ for the disjoint union of sets A and B, i.e. $A \uplus B = (A \times \{\emptyset\}) \cup (B \times \{\{\emptyset\}\})$. We write μ_1 and μ_2 for the associated injections $\mu_1 : A \to A \uplus B$ and $\mu_2 : B \to A \uplus B$, defined by $\mu_1(a) = (a, \emptyset)$ and $\mu_2(b) = (b, \{\emptyset\})$.

Let κ be an infinite cardinal. Then $\mathbb{CTS}_\kappa$ is the set of all connected conditional transition systems $T = (S, \rightarrow, \rightarrow\surd, s^0)$ such that $S \subset \mathcal{S}_\kappa$ and the branching degree of T is less than κ, i.e. for all $s \in S$, the cardinality of the set $\{(\ell, s') \in (\mathcal{C}^- \times \mathsf{A}) \times S \mid (s, s') \in \xrightarrow{\ell}\} \cup \{\ell \in \mathcal{C}^- \times \mathsf{A} \mid s \in \xrightarrow{\ell}\surd\}$ is less than κ.[3]

The condition $S \subset \mathcal{S}_\kappa$ guarantees that $\mathbb{CTS}_\kappa$ is indeed a set.

A conditional transition system is said to be *finitely branching* if its branching degree is less than $\aleph_0$. Otherwise, it is said to be *infinitely branching*.

Remark 4 *We consider both finitely branching and infinitely branching conditional transition systems. Infinitely branching conditional transition systems cannot be described by means of the constants and operators of* $\mathrm{PA}^{\mathrm{cc}}_\delta$*, not even together with guarded recursion (see Section 8). Like in Ref. [1], we prefer to consider not only finitely branching conditional transition systems in order to make later generalizations possible. For example, we are semantically prepared for removing the restriction that the set* A *of actions is finite and introducing an operator for alternative composition over an infinite set of alternatives.*

Conditional transition systems that differ only with respect to the identity of the states are isomorphic.

Let $T_1 = (S_1, \rightarrow_1, \rightarrow\surd_1, s_1^0)$ and $T_2 = (S_2, \rightarrow_2, \rightarrow\surd_2, s_2^0)$ be conditional transition systems. Then T_1 and T_2 are *isomorphic*, written $T_1 \cong T_2$, if there exists a bijective function $b : S_1 \to S_2$ such that:

- $b(s_1^0) = s_2^0$;
- $s_1 \xrightarrow{\ell}_1 s_1'$ iff $b(s_1) \xrightarrow{\ell}_2 b(s_1')$;
- $s \xrightarrow{\ell}\surd_1$ iff $b(s) \xrightarrow{\ell}\surd_2$.

Henceforth, we always consider two conditional transition systems essentially the same if they are isomorphic.

Remark 5 *The set* $\mathbb{CTS}_\kappa$ *is independent of* $\mathcal{S}_\kappa$*. By that we mean the following. Let* $\mathbb{CTS}_\kappa$ *and* $\mathbb{CTS}'_\kappa$ *result from different choices for* $\mathcal{S}_\kappa$*. Then there exists a bijection* $b : \mathbb{CTS}_\kappa \to \mathbb{CTS}'_\kappa$ *such that for all* $T \in \mathbb{CTS}_\kappa$*,* $T \cong b(T)$*.*

Bisimilarity has to be adapted to the setting with guarded actions. In the definition given below, we use a well-known notion from the field of Boolean algebra: the partial order relation $\sqsubseteq$ on $\mathcal{C}$ defined by

$$\alpha \sqsubseteq \beta \text{ iff } \alpha \sqcup \beta = \beta \,.$$

[3] In Ref. [1], the definition of $\mathbb{CTS}_\kappa$ is given for an arbitrary set of *atomic conditions*. In the case where the set $\{\mathcal{E}_a \mid a \in \mathsf{A}\}$ is taken as the set of atomic conditions, that definition and the definition given here are essentially the same.

Moreover, we use the notation $\bigsqcup A$, where $A = \{\alpha_1, \ldots, \alpha_n\} \subseteq \mathcal{C}$, for $\alpha_1 \sqcup \ldots \sqcup \alpha_n$.

Let $T_1 = (S_1, \rightarrow_1, \rightarrow\surd_1, s_1^0) \in \mathbb{CTTS}_\kappa$ and $T_2 = (S_2, \rightarrow_2, \rightarrow\surd_2, s_2^0) \in \mathbb{CTTS}_\kappa$ (for an infinite cardinal κ). Then a *splitting bisimulation* B between T_1 and T_2 is a binary relation $B \subseteq S_1 \times S_2$ such that $B(s_1^0, s_2^0)$ and for all s_1, s_2 such that $B(s_1, s_2)$:

- if $s_1 \xrightarrow{[\alpha]\,a}_1 s_1'$, then there is a set $CS_2' \subseteq \mathcal{C}^- \times S_2$ such that $\alpha \sqsubseteq \bigsqcup \mathrm{dom}(CS_2')$ and for all $(\alpha', s_2') \in CS_2'$, $s_2 \xrightarrow{[\alpha']\,a}_2 s_2'$ and $B(s_1', s_2')$;[4]
- if $s_2 \xrightarrow{[\alpha]\,a}_2 s_2'$, then there is a set $CS_1' \subseteq \mathcal{C}^- \times S_1$ such that $\alpha \sqsubseteq \bigsqcup \mathrm{dom}(CS_1')$ and for all $(\alpha', s_1') \in CS_1'$, $s_1 \xrightarrow{[\alpha']\,a}_1 s_1'$ and $B(s_1', s_2')$;
- if $s_1 \xrightarrow{[\alpha]\,a} \surd_1$, then there is a set $C' \subseteq \mathcal{C}^-$ such that $\alpha \sqsubseteq \bigsqcup C'$ and for all $\alpha' \in C'$, $s_2 \xrightarrow{[\alpha']\,a} \surd_2$;
- if $s_2 \xrightarrow{[\alpha]\,a} \surd_2$, then there is a set $C' \subseteq \mathcal{C}^-$ such that $\alpha \sqsubseteq \bigsqcup C'$ and for all $\alpha' \in C'$, $s_1 \xrightarrow{[\alpha']\,a} \surd_1$.

Two conditional transition systems $T_1, T_2 \in \mathbb{CTTS}_\kappa$ are *splitting bisimilar*, written $T_1 \leftrightarroweq T_2$, if there exists a splitting bisimulation B between T_1 and T_2. Let B be a splitting bisimulation between T_1 and T_2. Then we say that B is a splitting bisimulation *witnessing* $T_1 \leftrightarroweq T_2$.

It is easy to see that $\leftrightarroweq$ is an equivalence on $\mathbb{CTTS}_\kappa$. Let $T \in \mathbb{CTTS}_\kappa$. Then we write $[\,T\,]_{\leftrightarroweq}$ for $\{T' \in \mathbb{CTTS}_\kappa \mid T \leftrightarroweq T'\}$, i.e. the $\leftrightarroweq$-equivalence class of T. We write $\mathbb{CTTS}_\kappa/\!\leftrightarroweq$ for the set of equivalence classes $\{[\,T\,]_{\leftrightarroweq} \mid T \in \mathbb{CTTS}_\kappa\}$.

In Section 4, we will use $\mathbb{CTTS}_\kappa/\!\leftrightarroweq$ as domain of a structure that is a model of $\mathrm{PA}_\delta^{\mathrm{cc}}$. As domain of a structure, $\mathbb{CTTS}_\kappa/\!\leftrightarroweq$ must be a set. That is the case because $\mathbb{CTTS}_\kappa$ is a set. The latter is guaranteed by considering only conditional transition systems of which the set of states is a subset of $\mathcal{S}_\kappa$.

Remark 6 *The question arises whether $\mathcal{S}_\kappa$ is large enough if its cardinality is greater than or equal to κ. This question can be answered in the affirmative. Let $T = (S, \rightarrow, \rightarrow\surd, s^0)$ be a connected conditional transition system of which the branching degree is less than κ. Then there exists a connected conditional transition system $T' = (S', \rightarrow', \rightarrow\surd', s^{0\prime})$ of which the branching degree is less than κ such that $T \leftrightarroweq T'$ and the cardinality of S' is less than κ.*

It is easy to see that, if we would consider conditional transition systems with unreachable states as well, each conditional transition system would be splitting bisimilar to its connected part. This justifies the choice to consider

[4] If for some sets A and B, $R \subseteq A \times B$, then we write $\mathrm{dom}(R)$ for the domain of the relation R, i.e. $\mathrm{dom}(R) = \{a \in A \mid \exists b \in B \bullet (a, b) \in R\}$.

only connected conditional transition systems. It is easy to see that isomorphic conditional transition systems are splitting bisimilar. This justifies the choice to consider conditional transition systems essentially the same if they are isomorphic.

The name splitting bisimulation is used because a transition of one of the related transition systems may be simulated by a set of transitions of the other transition system. Splitting bisimulation should not be confused with split bisimulation [25]. We think that splitting bisimulation can be reformulated in a style that is similar to the style in which probabilistic bisimulation is formulated in Ref. [26]. We refrain from such a reformulation because it would require the introduction of various auxiliary notions and notations. At the end of this section, we sketch how splitting bisimilarity is related to ordinary bisimilarity.

Remark 7 *At first sight, splitting bisimulation resembles (early and late) symbolic bisimulation as introduced in Ref. [27]. However, the resemblance is incidental. Symbolic bisimulation concerns symbolic transition systems. The states of a symbolic transition system play the role of open terms and the transitions are labelled by "symbolic guarded actions" which evaluate to guarded actions, with $\top$ or $\bot$ as condition, for each assignment of values to free variables. Thus, the concrete transitions from a state are assignment dependent. Splitting bisimulation concerns conditional transition systems in which variables ranging over values play no part at all. The differences between standard bisimulation and splitting bisimulation find their origin in the fact that a transition of one of the related conditional transition systems may be simulated by a set of transitions of the other conditional transition system. The differences between standard bisimulation and symbolic bisimulation find their origin in the fact that a state of one of the related symbolic transition systems may correspond to different states of the other symbolic transition system under different assignments of values to free variables. Taking these different origins into account, it is surprising that splitting bisimulation resembles symbolic bisimulation at first sight. However, the resemblance is disturbed by the following difference between splitting bisimulation and symbolic bisimulation: splitting bisimulation is a binary relation on states, whereas symbolic bisimulation is a family of binary relation on states indexed by a set of Boolean expressions. Being such indexed families of relations happens to be the essence of symbolic bisimulations.*

In the remainder of this section, we sketch how splitting bisimilarity is related to ordinary bisimilarity.

Let $T = (S, \rightarrow, \rightarrow\!\!\surd, s^0) \in \mathbb{CTS}_\kappa$ (for an infinite cardinal κ). We write $\simeq_T$ for the maximal splitting bisimulation witnessing $T \leftrightarroweq T$ (such a relation always exists). It is easy to see that $\simeq_T$ is an equivalence relation on S. It identifies

states of T that can simulate the conditional transitions of each other. The *condition-normal form* of T, written $\mathrm{CN}(T)$, is defined as follows:

$$\mathrm{CN}(T) = (S, \rightarrow', \rightarrow\surd', s^0) \ ,$$

where for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\begin{aligned}
\xrightarrow{(\alpha,a)}{}' \quad &= \Big\{(s,s') \;\Big|\; \exists \beta \bullet s \xrightarrow{[\beta]\,a} s' \wedge \\
&\qquad \alpha = \bigsqcup\Big\{\beta' \;\Big|\; \exists s'' \bullet \Big(s' \simeq_T s'' \wedge s \xrightarrow{[\beta']\,a} s''\Big)\Big\}\Big\} \ , \\
\xrightarrow{(\alpha,a)}\surd' &= \Big\{s \;\Big|\; \exists \beta \bullet s \xrightarrow{[\beta]\,a}\surd \wedge \alpha = \bigsqcup\Big\{\beta' \;\Big|\; s \xrightarrow{[\beta']\,a}\surd\Big\}\Big\} \ .
\end{aligned}$$

It is easy to see that $\mathrm{CN}(T) \in \mathbb{CTTS}_\kappa$ and $T \leftrightarrows \mathrm{CN}(T)$. We have $T = \mathrm{CN}(T)$ iff T has the following properties:

- if $s_1 \xrightarrow{[\alpha]\,a} s_1'$, $s_1 \xrightarrow{[\beta]\,a} s_1''$ and $s_1' \simeq_T s_1''$, then $\alpha = \beta$;
- if $s_1 \xrightarrow{[\alpha]\,a}\surd$ and $s_1 \xrightarrow{[\beta]\,a}\surd$, then $\alpha = \beta$.

We say that T is *condition-normal* if $T = \mathrm{CN}(T)$.

Let $T_1 = (S_1, \rightarrow_1, \rightarrow\surd_1, s_1^0) \in \mathbb{CTTS}_\kappa$ and $T_2 = (S_2, \rightarrow_2, \rightarrow\surd_2, s_2^0) \in \mathbb{CTTS}_\kappa$ (for an infinite cardinal κ). Then a *bisimulation* B between T_1 and T_2 is a binary relation $B \subseteq S_1 \times S_2$ such that $B(s_1^0, s_2^0)$ and for all s_1, s_2 such that $B(s_1, s_2)$:

- if $s_1 \xrightarrow{\ell}_1 s_1'$, then there is a $s_2' \in S_2$ such that $s_2 \xrightarrow{\ell}_2 s_2'$ and $B(s_1', s_2')$;
- if $s_2 \xrightarrow{\ell}_2 s_2'$, then there is a $s_1' \in S_1$ such that $s_1 \xrightarrow{\ell}_1 s_1'$ and $B(s_1', s_2')$;
- $s_1 \xrightarrow{\ell}\surd_1$ iff $s_2 \xrightarrow{\ell}\surd_2$.

Two conditional transition systems $T_1, T_2 \in \mathbb{CTTS}_\kappa$ are *bisimilar*, written $T_1 \leftrightarrows T_2$, if there exists a bisimulation B between T_1 and T_2. We have $\mathrm{CN}(T_1) \leftrightarrows \mathrm{CN}(T_2)$ iff $\mathrm{CN}(T_1) \leftrightarrows \mathrm{CN}(T_2)$. So, splitting bisimilarity and ordinary bisimilarity coincide on condition-normal conditional transition systems. It is worth mentioning that we do not have this result if we replace $s' \simeq_T s''$ by $s' = s''$ in the definition of CN.

4 Full Splitting Bisimulation Models of $\mathrm{PA}_\delta^{\mathrm{cc}}$

In this section, we introduce the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$. They are models in which equivalence classes of conditional transition systems modulo splitting bisimilarity are taken as processes. The qualification

"full" originates from Ref. [28]. It expresses that there exist other splitting bisimulation models, but each of them is isomorphically embedded in a full splitting bisimulation model.

There is a full splitting bisimulation model of $\mathrm{PA}_\delta^{\mathrm{cc}}$ for each infinite cardinal. To obtain the full splitting bisimulation model of $\mathrm{PA}_\delta^{\mathrm{cc}}$ for a fixed infinite cardinal κ, we associate the set $\mathbb{CTS}_\kappa/\leftrightarrows$ with the sort $\mathbf{P}$, an element of $\mathbb{CTS}_\kappa/\leftrightarrows$ with each of the constants δ and a ($a \in \mathsf{A}$), and an operation on $\mathbb{CTS}_\kappa/\leftrightarrows$ with each of the operators $+$, $\cdot$, $:\!\!\rightarrow$, $\parallel$, $\between$, ∂_H ($H \subseteq \mathsf{A} \setminus \{t\}$) and t_I ($I \subseteq \mathsf{A}$).[5] We begin by associating elements of $\mathbb{CTS}_\kappa$ and operations on $\mathbb{CTS}_\kappa$ with these constants and operators. The result of this is subsequently lifted to $\mathbb{CTS}_\kappa/\leftrightarrows$.

It is assumed that for each infinite cardinal κ a fixed but arbitrary choice function $\mathrm{ch}_\kappa : (\mathcal{P}(\mathcal{S}_\kappa) \setminus \emptyset) \to \mathcal{S}_\kappa$ such that for all $S \in \mathcal{P}(\mathcal{S}_\kappa) \setminus \emptyset$, $\mathrm{ch}_\kappa(S) \in S$ has been given. The function ch_κ is used whenever there is a need to get a fresh state from $\mathcal{S}_\kappa$.

We will use the abbreviation $s \xrightarrow{a} s' \between s''$ for $s \xrightarrow{a} s' \vee (s \xrightarrow{a} \surd \wedge s' = s'')$. Usually, s'' is a state that takes the place of s' in the case of termination. This is useful where termination has to be turned into a state, as with parallel composition of conditional transition systems.

We associate with each constant c mentioned above an element $\widehat{c}$ of $\mathbb{CTS}_\kappa$ and with each operator f mentioned above an operation $\widehat{f}$ on $\mathbb{CTS}_\kappa$ as follows.

- $\widehat{\delta} = (\{s^0\}, \emptyset, \emptyset, s^0)$,
  where

  $s^0 = \mathrm{ch}_\kappa(\mathcal{S}_\kappa)$.

- $\widehat{a} = (\{s^0\}, \emptyset, \rightarrow\surd, s^0)$,
  where

  $s^0 = \mathrm{ch}_\kappa(\mathcal{S}_\kappa)$,
  $\xrightarrow{(\top,a)}\surd = \{s^0\}$,

  and for every $(\alpha, a') \in (\mathcal{C}^- \times \mathsf{A}) \setminus \{(\top, a)\}$:

  $\xrightarrow{(\alpha,a')}\surd = \emptyset$.

[5] In this paper, we loosely include the operation associated with the operator $:\!\!\rightarrow$ in the operations on $\mathbb{CTS}_\kappa/\leftrightarrows$. Actually, it is an operation from $\mathcal{C} \times \mathbb{CTS}_\kappa/\leftrightarrows$ to $\mathbb{CTS}_\kappa/\leftrightarrows$.

- Let $T_i = (S_i, \rightarrow_i, \rightarrow\surd_i, s_i^0) \in \mathbb{CTTS}_\kappa$ for $i = 1, 2$. Then

$$T_1 \mathbin{\widehat{+}} T_2 = \Gamma(S, \rightarrow, \rightarrow\surd, s^0) \ ,$$

where

$$\begin{aligned} s^0 &= \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1 \uplus S_2)) \ , \\ S &= \{s^0\} \cup (S_1 \uplus S_2) \ , \end{aligned}$$

and for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\begin{aligned} \xrightarrow{(\alpha,a)} &= \left\{(s^0, \mu_1(s)) \;\middle|\; s_1^0 \xrightarrow{[\alpha]\,a}_1 s\right\} \\ &\cup \left\{(s^0, \mu_2(s)) \;\middle|\; s_2^0 \xrightarrow{[\alpha]\,a}_2 s\right\} \\ &\cup \left\{(\mu_1(s), \mu_1(s')) \;\middle|\; s \xrightarrow{[\alpha]\,a}_1 s'\right\} \\ &\cup \left\{(\mu_2(s), \mu_2(s')) \;\middle|\; s \xrightarrow{[\alpha]\,a}_2 s'\right\} , \\ \xrightarrow{(\alpha,a)}\surd &= \left\{s^0 \;\middle|\; s_1^0 \xrightarrow{[\alpha]\,a} \surd_1\right\} \\ &\cup \left\{s^0 \;\middle|\; s_2^0 \xrightarrow{[\alpha]\,a} \surd_2\right\} \\ &\cup \left\{\mu_1(s) \;\middle|\; s \xrightarrow{[\alpha]\,a} \surd_1\right\} \\ &\cup \left\{\mu_2(s) \;\middle|\; s \xrightarrow{[\alpha]\,a} \surd_2\right\} . \end{aligned}$$

- Let $T_i = (S_i, \rightarrow_i, \rightarrow\surd_i, s_i^0) \in \mathbb{CTTS}_\kappa$ for $i = 1, 2$. Then

$$T_1 \mathbin{\widehat{\cdot}} T_2 = \Gamma(S, \rightarrow, \rightarrow\surd, s^0) \ ,$$

where

$$\begin{aligned} S &= S_1 \uplus S_2 \ , \\ s^0 &= \mu_1(s_1^0) \ , \end{aligned}$$

and for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\begin{aligned} \xrightarrow{(\alpha,a)} &= \left\{(\mu_1(s), \mu_1(s')) \;\middle|\; s \xrightarrow{[\alpha]\,a}_1 s'\right\} \\ &\cup \left\{(\mu_1(s), \mu_2(s_2^0)) \;\middle|\; s \xrightarrow{[\alpha]\,a} \surd_1\right\} \\ &\cup \left\{(\mu_2(s), \mu_2(s')) \;\middle|\; s \xrightarrow{[\alpha]\,a}_2 s'\right\} , \\ \xrightarrow{(\alpha,a)}\surd &= \left\{\mu_2(s) \;\middle|\; s \xrightarrow{[\alpha]\,a} \surd_2\right\} . \end{aligned}$$

- Let $\alpha \in \mathcal{C}$ and $T = (S, \rightarrow, \rightarrow\surd, s^0) \in \mathbb{CTTS}_\kappa$. Then

$$\alpha \mathbin{\widehat{:\!\rightarrow}} T = \Gamma(S, \rightarrow', \rightarrow\surd', s^0) \ ,$$

where for every $(\alpha', a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\begin{aligned}\xrightarrow{(\alpha',a)}{}' &= \left\{(s^0, s') \;\middle|\; \exists \beta \bullet \left(s^0 \xrightarrow{[\beta]\, a} s' \wedge \alpha' = \alpha \sqcap \beta\right)\right\} \\ &\quad \cup \left\{(s, s') \;\middle|\; s \xrightarrow{[\alpha']\, a} s' \wedge s \neq s^0\right\} , \\ \xrightarrow{(\alpha',a)}\surd' &= \left\{s^0 \;\middle|\; \exists \beta \bullet \left(s^0 \xrightarrow{[\beta]\, a} \surd \wedge \alpha' = \alpha \sqcap \beta\right)\right\} \\ &\quad \cup \left\{s \;\middle|\; s \xrightarrow{[\alpha']\, a} \surd \wedge s \neq s^0\right\} .\end{aligned}$$

- Let $T_i = (S_i, \rightarrow_i, \rightarrow\surd_i, s_i^0) \in \mathbb{CTTS}_\kappa$ for $i = 1, 2$. Then

$$T_1 \mathbin{\widehat{\|}} T_2 = (S, \rightarrow, \rightarrow\surd, s^0) ,$$

where

$$\begin{aligned}s^0 &= (s_1^0, s_2^0) , \\ s^\surd &= \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1 \cup S_2)) , \\ S &= ((S_1 \cup \{s^\surd\}) \times (S_2 \cup \{s^\surd\})) \setminus \{(s^\surd, s^\surd)\} ,\end{aligned}$$

and for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\begin{aligned}\xrightarrow{(\alpha,a)} &= \left\{((s_1, s_2), (s_1', s_2)) \;\middle|\; (s_1', s_2) \in S \wedge s_1 \xrightarrow{[\alpha]\, a}_1 s_1' \wr s^\surd\right\} \\ &\quad \cup \left\{((s_1, s_2), (s_1, s_2')) \;\middle|\; (s_1, s_2') \in S \wedge s_2 \xrightarrow{[\alpha]\, a}_2 s_2' \wr s^\surd\right\} , \\ \xrightarrow{(\alpha,a)}\surd &= \left\{(s_1, s^\surd) \;\middle|\; s_1 \xrightarrow{[\alpha]\, a} \surd_1\right\} \\ &\quad \cup \left\{(s^\surd, s_2) \;\middle|\; s_2 \xrightarrow{[\alpha]\, a} \surd_2\right\} .\end{aligned}$$

- Let $T_i = (S_i, \rightarrow_i, \rightarrow\surd_i, s_i^0) \in \mathbb{CTTS}_\kappa$ for $i = 1, 2$. Suppose that $T_1 \mathbin{\widehat{\|}} T_2 = (S, \rightarrow, \rightarrow\surd, s^0)$ where $S = ((S_1 \cup \{s^\surd\}) \times (S_2 \cup \{s^\surd\})) \setminus \{(s^\surd, s^\surd)\}$ and $s^\surd = \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1 \cup S_2))$. Then

$$T_1 \mathbin{\widehat{\|\!|}} T_2 = \Gamma(S', \rightarrow', \rightarrow\surd, s^{0\prime}) ,$$

where

$$\begin{aligned}s^{0\prime} &= \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus S) , \\ S' &= \{s^{0\prime}\} \cup S ,\end{aligned}$$

and for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\xrightarrow{(\alpha,a)}{}' = \left\{(s^{0\prime}, (s, s_2^0)) \;\middle|\; s_1^0 \xrightarrow{[\alpha]\, a}_1 s \wr s^\surd\right\} \cup \xrightarrow{(\alpha,a)} .$$

- Let $T = (S, \rightarrow, \rightarrow\surd, s^0) \in \mathbb{CTTS}_\kappa$. Then

  $$\widehat{\partial_H}(T) = \Gamma(S, \rightarrow', \rightarrow\surd', s^0) \ ,$$

  where for every $(\alpha, a) \in \mathcal{C}^- \times (\mathsf{A} \setminus H)$:

  $$\begin{aligned} \xrightarrow{(\alpha,a)}{}' &= \Big\{(s, s') \;\Big|\; \exists \alpha' \bullet \Big(s \xrightarrow{[\alpha']\, a} s' \wedge \alpha = \partial_H(\alpha')\Big)\Big\} \ , \\ \xrightarrow{(\alpha,a)}\surd' &= \Big\{s \;\Big|\; \exists \alpha' \bullet \Big(s \xrightarrow{[\alpha']\, a} \surd \wedge \alpha = \partial_H(\alpha')\Big)\Big\} \ , \end{aligned}$$

  and for every $(\alpha, a) \in \mathcal{C}^- \times H$:

  $$\begin{aligned} \xrightarrow{(\alpha,a)}{}' &= \emptyset \ , \\ \xrightarrow{(\alpha,a)}\surd' &= \emptyset \ . \end{aligned}$$

- Let $T = (S, \rightarrow, \rightarrow\surd, s^0) \in \mathbb{CTTS}_\kappa$. Then

  $$\widehat{t_I}\,(T) = \Gamma(S, \rightarrow', \rightarrow\surd', s^0) \ ,$$

  where for every $(\alpha, a) \in \mathcal{C}^- \times ((\mathsf{A} \setminus I) \setminus \{t\})$:

  $$\begin{aligned} \xrightarrow{(\alpha,a)}{}' &= \Big\{(s, s') \;\Big|\; \exists \alpha' \bullet \Big(s \xrightarrow{[\alpha']\, a} s' \wedge \alpha = t_I(\alpha')\Big)\Big\} \ , \\ \xrightarrow{(\alpha,a)}\surd' &= \Big\{s \;\Big|\; \exists \alpha' \bullet \Big(s \xrightarrow{[\alpha']\, a} \surd \wedge \alpha = t_I(\alpha')\Big)\Big\} \ , \end{aligned}$$

  and for every $\alpha \in \mathcal{C}^-$:

  $$\begin{aligned} \xrightarrow{(\alpha,t)}{}' &= \Big\{(s, s') \;\Big|\; \exists \alpha', a \bullet \Big(s \xrightarrow{[\alpha']\, a} s' \wedge a \in I \cup \{t\} \wedge \alpha = t_I(\alpha')\Big)\Big\} \ , \\ \xrightarrow{(\alpha,t)}\surd' &= \Big\{s \;\Big|\; \exists \alpha', a \bullet \Big(s \xrightarrow{[\alpha']\, a} \surd \wedge a \in I \cup \{t\} \wedge \alpha = t_I(\alpha')\Big)\Big\} \ , \end{aligned}$$

  and for every $(\alpha, a) \in \mathcal{C}^- \times (I \setminus \{t\})$:

  $$\begin{aligned} \xrightarrow{(\alpha,a)}{}' &= \emptyset \ , \\ \xrightarrow{(\alpha,a)}\surd' &= \emptyset \ . \end{aligned}$$

In the definition of alternative composition on $\mathbb{CTTS}_\kappa$, the connected part of a conditional transition system is extracted because the initial states of the conditional transition systems T_1 and T_2 may be unreachable from the new initial state. The new initial state is introduced because, in T_1 and/or T_2, there may exist a transition back to the initial state. In the definition of sequential composition on $\mathbb{CTTS}_\kappa$, the connected part of a conditional transition system

is extracted because the initial state of the conditional transition system T_2 may be unreachable from the initial state of the conditional transition system T_1, due to absence of termination in T_1.

Remark 8 *The elements of $\mathbb{CTTS}_\kappa$ and the operations on $\mathbb{CTTS}_\kappa$ defined above are independent of ch_κ. Different choices for ch_κ lead for each constant to isomorphic elements of $\mathbb{CTTS}_\kappa$ and lead for each operator to operations on $\mathbb{CTTS}_\kappa$ with isomorphic results.*

We can easily show that splitting bisimilarity is a congruence with respect to alternative composition, sequential composition, guarded command, parallel composition, left merge, encapsulation and pre-abstraction.

Lemma 9 (Congruence) *Let κ be an infinite cardinal. Then for all $T_1, T_2, T_1', T_2' \in \mathbb{CTTS}_\kappa$ and $\alpha \in \mathcal{C}$, $T_1 \leftrightarroweq T_1'$ and $T_2 \leftrightarroweq T_2'$ imply $T_1 \mathbin{\widehat{+}} T_2 \leftrightarroweq T_1' \mathbin{\widehat{+}} T_2'$, $T_1 \mathbin{\widehat{\cdot}} T_2 \leftrightarroweq T_1' \mathbin{\widehat{\cdot}} T_2'$, $\alpha \mathbin{\widehat{:\!\!\rightarrow}} T_1 \leftrightarroweq \alpha \mathbin{\widehat{:\!\!\rightarrow}} T_1'$, $T_1 \mathbin{\widehat{\parallel}} T_2 \leftrightarroweq T_1' \mathbin{\widehat{\parallel}} T_2'$, $T_1 \mathbin{\widehat{\between}} T_2 \leftrightarroweq T_1' \mathbin{\widehat{\between}} T_2'$, $\widehat{\partial_H}(T_1) \leftrightarroweq \widehat{\partial_H}(T_1')$ and $\widehat{t_I}(T_1) \leftrightarroweq \widehat{t_I}(T_1')$.*

PROOF. Let $T_i = (S_i, \rightarrow_i, \rightarrow\!\!\surd_i, s_i^0)$ and $T_i' = (S_i', \rightarrow_i', \rightarrow\!\!\surd_i', s_i^{0\prime})$ for $i = 1, 2$. Let R_1 and R_2 be splitting bisimulations witnessing $T_1 \leftrightarroweq T_1'$ and $T_2 \leftrightarroweq T_2'$, respectively. Then we construct relations $R_{\widehat{+}}$, $R_{\widehat{\cdot}}$, $R_{\widehat{:\!\!\rightarrow}}$, $R_{\widehat{\parallel}}$, $R_{\widehat{\between}}$, $R_{\widehat{\partial_H}}$ and $R_{\widehat{t_I}}$, as follows:

- $R_{\widehat{+}} = (\{(s^0, s^{0\prime})\} \cup \mu_1(R_1) \cup \mu_2(R_2)) \cap (S \times S')$, where S and S' are the sets of states of $T_1 \mathbin{\widehat{+}} T_2$ and $T_1' \mathbin{\widehat{+}} T_2'$, respectively, and s^0 and $s^{0\prime}$ are the initial states of $T_1 \mathbin{\widehat{+}} T_2$ and $T_1' \mathbin{\widehat{+}} T_2'$, respectively;
- $R_{\widehat{\cdot}} = (\mu_1(R_1) \cup \mu_2(R_2)) \cap (S \times S')$, where S and S' are the sets of states of $T_1 \mathbin{\widehat{\cdot}} T_2$ and $T_1' \mathbin{\widehat{\cdot}} T_2'$, respectively;
- $R_{\widehat{:\!\!\rightarrow}} = R_1 \cap (S \times S')$, where S and S' are the sets of states of $\alpha \mathbin{\widehat{:\!\!\rightarrow}} T_1$ and $\alpha \mathbin{\widehat{:\!\!\rightarrow}} T_1'$, respectively;
- $R_{\widehat{\parallel}} = \{((s_1, s_2), (s_1', s_2')) \in S \times S' \mid (s_1, s_1') \in R_1 \cup R^{\surd}, (s_2, s_2') \in R_2 \cup R^{\surd}\}$, where S and S' are the sets of states of $T_1 \mathbin{\widehat{\parallel}} T_2$ and $T_1' \mathbin{\widehat{\parallel}} T_2'$, respectively, and $R^{\surd} = \{(\mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1 \cup S_2)), \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1' \cup S_2')))\}$;
- $R_{\widehat{\between}} = (\{(s^0, s^{0\prime})\} \cup R_{\widehat{\parallel}}) \cap (S \times S')$, where S and S' are the sets of states of $T_1 \mathbin{\widehat{\between}} T_2$ and $T_1' \mathbin{\widehat{\between}} T_2'$, respectively, and s^0 and $s^{0\prime}$ are the initial states of $T_1 \mathbin{\widehat{\between}} T_2$ and $T_1' \mathbin{\widehat{\between}} T_2'$, respectively;
- $R_{\widehat{\partial_H}} = R_1 \cap (S \times S')$, where S and S' are the sets of states of $\widehat{\partial_H}(T_1)$ and $\widehat{\partial_H}(T_1')$, respectively;
- $R_{\widehat{t_I}} = R_1 \cap (S \times S')$, where S and S' are the sets of states of $\widehat{t_I}(T_1)$ and $\widehat{t_I}(T_1')$, respectively.

Here, we write $\mu_i(R_i)$ for $\{(\mu_i(s), \mu_i(s')) \mid R_i(s, s')\}$, where μ_i is used to denote both the injection of S_i into $S_1 \uplus S_2$ and the injection of S_i' into $S_1' \uplus S_2'$. Given

the definitions of alternative composition, sequential composition, guarded command, parallel composition, left merge, encapsulation and pre-abstraction, it is easy to see that $R_{\widehat{+}}$, $R_{\widehat{\cdot}}$, $R_{\widehat{:\rightarrow}}$, $R_{\widehat{\parallel}}$, $R_{\widehat{\between}}$, $R_{\widehat{\partial_H}}$ and $R_{\widehat{t_I}}$ are splitting bisimulations witnessing $T_1 \mathbin{\widehat{+}} T_2 \leftrightarrows T_1' \mathbin{\widehat{+}} T_2'$, $T_1 \mathbin{\widehat{\cdot}} T_2 \leftrightarrows T_1' \mathbin{\widehat{\cdot}} T_2'$, $\alpha \mathbin{\widehat{:\rightarrow}} T_1 \leftrightarrows \alpha \mathbin{\widehat{:\rightarrow}} T_1'$, $T_1 \mathbin{\widehat{\parallel}} T_2 \leftrightarrows T_1' \mathbin{\widehat{\parallel}} T_2'$, $T_1 \mathbin{\widehat{\between}} T_2 \leftrightarrows T_1' \mathbin{\widehat{\between}} T_2'$, $\widehat{\partial_H}(T_1) \leftrightarrows \widehat{\partial_H}(T_1')$ and $\widehat{t_I}(T_1) \leftrightarrows \widehat{t_I}(T_1')$, respectively. $\Box$

The *full splitting bisimulation models* $\mathfrak{P}^{\mathrm{cc}}_\kappa$, one for each infinite cardinal κ, are the expansions of $\mathcal{C}$ with:[6]

- for the sort $\mathbf{P}$, a non-empty set $\mathcal{P}$;
- for the constant δ, an element $\widetilde{\delta}$ of $\mathcal{P}$;
- for each constant a ($a \in \mathsf{A}$), an element $\widetilde{a}$ of $\mathcal{P}$;
- for the operator $+$, an operation $\mathbin{\widetilde{+}} : \mathcal{P} \times \mathcal{P} \to \mathcal{P}$;
- for the operator $\cdot$, an operation $\mathbin{\widetilde{\cdot}} : \mathcal{P} \times \mathcal{P} \to \mathcal{P}$;
- for the operator $:\!\rightarrow$, an operation $\widetilde{:\rightarrow} : \mathcal{C} \times \mathcal{P} \to \mathcal{P}$;
- for the operator $\parallel$, an operation $\mathbin{\widetilde{\parallel}} : \mathcal{P} \times \mathcal{P} \to \mathcal{P}$;
- for the operator $\between$, an operation $\mathbin{\widetilde{\between}} : \mathcal{P} \times \mathcal{P} \to \mathcal{P}$;
- for each operator ∂_H ($H \subseteq \mathsf{A} \setminus \{t\}$), an operation $\widetilde{\partial_H} : \mathcal{P} \to \mathcal{P}$;
- for each operator t_I ($I \subseteq \mathsf{A}$), an operation $\widetilde{t_I} : \mathcal{P} \to \mathcal{P}$;

where those ingredients are defined as follows:

$$\begin{aligned}
\mathcal{P} &= \mathbb{CTS}_\kappa/\leftrightarrows\,, & \alpha \mathbin{\widetilde{:\rightarrow}} [\,T_1\,]_{\leftrightarrows} &= [\,\alpha \mathbin{\widehat{:\rightarrow}} T_1\,]_{\leftrightarrows}\,,\\
\widetilde{\delta} &= [\,\widehat{\delta}\,]_{\leftrightarrows}\,, & [\,T_1\,]_{\leftrightarrows} \mathbin{\widetilde{\parallel}} [\,T_2\,]_{\leftrightarrows} &= [\,T_1 \mathbin{\widehat{\parallel}} T_2\,]_{\leftrightarrows}\,,\\
\widetilde{a} &= [\,\widehat{a}\,]_{\leftrightarrows}\,, & [\,T_1\,]_{\leftrightarrows} \mathbin{\widetilde{\between}} [\,T_2\,]_{\leftrightarrows} &= [\,T_1 \mathbin{\widehat{\between}} T_2\,]_{\leftrightarrows}\,,\\
[\,T_1\,]_{\leftrightarrows} \mathbin{\widetilde{+}} [\,T_2\,]_{\leftrightarrows} &= [\,T_1 \mathbin{\widehat{+}} T_2\,]_{\leftrightarrows}\,, & \widetilde{\partial_H}([\,T_1\,]_{\leftrightarrows}) &= [\,\widehat{\partial_H}(T_1)\,]_{\leftrightarrows}\,,\\
[\,T_1\,]_{\leftrightarrows} \mathbin{\widetilde{\cdot}} [\,T_2\,]_{\leftrightarrows} &= [\,T_1 \mathbin{\widehat{\cdot}} T_2\,]_{\leftrightarrows}\,, & \widetilde{t_I}([\,T_1\,]_{\leftrightarrows}) &= [\,\widehat{t_I}(T_1)\,]_{\leftrightarrows}\,.
\end{aligned}$$

Alternative composition, sequential composition, guarded command, parallel composition, left merge, encapsulation and pre-abstraction on $\mathbb{CTS}_\kappa/\leftrightarrows$ are well-defined because $\leftrightarrows$ is a congruence with respect to the corresponding operations on $\mathbb{CTS}_\kappa$.

The structures $\mathfrak{P}^{\mathrm{cc}}_\kappa$ are models of $\mathrm{PA}^{\mathrm{cc}}_\delta$.

[6] Here, the expansions involve the addition of a domain because they go from a one-sorted algebra to a two-sorted algebra.

Theorem 10 (Soundness) *For each infinite cardinal κ, we have $\mathfrak{P}^{\rm cc}_\kappa \models {\rm PA}^{\rm cc}_\delta$.*

PROOF. Because $\mathfrak{P}^{\rm cc}_\kappa$ is an expansion of $\mathcal{C}$, it is not necessary to show that the axioms of BA are sound. The soundness of all remaining axioms follows easily from the definitions of the ingredients of $\mathfrak{P}^{\rm cc}_\kappa$. $\Box$

The axioms of ${\rm PA}^{\rm cc}_\delta$ constitute a complete axiomatization of the full splitting bisimulation models.

Theorem 11 (Completeness) *Let κ be an infinite cardinal. Then we have, for all closed terms p and q of sort $\mathbf{P}$ from the language of ${\rm PA}^{\rm cc}_\delta$, $\mathfrak{P}^{\rm cc}_\kappa \models p = q$ implies ${\rm PA}^{\rm cc}_\delta \vdash p = q$.*

PROOF. By Lemma 2, for each closed term p of sort $\mathbf{P}$ from the language of ${\rm PA}^{\rm cc}_\delta$, there is a closed term q of sort $\mathbf{P}$ from the language of ${\rm BPA}^{\rm cc}_\delta$ such that $p = q$ is derivable from the axioms of ${\rm PA}^{\rm cc}_\delta$; and by Lemma 3, all equations between closed terms of sort $\mathbf{P}$ from the language of ${\rm BPA}^{\rm cc}_\delta$ that can be derived from the axioms of ${\rm PA}^{\rm cc}_\delta$ can be derived from the axioms of ${\rm BPA}^{\rm cc}_\delta$. Therefore, it is sufficient to prove that the axioms of ${\rm BPA}^{\rm cc}_\delta$ constitute a complete axiomatization of the restrictions of the full splitting bisimulation models of ${\rm PA}^{\rm cc}_\delta$ to the constants and operators of ${\rm BPA}^{\rm cc}_\delta$. To prove this, we adapt the proof of completeness of the axioms of ${\rm BPA}_\tau$ for the graph models of ${\rm BPA}_\tau$ given in Ref. [24].

We use two functions relating basic terms and finite acyclic elements of $\mathbb{CTS}_\kappa$. We write $\mathbb{CTS}^{\rm fa}_\kappa$ for the set of all finite acyclic elements of $\mathbb{CTS}_\kappa$. Let ${\rm cts}\colon \mathcal{B} \to \mathbb{CTS}^{\rm fa}_\kappa$ be a function such that for all $p \in \mathcal{B}$, ${\rm cts}(p)$ is a representative of the interpretation of p in $\mathfrak{P}^{\rm cc}_\kappa$, and let ${\rm term}\colon \mathbb{CTS}^{\rm fa}_\kappa \to \mathcal{B}$ be a function such that for all $T = (S, \to, \to\surd, s^0) \in \mathbb{CTS}^{\rm fa}_\kappa$, ${\rm term}(T)$ is ${\rm term}(s^0)$, where for every state $s \in S$, ${\rm term}(s) = \sum_{1 \leq i \leq n} \underline{\alpha_i} :\!\to a_i \cdot {\rm term}(s_i) + \sum_{1 \leq j \leq m} \underline{\beta_j} :\!\to b_j$ if $s \xrightarrow{[\alpha_1]\,a_1} s_1$, $\ldots$, $s \xrightarrow{[\alpha_n]\,a_n} s_n$ and $s \xrightarrow{[\beta_1]\,b_1} \surd$, $\ldots$, $s \xrightarrow{[\beta_m]\,b_n} \surd$ are all transitions from state s, and for every $\alpha \in \mathcal{C}$, $\underline{\alpha}$ is a closed term of sort $\mathbf{C}$ of which the value in $\mathcal{C}$ is α. It is clear that the functions cts and term are not uniquely defined, but it is easy to see that ${\rm BPA}^{\rm cc}_\delta \vdash {\rm term}({\rm cts}(p)) = p$ for all $p \in \mathcal{B}$.

We proceed with the crucial step of the proof. We introduce a reduction relation $\twoheadrightarrow$ on the set of finite acyclic elements of $\mathbb{CTS}_\kappa$. The one-step reductions are *sharing* of double states as in Ref. [24], and two variants of *joining* of transitions: (a) replacing $s \xrightarrow{[\alpha]\,a} s''$ and $s \xrightarrow{[\beta]\,a} s''$ by $s \xrightarrow{[\alpha \sqcup \beta]\,a} s''$ and (b) replacing $s \xrightarrow{[\alpha]\,a} \surd$ and $s \xrightarrow{[\beta]\,a} \surd$ by $s \xrightarrow{[\alpha \sqcup \beta]\,a} \surd$. The one-step reduction relation $\rightarrowtail$ on $\mathbb{CTS}^{\rm fa}_\kappa$ is defined by $T \rightarrowtail T'$ iff T reduces to T' by one of the above-mentioned

one-step reductions. We write $\rightarrowtail\!\!\!\!\!\rightarrow$ for the reflexive and transitive closure of $\rightarrowtail$, and $\rightarrowtail\!\!\!\!\!\leftarrow$ for the reflexive and transitive closure of $\rightarrowtail \cup \rightarrowtail^{-1}$.

The following are important properties of $\rightarrowtail$:

(1) $\rightarrowtail\!\!\!\!\!\rightarrow$ is strongly normalizing;
(2) for all $T, T' \in \mathbb{CTS}^{\mathrm{fa}}_{\kappa}$, $T \rightarrowtail\!\!\!\!\!\rightarrow T'$ implies $T \leftrightarrows T'$;
(3) for all $T, T' \in \mathbb{CTS}^{\mathrm{fa}}_{\kappa}$ that are in normal form, $T \leftrightarrows T'$ implies $T = T'$;
(4) for all $T, T' \in \mathbb{CTS}^{\mathrm{fa}}_{\kappa}$, $T \rightarrowtail T'$ implies $\mathrm{BPA}^{\mathrm{cc}}_{\delta} \vdash \mathrm{term}(T) = \mathrm{term}(T')$.

Verifying properties 1, 2 and 4 is trivial. Property 3 can be verified by proving the property

> for all $T \in \mathbb{CTS}^{\mathrm{fa}}_{\kappa}$ that are in normal form, any splitting bisimulation between T and itself is the identity relation,

together with property 3 simultaneously by induction on the number of transitions of the conditional transition systems concerned. This is similar to the proof of Theorem 2.12 from Ref. [24], but is easier.

It follows immediately from properties 1, 2 and 3 that, for all $T, T' \in \mathbb{CTS}^{\mathrm{fa}}_{\kappa}$, $T \leftrightarrows T'$ iff $T \rightarrowtail\!\!\!\!\!\leftarrow T'$. From this, property 4, the fact that $\mathfrak{P}^{\mathrm{cc}}_{\kappa} \models p = q$ implies $\mathrm{cts}(p) \leftrightarrows \mathrm{cts}(q)$ and the fact that $\mathrm{BPA}^{\mathrm{cc}}_{\delta} \vdash \mathrm{term}(\mathrm{cts}(p)) = p$ for all $p \in \mathcal{B}$, it follows immediately that for all basic terms p and q, $\mathfrak{P}^{\mathrm{cc}}_{\kappa} \models p = q$ implies $\mathrm{BPA}^{\mathrm{cc}}_{\delta} \vdash p = q$. This results extends to all closed terms p and q of sort $\mathbf{P}$ from the language of $\mathrm{BPA}^{\mathrm{cc}}_{\delta}$ by Lemma 1. $\Box$

As to be expected, the full splitting bisimulation models are related by isomorphic embeddings.

Theorem 12 (Isomorphic Embedding) *Let κ and κ' be infinite cardinals such that $\kappa < \kappa'$. Then $\mathfrak{P}^{\mathrm{cc}}_{\kappa}$ is isomorphically embedded in $\mathfrak{P}^{\mathrm{cc}}_{\kappa'}$.*

PROOF. The proof is analogous to the proof of the corresponding property for the full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{c}}$ given in Ref. [1]. $\Box$

5 SOS-Based Splitting Bisimilarity for $\mathrm{PA}^{\mathrm{cc}}_{\delta}$

It is customary to associate transition systems with closed terms (of sort $\mathbf{P}$) from the language of an ACP-like theory about processes by means of structural operational semantics and to identify closed terms if their associated transition systems are equivalent by a bisimilarity-based notion of equivalence.

The structural operational semantics of $\mathrm{PA}_\delta^{\mathrm{cc}}$ presented in Section 2 determines a conditional transition system for each closed term of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$. These transition systems are special in the sense that their states are closed terms of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$.

Let p be a closed term of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$. Then the transition system of p *induced by* the structural operational semantics of $\mathrm{PA}_\delta^{\mathrm{cc}}$, written $\mathrm{CTS}(p)$, is the connected conditional transition system $\Gamma(S, \rightarrow, \rightarrow\surd, s^0)$, where:

- S is the set of closed terms of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$;
- $\xrightarrow{(\alpha,a)} \subseteq S \times S$ and $\xrightarrow{(\alpha,a)}\surd \subseteq S$, for each $\alpha \in \mathcal{C} \setminus \{\bot\}$ and $a \in \mathsf{A}$, are the smallest subsets of $S \times S$ and S, respectively, for which the transition rules from Table 3 hold;
- $s^0 \in S$ is the closed term p.

Let p and q be closed terms of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$. Then we say that p and q are *splitting bisimilar*, written $p \leftrightarrows q$, if $\mathrm{CTS}(p) \leftrightarrows \mathrm{CTS}(q)$.

Clearly, the structural operational semantics does not give rise to infinitely branching conditional transition systems. For each closed term p of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$, there exists a $T \in \mathbb{CTS}_{\aleph_0}$ such that $\mathrm{CTS}(p) \mathrel{\hat{=}} T$. In Section 4, it has been shown that it is possible to consider infinitely branching conditional transition systems as well.

Remark 13 *Let p be a closed term of sort $\mathbf{P}$ from the language of $\mathrm{PA}_\delta^{\mathrm{cc}}$. Then $[\,\mathrm{CTS}(p)\,]_{\leftrightarrows}$ is the interpretation of p in the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$. In other words, the structural operational semantics of $\mathrm{PA}_\delta^{\mathrm{cc}}$ just provides a way to associate with each closed term of sort $\mathbf{P}$ a representative of the interpretation of that term in the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$. With the incorporation of communication below, it becomes troublesome to present a structural operational semantics. Therefore, we focus in this paper with respect to semantics on the full splitting bisimulation models. The structural operational semantics of $\mathrm{PA}_\delta^{\mathrm{cc}}$ is presented as well, because it is considered to be intuitively clear.*

6 ACP with Coordination Conditions

In order to support communication, we generalize the parallel composition operator of $\mathrm{PA}_\delta^{\mathrm{cc}}$, resulting in $\mathrm{ACP}^{\mathrm{cc}}$.

Just as in $\mathrm{PA}_\delta^{\mathrm{cc}}$, it is assumed that a fixed but arbitrary finite set of *actions* A, with $\delta \notin \mathsf{A}$ and $t \in \mathsf{A}$, has been given. In $\mathrm{ACP}^{\mathrm{cc}}$, it is further assumed that

a fixed but arbitrary commutative and associative *communication* function $|:\mathsf{A}_\delta \times \mathsf{A}_\delta \to \mathsf{A}_\delta$, such that $\delta \,|\, a = \delta$ and $t \,|\, a = \delta$ for all $a \in \mathsf{A}_\delta$, has been given. The function $|$ is regarded to give the result of synchronously performing any two actions for which this is possible, and to be δ otherwise.

The theory $\mathrm{ACP}^{\mathrm{cc}}$ has the constants and operators of $\mathrm{PA}^{\mathrm{cc}}_\delta$, with the exception of the left merge operator, and in addition:

- for each $E \subseteq \mathsf{A}$ with $t \in E$, the unary *enabledness update* operator $\Psi_E : \mathbf{C} \to \mathbf{C}$.

Let ζ be a closed term of sort $\mathbf{C}$, and let $E \subseteq \mathsf{A}$ with $t \in E$. Intuitively, the enabledness update operators can be explained as follows:

- $\Psi_E(\zeta)$ is ζ with all enabledness conditions $\mathcal{E}_a$ with $a \notin E$ replaced by $\bot$ and all enabledness conditions $\mathcal{E}_a$ with $a \in E$ replaced by $\top$.

In $\mathrm{PA}^{\mathrm{cc}}_\delta$, the enabledness of actions is affected only by encapsulation and pre-abstraction. In $\mathrm{ACP}^{\mathrm{cc}}$, the enabledness of actions is affected by parallel composition with another process as well. Parallel composition with another process places a process in a context in which each $a \in \mathsf{A}$ is either enabled or disabled, depending upon the capability of the other process to perform an action $b \in \mathsf{A}$ such that $a \,|\, b$ is an enabled action in the context in which the whole is placed.

The axioms of $\mathrm{ACP}^{\mathrm{cc}}$ are the axioms of $\mathrm{PA}^{\mathrm{cc}}_\delta$, with axioms M1–M4 and GC8 replaced by axioms CME and EE1–EE7 from Table 4. Several axioms given in Table 4 are actually axiom schemas: $a_1, b_1, a'_1, b'_1, a_2, b_2, a'_2, b'_2, \ldots$ stand for arbitrary elements of A_δ, c stands for an arbitrary element of A, and E stands for an arbitrary subset of A that contains t. In axiom schema CME, for every $U, V, W \subseteq \mathsf{A}$ with $t \in U, V, W$, $\Phi_{U,V,W}$ iff for all $u \in \mathsf{A} \setminus \{t\}$:

$$\begin{aligned} u \in V \;\Leftrightarrow\; & \exists j < n' \bullet (\Psi_W(\zeta'_j) = \top \wedge u \,|\, a'_j \in U) \\ & \vee \exists j < m' \bullet (\Psi_W(\xi'_j) = \top \wedge u \,|\, b'_j \in U) \end{aligned}$$

and

$$\begin{aligned} u \in W \Leftrightarrow\; & \exists i < n \bullet (\Psi_V(\zeta_i) = \top \wedge a_i \,|\, u \in U) \\ & \vee \exists i < m \bullet (\Psi_V(\xi_i) = \top \wedge b_i \,|\, u \in U)\ . \end{aligned}$$

The axioms of $\mathrm{ACP}^{\mathrm{cc}}$ do not include axioms CM1–CM9 of ACP (see e.g. Ref. [2]), i.e. the axioms of ACP for parallel composition, left merge and communication merge, and axioms GC8–GC10 of $\mathrm{ACP}^{\mathrm{c}}$ (see e.g. Ref. [1]), i.e. the additional axioms of $\mathrm{ACP}^{\mathrm{c}}$ for left merge and communication merge. The reason for this is that these axioms cause problems. The point is that applying them leads to changes of the context in which the processes involved in the

Table 4
Axioms for ACP^{cc}

$$x = \sum_{i<n} \phi_i :\rightarrow a_i \cdot x_i + \sum_{i<m} \psi_i :\rightarrow b_i \wedge y = \sum_{j<n'} \phi'_j :\rightarrow a'_j \cdot y_j + \sum_{j<m'} \psi'_j :\rightarrow b'_j \Rightarrow$$

$$x \parallel y = \sum_{\substack{t\in U\subseteq \mathsf{A}, t\in V\subseteq \mathsf{A}, t\in W\subseteq \mathsf{A}, \\ \Phi_{U,V,W}}} \bigsqcap_{u\in U} \mathcal{E}_u \sqcap \bigsqcap_{u\in \mathsf{A}\setminus U} -\mathcal{E}_u :\rightarrow$$
$$\Big(\sum_{i<n} \Psi_V(\phi_i) :\rightarrow a_i \cdot (x_i \parallel y) +$$
$$\sum_{i<m} \Psi_V(\psi_i) :\rightarrow b_i \cdot y +$$
$$\sum_{j<n'} \Psi_W(\phi'_j) :\rightarrow a'_j \cdot (x \parallel y_j) +$$
$$\sum_{j<m'} \Psi_W(\psi'_j) :\rightarrow b'_j \cdot x +$$
$$\sum_{i<n, j<n'} \Psi_V(\phi_i) \sqcap \Psi_W(\phi'_j) :\rightarrow (a_i \mid a'_j) \cdot (x_i \parallel y_j) +$$
$$\sum_{i<m, j<n'} \Psi_V(\psi_i) \sqcap \Psi_W(\phi'_j) :\rightarrow (b_i \mid a'_j) \cdot y_j +$$
$$\sum_{i<n, j<m'} \Psi_V(\phi_i) \sqcap \Psi_W(\psi'_j) :\rightarrow (a_i \mid b'_j) \cdot x_i +$$
$$\sum_{i<m, j<m'} \Psi_V(\psi_i) \sqcap \Psi_W(\psi'_j) :\rightarrow (b_i \mid b'_j)\Big) \qquad \text{CME}$$

$\Psi_E(\bot) = \bot$		EE1	$\Psi_E(-\phi) = -\Psi_E(\phi)$	EE5
$\Psi_E(\top) = \top$		EE2	$\Psi_E(\phi \sqcup \psi) = \Psi_E(\phi) \sqcup \Psi_E(\psi)$	EE6
$\Psi_E(\mathcal{E}_c) = \bot$	if $c \notin E$	EE3	$\Psi_E(\phi \sqcap \psi) = \Psi_E(\phi) \sqcap \Psi_E(\psi)$	EE7
$\Psi_E(\mathcal{E}_c) = \top$	if $c \in E$	EE4		

parallel composition are placed. This point will be illustrated below, but first CME, the axiom schema that replaces CM1–CM9 and GC8–GC10 in ACP^{cc}, is explained.

Consider processes p, q and $p \parallel q$. The process p and the context in which $p \parallel q$ is placed form parts of the context in which q is placed; and the process q and the context in which $p \parallel q$ is placed form parts of the context in which p is placed. Any subset of A that includes t may be the set of all actions that are enabled in the context in which $p \parallel q$ is placed. Suppose that $U \subseteq \mathsf{A}$ with $t \in U$ is the set of all actions that are enabled in the context in which $p \parallel q$ is placed. Furthermore, suppose that $V \subseteq \mathsf{A}$ with $t \in V$ and $W \subseteq \mathsf{A}$ with $t \in W$ are the sets of all actions that are enabled in the contexts in which p and q, respectively, are placed. Then the following must hold for all $a \in \mathsf{A} \setminus \{t\}$:

- $a \in V$ iff q, with exactly the actions in W enabled, can perform an action b

such that $a \mid b \in U$ (because a is enabled in q together with the context in which $p \parallel q$ is placed);
- $a \in W$ iff p, with exactly the actions in V enabled, can perform an action b such that $a \mid b \in U$ (because a is enabled in p together with the context in which $p \parallel q$ is placed).

V and W determine whether the conditions under which p and q can perform their initial actions evaluate to $\top$ or $\bot$. In the case of p, for example, the basis is that $\mathcal{E}_a$ evaluates to $\top$ if $a \in V$, and $\bot$ otherwise. All this is made precise in axiom schema CME. Notice that V and W need not exist for each U: the behaviour of p and q may inhibit proper mutual enabling of actions for $p \parallel q$. In such cases, $p \parallel q$ is considered to be incapable of doing anything. In other words, if V and W do not exist for some U, $p \parallel q$ is considered to behave the same as δ in the event of U being the set of all actions that are enabled in the context in which $p \parallel q$ is placed.

We proceed with giving a few examples. In the examples, we take A such that $\mathsf{A} = A \cup \{\overline{a} \mid a \in A\} \cup \{t\}$ for some set A such that $t \notin A$. Moreover, we take $\mid : \mathsf{A}_\delta \times \mathsf{A}_\delta \to \mathsf{A}_\delta$ such that, for all $a \in A$ and $b \in \mathsf{A}$, $a \mid \overline{a} = t$, $a \mid b = \delta$ if $b \neq \overline{a}$, $\overline{a} \mid b = \delta$ if $b \neq a$, and $t \mid b = \delta$. This kind of communication is what we call CCS-style communication. We start with giving an example of processes p and q of which the behaviour inhibits proper mutual enabling of actions for $p \parallel q$, whatever actions are enabled in the context in which $p \parallel q$ is placed. Consider the processes $p \equiv \mathcal{E}_a :\rightarrow b$ and $q \equiv -\mathcal{E}_{\overline{b}} :\rightarrow \overline{a}$. The behaviour of these processes inhibits proper mutual enabling of actions for $p \parallel q$ whatever actions are enabled in the context in which $p \parallel q$ is placed: if a is enabled by q, then $\overline{b}$ is enabled by p and consequently a is not enabled by q; and if a is not enabled by q, then $\overline{b}$ is not enabled by p and consequently a is enabled by q. Hence, we have

$$\mathcal{E}_a :\rightarrow b \parallel -\mathcal{E}_{\overline{b}} :\rightarrow \overline{a} = \delta \; . \tag{1}$$

More precisely, for all $U \subseteq \mathsf{A}$ with $t \in U$, there do not exist $V, W \subseteq \mathsf{A}$ with $t \in V$ and $t \in W$ such that $\Phi_{U,V,W}$. Using this, equation (1) follows immediately from CME. We proceed with giving an example of processes p and q of which the behaviour does not inhibit proper mutual enabling of actions for $p \parallel q$, whatever actions are enabled in the context in which $p \parallel q$ is placed. Consider the processes $p \equiv \mathcal{E}_a :\rightarrow b + \overline{c}$ and $q \equiv \overline{a} + \mathcal{E}_{\overline{b}} :\rightarrow c$. There is a unique proper mutual enabling, viz. the mutual enabling in which exactly the actions a and $\overline{c}$ are enabled by q, and exactly the actions $\overline{b}$ and c are enabled by p. Hence, we have

$$(\mathcal{E}_a :\rightarrow b + \overline{c}) \parallel (\overline{a} + \mathcal{E}_{\overline{b}} :\rightarrow c)$$
$$= b \cdot (\overline{a} + c) + \overline{c} \cdot (\overline{a} + c) + \overline{a} \cdot (b + \overline{c}) + c \cdot (b + \overline{c}) + t \; . \tag{2}$$

More precisely, for all $U \subseteq \mathsf{A}$ with $t \in U$, $\Phi_{U,V,W}$ iff $V = \{a, \overline{c}\}$ and $W = \{\overline{b}, c\}$. Using this, equation (2) follows easily from CME, EE4, GC1, GC7 and BA. Finally, we give an example of processes p and q of which the behaviour is such that there are two proper mutual enablings of actions for $p \| q$, whatever actions are enabled in the context in which $p \parallel q$ is placed. Consider the processes $p \equiv \mathcal{E}_a :\rightarrow b + -\mathcal{E}_a :\rightarrow c$ and $q \equiv \mathcal{E}_{\overline{b}} :\rightarrow \overline{a} + -\mathcal{E}_{\overline{b}} :\rightarrow \overline{c}$. There are two proper mutual enablings, viz. the mutual enabling in which only the action a is enabled by q and only the action $\overline{b}$ is enabled by p, and the mutual enabling in which only the action c is enabled by q and only the action $\overline{c}$ is enabled by p. Hence, we have

$$(\mathcal{E}_a :\rightarrow b + -\mathcal{E}_a :\rightarrow c) \parallel (\mathcal{E}_{\overline{b}} :\rightarrow \overline{a} + -\mathcal{E}_{\overline{b}} :\rightarrow \overline{c})$$
$$= b \cdot \overline{a} + \overline{a} \cdot b + c \cdot \overline{c} + \overline{c} \cdot c + t \; . \tag{3}$$

More precisely, for all $U \subseteq \mathsf{A}$ with $t \in U$, $\Phi_{U,V,W}$ iff $V = \{a\}$ and $W = \{\overline{b}\}$ or $V = \{c\}$ and $W = \{\overline{c}\}$. Using this, equation (3) follows easily from CME, EE3, EE4, EE5, GC1, GC2, GC7 and BA.

Axioms similar to the axioms of ACP and ACP$^{\mathrm{c}}$ for parallel composition are too much to expect for ACP$^{\mathrm{cc}}$. The mutual enabling of actions involved in the parallel composition of two processes is a matter which can only be resolved by looking at the processes as a whole. As a case in point, let us consider the process $\partial_H((a + \overline{b}) \parallel \mathcal{E}_{\overline{a}} :\rightarrow b)$, where $H = \{a, \overline{a}, b, \overline{b}\}$, with CCS-style communication. Using CME, we derive as intended

$$\partial_H((a + \overline{b}) \parallel \mathcal{E}_{\overline{a}} :\rightarrow b) = \partial_H(a \cdot b + \overline{b} \cdot b + b \cdot (a + \overline{b}) + t) = t \; .$$

If we could use axioms CM1–CM9 of ACP, then we would be able to derive $\partial_H((a + \overline{b}) \parallel \mathcal{E}_{\overline{a}} :\rightarrow b) = \partial_H(a \between \mathcal{E}_{\overline{a}} :\rightarrow b + \overline{b} \between \mathcal{E}_{\overline{a}} :\rightarrow b + \mathcal{E}_{\overline{a}} :\rightarrow b \between (a + \overline{b}) + a \mid \mathcal{E}_{\overline{a}} :\rightarrow b + \overline{b} \mid \mathcal{E}_{\overline{a}} :\rightarrow b)$. Applying CM4, CM8 and CM9 leads to splitting up the context in which $\mathcal{E}_{\overline{a}} :\rightarrow b$ is placed. The main problem with that is the loss in $\overline{b} \between \mathcal{E}_{\overline{a}} :\rightarrow b$ and $\overline{b} \mid \mathcal{E}_{\overline{a}} :\rightarrow b$ of the enabling of $\overline{a}$ by $a + \overline{b}$. If we could use axioms GC8–GC10 of ACP$^{\mathrm{c}}$ as well, then we would even be able to derive $\partial_H((a + \overline{b}) \parallel \mathcal{E}_{\overline{a}} :\rightarrow b) = \partial_H(\mathcal{E}_{\overline{a}} :\rightarrow (a \cdot b + \overline{b} \cdot b + b \cdot (a + \overline{b}) + t)) = \delta$. Applying GC8–GC10 leads to moving the condition $\mathcal{E}_{\overline{a}}$ outwards. The problem with that is that it gets lost that the condition $\mathcal{E}_{\overline{a}}$ concern the enabledness of action a in a different context.

Note that, in the example just given, axioms CM1–CM3 and CM5–CM7 do

not cause any problem. In fact, we can extend $\mathrm{ACP}^{\mathrm{cc}}$ with the auxiliary operators $\between$ and $\mid$ and axioms CM1–CM3 and CM5–CM7. However, we cannot obtain a finite axiomatization of the parallel composition operator $\parallel$ in that way, whereas the usual reason to introduce $\between$ and $\mid$ is to obtain such a finite axiomatization.

Like in the case of $\mathrm{PA}_\delta^{\mathrm{cc}}$, we can prove that all closed terms of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$ are derivably equal to a basic term.

Lemma 14 (Elimination for $\mathrm{ACP}^{\mathrm{cc}}$) *For all closed terms p of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$, there exists a basic term $q \in \mathcal{B}$ such that $\mathrm{ACP}^{\mathrm{cc}} \vdash p = q$.*

PROOF. CME is in the shape of an implication. Let CME$'$ be the equation obtained from CME by taking the right-hand side of this implication and replacing every occurrence of x and y by the right-hand side of the equation for x and y, respectively, at the left-hand side of this implication. Let CME$''$ be CME$'$ with the conditions $\bigsqcap_{u \in U} \mathcal{E}_u \sqcap \bigsqcap_{u \in \mathsf{A} \setminus U} -\mathcal{E}_u$ moved inward (applying GC4) and combined with the conditions $\Psi_V(\zeta_i) \sqcap \Psi_W(\zeta'_j)$, $\Psi_V(\xi_i) \sqcap \Psi_W(\zeta'_j)$, $\Psi_V(\zeta_i) \sqcap \Psi_W(\xi'_j)$ and $\Psi_V(\xi_i) \sqcap \Psi_W(\xi'_j)$ (applying GC6). The term rewriting system consisting of all axioms of $\mathrm{ACP}^{\mathrm{cc}}$, except BA1–BA8 and A1–A3 and with CME replaced by CME$''$, oriented from left to right is strongly normalizing. This can be proved by using the method of lexicographical path ordering of Kamin and Lévy, making the signature one-sorted, taking the ordering $\ldots > \between > :\!\!\rightarrow > \cdot > +$, $:\!\!\rightarrow > \delta$, $:\!\!\rightarrow > \sqcap$, $\partial_H > :\!\!\rightarrow$, $\partial_H > \bot$, $\partial_H > -$, $\partial_H > \sqcup$, $t_I > :\!\!\rightarrow$, $t_I > t$, $t_I > \top$, $t_I > -$, $t_I > \sqcup$, $\between > a$, $\between > \mathcal{E}_a$, $\between > -$, $\between > \Psi_E$, for all $H \subseteq \mathsf{A} \setminus \{t\}$, $I \subseteq \mathsf{A}$, $a \in \mathsf{A}$ and $E \subseteq \mathsf{A}$ with $t \in E$, and giving the lexicographical status for the first argument to $\cdot$ and the lexicographical status for the second argument to $:\!\!\rightarrow$. Moreover, it is easy to see that each normal form is a basic term. $\Box$

We can also prove that $\mathrm{ACP}^{\mathrm{cc}}$ is a conservative extension of $\mathrm{BPA}_\delta^{\mathrm{cc}}$.

Lemma 15 (Conservative extension) *If p and q are closed terms of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$, then $\mathrm{BPA}_\delta^{\mathrm{cc}} \vdash p = q$ iff $\mathrm{ACP}^{\mathrm{cc}} \vdash p = q$.*

PROOF. The proof is analogous to the proof of Lemma 3. $\Box$

The preceding two lemmas will be useful in the completeness proof of $\mathrm{ACP}^{\mathrm{cc}}$ for the full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$ that will be introduced in Section 7.

Remark 16 *The priority guards added to CCS in Ref. [13] are similar to the coordination conditions of* $\mathrm{ACP}^{\mathrm{cc}}$*, but subject to the restriction that CCS-style actions must be taken. Take CCS-style actions, i.e.* $\mathsf{A} = A \cup \{\overline{a} \mid a \in A\} \cup \{t\}$ *for some set* A *such that* $t \notin A$*. Let* $a, b \in \mathsf{A}$*,* $a_1, \ldots, a_n \in \mathsf{A} \setminus \{t\}$*, and let* p *and* q *be terms of sort* $\mathbf{P}$*. Then* $-(\mathcal{E}_{\overline{a_1}} \sqcup \ldots \sqcup \mathcal{E}_{\overline{a_n}}) :\rightarrow a \cdot p$*, where* $\overline{\overline{a}} = a$*, is written* $\{a_1, \ldots, a_n\} : a \,.\, p$ *with priority guards. The following shows an inconvenience of priority guards:* $-(\mathcal{E}_{\overline{a_1}} \sqcup \ldots \sqcup \mathcal{E}_{\overline{a_n}}) :\rightarrow (a \cdot p + b \cdot q)$ *must be written* $\{a_1, \ldots, a_n\}{:}a.p + \{a_1, \ldots, a_n\}{:}b.q$ *with priority guards. The following shows a limitation of priority guards:* $\zeta :\rightarrow a \cdot p$*, where* ζ *is of a form different from* $-(\mathcal{E}_{\overline{a_1}} \sqcup \ldots \sqcup \mathcal{E}_{\overline{a_n}})$*, cannot be written with priority guards. A consequence of the limitations of priority guards is that splitting bisimilarity can be replaced by the simpler strong offer bisimilarity of Ref. [13].*

Remark 17 $\mathrm{ACP}^{\mathrm{cc}}$ *includes pre-abstraction, but not abstraction. Abstraction is usually based on observation equivalence [11] or branching bisimulation equivalence [29], which both abstract from both the structure of finitary internal activity and its presence. That way, a process without internal actions can be equivalent to a process with internal actions. For example,* $a \cdot b = a \cdot t \cdot b$ *under observation equivalence and branching bisimulation equivalence. However,* $a \cdot b$ *cannot always be replaced by* $a \cdot t \cdot b$ *in* $\mathrm{ACP}^{\mathrm{cc}}$*: with CCS-style communication,* $a \cdot b \parallel -(\mathcal{E}_{\overline{a}} \sqcup \mathcal{E}_{\overline{b}}) :\rightarrow c = a \cdot b \cdot c$*, but* $a \cdot t \cdot b \parallel -(\mathcal{E}_{\overline{a}} \sqcup \mathcal{E}_{\overline{b}}) :\rightarrow c \neq a \cdot b \cdot c$*.*

Terms of sort $\mathbf{C}$ are interpreted in $\mathcal{C}$ as in the case of $\mathrm{PA}^{\mathrm{cc}}_{\delta}$, the association of operations with the extra-Boolean operators Ψ_E excepted. With the operator Ψ_E is associated the unique endomorphism of $\mathcal{C}$ extending the function Ψ_E on $\{\mathcal{E}_a \mid a \in \mathsf{A}\}$ defined by $\Psi_E(\mathcal{E}_a) = \top$ if $a \in E$ and $\Psi_E(\mathcal{E}_a) = \bot$ if $a \notin E$. It follows automatically that axioms BA1–BA8, ED1–ED7, EI1–EI7 and EE1–EE7 constitute a sound and complete axiomatization of the expansion of $\mathcal{C}$ with the operations ∂_H, t_I defined in Section 2 and Ψ_E defined above. Henceforth, we loosely write $\mathcal{C}$ for this expansion of $\mathcal{C}$.

Associating transition systems with closed terms of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$ by means of structural operational semantics is troublesome (see also Remark 21). Moreover, the structural operational semantics of $\mathrm{ACP}^{\mathrm{cc}}$ would just provide a way to associate with each closed term of sort $\mathbf{P}$ a representative of the interpretation of that term in the full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$, which are presented in Section 7. For these reasons, we refrain from presenting the structural operational semantics of $\mathrm{ACP}^{\mathrm{cc}}$.

7 Full Splitting Bisimulation Models of $\mathrm{ACP}^{\mathrm{cc}}$

In this section, we adapt the full splitting bisimulation models of $\mathrm{PA}^{\mathrm{cc}}_{\delta}$ to $\mathrm{ACP}^{\mathrm{cc}}$. In order to cover communication, the operation on $\mathbb{CTS}_{\kappa}/\leftrightarrows$ associated

with the operator $\|$ has to be adapted.

Like before, we begin by associating an operation $\widehat{\|}'$ on $\mathbb{CTTS}_\kappa$ with the operator $\|$.

- Let $T_i = (S_i, \rightarrow_i, \rightarrow\surd_i, s_i^0) \in \mathbb{CTTS}_\kappa$ for $i = 1, 2$. Then

$$T_1 \,\widehat{\|}'\, T_2 = \Gamma(S, \rightarrow, \rightarrow\surd, s^0) \ ,$$

where

$$\begin{aligned} s^0 &= (s_1^0, s_2^0) \ , \\ s^\surd &= \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1 \cup S_2)) \ , \\ S &= ((S_1 \cup \{s^\surd\}) \times (S_2 \cup \{s^\surd\})) \setminus \{(s^\surd, s^\surd)\} \ , \end{aligned}$$

and for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$
\begin{aligned}
\xrightarrow{(\alpha,a)} \quad = \Big\{ ((s_1,s_2),(s_1',s_2)) \;\Big|\; (s_1',s_2) \in S \wedge \\
\bigvee_{\substack{\alpha' \in \mathcal{C}^-, \\ t \in U,V,W \subseteq \mathsf{A}}} \Big(s_1 \xrightarrow{[\alpha']\, a}_1 s_1' \wr s^{\checkmark} \wedge \\
\Phi'_{U,V,W}(s_1,s_2) \wedge \\
\textstyle\bigsqcap_{u \in U} \mathcal{E}_u \sqcap \bigsqcap_{u \in \mathsf{A} \setminus U} -\mathcal{E}_u = \alpha \wedge \\
\Psi_V(\alpha') = \top \Big) \Big\} \\
\cup \Big\{ ((s_1,s_2),(s_1,s_2')) \;\Big|\; (s_1,s_2') \in S \wedge \\
\bigvee_{\substack{\alpha' \in \mathcal{C}^-, \\ t \in U,V,W \subseteq \mathsf{A}}} \Big(s_2 \xrightarrow{[\alpha']\, a}_2 s_2' \wr s^{\checkmark} \wedge \\
\Phi'_{U,V,W}(s_1,s_2) \wedge \\
\textstyle\bigsqcap_{u \in U} \mathcal{E}_u \sqcap \bigsqcap_{u \in \mathsf{A} \setminus U} -\mathcal{E}_u = \alpha \wedge \\
\Psi_W(\alpha') = \top \Big) \Big\} \\
\cup \Big\{ ((s_1,s_2),(s_1',s_2')) \;\Big|\; (s_1',s_2') \in S \wedge \\
\bigvee_{\substack{\alpha',\beta' \in \mathcal{C}^-, a',b' \in \mathsf{A}, \\ t \in U,V,W \subseteq \mathsf{A}}} \Big(s_1 \xrightarrow{[\alpha']\, a'}_1 s_1' \wr s^{\checkmark} \wedge s_2 \xrightarrow{[\beta']\, b'}_2 s_2' \wr s^{\checkmark} \wedge \\
\Phi'_{U,V,W}(s_1,s_2) \wedge \\
\textstyle\bigsqcap_{u \in U} \mathcal{E}_u \sqcap \bigsqcap_{u \in \mathsf{A} \setminus U} -\mathcal{E}_u = \alpha \wedge \\
\Psi_V(\alpha') \sqcap \Psi_W(\beta') = \top \wedge a' \mid b' = a \Big) \Big\}\,,
\end{aligned}
$$

$$
\begin{aligned}
\xrightarrow{(\alpha,a)} \surd = \Big\{ (s_1, s^{\checkmark}) \;\Big|\; s_1 \xrightarrow{[\alpha]\, a} \surd_1 \Big\} \\
\cup \Big\{ (s^{\checkmark}, s_2) \;\Big|\; s_2 \xrightarrow{[\alpha]\, a} \surd_2 \Big\} \\
\cup \Big\{ (s_1,s_2) \;\Big|\; \\
\bigvee_{\substack{\alpha',\beta' \in \mathcal{C}^-, a',b' \in \mathsf{A}, \\ t \in U,V,W \subseteq \mathsf{A}}} \Big(s_1 \xrightarrow{[\alpha']\, a'} \surd_1 \wedge s_2 \xrightarrow{[\beta']\, b'} \surd_2 \wedge \\
\Phi'_{U,V,W}(s_1,s_2) \wedge \\
\textstyle\bigsqcap_{u \in U} \mathcal{E}_u \sqcap \bigsqcap_{u \in \mathsf{A} \setminus U} -\mathcal{E}_u = \alpha \wedge \\
\Psi_V(\alpha') \sqcap \Psi_W(\beta') = \top \wedge a' \mid b' = a \Big) \Big\}\,.
\end{aligned}
$$

and for every $U, V, W \subseteq \mathsf{A}$ with $t \in U, V, W$ and for every $(s_1, s_2) \in S$,

$\Phi'_{U,V,W}(s_1, s_2)$ iff for all $u \in \mathsf{A} \setminus \{t\}$:

$$
\begin{aligned}
&u \in V \Leftrightarrow \\
&\quad \exists \alpha'', a'', s'' \bullet \Big(s_2 \xrightarrow{[\alpha'']\, a''}_2 s'' \wedge \Psi_W(\alpha'') = \top \wedge u \mid a'' \in U \Big) \\
&\quad \vee \exists \alpha'', a'' \bullet \Big(s_2 \xrightarrow{[\alpha'']\, a''} \surd_2 \wedge \Psi_W(\alpha'') = \top \wedge u \mid a'' \in U \Big) \\
&\text{and} \\
&u \in W \Leftrightarrow \\
&\quad \exists \alpha'', a'', s'' \bullet \Big(s_1 \xrightarrow{[\alpha'']\, a''}_1 s'' \wedge \Psi_V(\alpha'') = \top \wedge a'' \mid u \in U \Big) \\
&\quad \vee \exists \alpha'', a'' \bullet \Big(s_1 \xrightarrow{[\alpha'']\, a''} \surd_1 \wedge \Psi_V(\alpha'') = \top \wedge a'' \mid u \in U \Big) \,.
\end{aligned}
$$

We can show that splitting bisimilarity is a congruence with respect to parallel composition.

Lemma 18 (Congruence) *Let κ be an infinite cardinal. Then for all T_1, T_2, $T'_1, T'_2 \in \mathbb{CTTS}_\kappa$, $T_1 \leftrightarroweq T'_1$ and $T_2 \leftrightarroweq T'_2$ imply $T_1 \mathbin{\widehat{\|}'} T_2 \leftrightarroweq T'_1 \mathbin{\widehat{\|}'} T'_2$.*

PROOF. Although parallel composition as considered in the setting of $\mathrm{ACP}^{\mathrm{cc}}$ differs from parallel composition as considered in the setting of $\mathrm{PA}^{\mathrm{cc}}_\delta$, a witnessing splitting bisimulation can be constructed in the same way as in the proof of Lemma 9. It is straightforward to show that the constructed relation is a splitting bisimulation indeed. However, it is not so easy as in the proof of Lemma 9. The most important complication is that we have to verify whether the constructed relation, say R, has the following property: $R((s_1, s_2), (s'_1, s'_2))$ implies $\Phi'_{U,V,W}(s_1, s_2)$ iff $\Phi'_{U,V,W}(s'_1, s'_2)$ for all $U, V, W \subseteq \mathsf{A}$ with $t \in U, V, W$. $\Box$

The *full splitting bisimulation models* $\mathfrak{P}^{\mathrm{cc}\prime}_\kappa$ of $\mathrm{ACP}^{\mathrm{cc}}$, one for each infinite cardinal κ, are the restrictions of the full splitting bisimulation models $\mathfrak{P}^{\mathrm{cc}}_\kappa$ of $\mathrm{PA}^{\mathrm{cc}}_\delta$ to the constants and operators of $\mathrm{BPA}^{\mathrm{cc}}_\delta$ expanded with an adapted operation $\mathbin{\widetilde{\|}'}$ on $\mathbb{CTTS}_\kappa/\leftrightarroweq$ for the operator $\|$. The operation $\mathbin{\widetilde{\|}'}$ is defined as follows:

$$[\,T_1\,]_{\leftrightarroweq} \mathbin{\widetilde{\|}'} [\,T_2\,]_{\leftrightarroweq} = [\,T_1 \mathbin{\widehat{\|}'} T_2\,]_{\leftrightarroweq} \,.$$

The operation $\mathbin{\widetilde{\|}'}$ on $\mathbb{CTTS}_\kappa/\leftrightarroweq$ is well-defined because $\leftrightarroweq$ is a congruence with respect to the operation $\mathbin{\widehat{\|}'}$ on $\mathbb{CTTS}_\kappa$.

The structures $\mathfrak{P}^{\mathrm{cc}\prime}_\kappa$ are models of $\mathrm{ACP}^{\mathrm{cc}}$.

Theorem 19 (Soundness) *For each infinite cardinal κ, we have $\mathfrak{P}_\kappa^{\mathrm{cc}\prime} \models \mathrm{ACP}^{\mathrm{cc}}$.*

PROOF. Because all changes with respect to $\mathfrak{P}_\kappa^{\mathrm{cc}}$ concern the operation associated with the operator $\|$, it is sufficient to show that axiom schema CME from Table 4 is sound. The soundness of CME follows easily from the definitions of the ingredients of $\mathfrak{P}_\kappa^{\mathrm{cc}}$. $\Box$

The axioms of $\mathrm{ACP}^{\mathrm{cc}}$ constitute a complete axiomatization of the full splitting bisimulation models.

Theorem 20 (Completeness) *Let κ be an infinite cardinal. Then we have, for all closed terms p and q of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$, $\mathfrak{P}_\kappa^{\mathrm{cc}} \models p = q$ implies $\mathrm{ACP}^{\mathrm{cc}} \vdash p = q$.*

PROOF. By Lemma 14, for each closed term p of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$, there is a closed term q of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ such that $p = q$ is derivable from the axioms of $\mathrm{ACP}^{\mathrm{cc}}$; and by Lemma 15, all equations between closed terms of sort $\mathbf{P}$ from the language of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ that can be derived from the axioms of $\mathrm{ACP}^{\mathrm{cc}}$ can be derived from the axioms of $\mathrm{BPA}_\delta^{\mathrm{cc}}$. Therefore, it is sufficient to prove that the axioms of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ constitute a complete axiomatization of the restrictions of the full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$ to the constants and operators of $\mathrm{BPA}_\delta^{\mathrm{cc}}$. In the proof of Theorem 11, it is shown that the axioms of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ constitute a complete axiomatization of the restrictions of the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$ to the constants and operators of $\mathrm{BPA}_\delta^{\mathrm{cc}}$. Because the restrictions of the full splitting bisimulation models of $\mathrm{PA}_\delta^{\mathrm{cc}}$ to the constants and operators of $\mathrm{BPA}_\delta^{\mathrm{cc}}$ coincide with the restrictions of the full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$ to the constants and operators of $\mathrm{BPA}_\delta^{\mathrm{cc}}$, the proof is completed. $\Box$

It is easy to see that Theorem 12 goes through for $\mathfrak{P}_\kappa^{\mathrm{cc}\prime}$.

In this section, the full splitting bisimulation models $\mathfrak{P}_\kappa^{\mathrm{cc}}$ of $\mathrm{PA}_\delta^{\mathrm{cc}}$ have been adapted to obtain the full splitting bisimulation models $\mathfrak{P}_\kappa^{\mathrm{cc}\prime}$ of $\mathrm{ACP}^{\mathrm{cc}}$. Henceforth, we loosely write $\mathfrak{P}_\kappa^{\mathrm{cc}}$ for $\mathfrak{P}_\kappa^{\mathrm{cc}\prime}$.

Remark 21 *Associating transition systems with closed terms of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$ by means of structural operational semantics is troublesome. The main problem is to define by means of transition rules, for every $U, V, W \subseteq \mathsf{A}$ with $t \in U, V, W$, a binary relation on closed terms of sort $\mathbf{P}$ that corresponds to $\Phi'_{U,V,W}$ as in the definition of $\widehat{\|}'$ above. This requires, among*

other things, stretching the style of structural operational semantics. Transition rule schemas, i.e. transition rules in which meta-variables standing for arbitrary members of some set occur, are quite usual. However, here a "transition rule schema schema", i.e. a transition rule schema in which "meta-meta-variables" standing for arbitrary members of some set occur, is needed. These meta-meta-variables stand for sets that serve as index sets of indexed families of meta-variables. A complicated side-condition, involving both meta-variables and meta-meta-variables, is needed as well.

8 Guarded Recursion

In order to allow for the description of (potentially) non-terminating processes, we add guarded recursion to $\mathrm{ACP}^{\mathrm{cc}}$.

A *recursive specification* over $\mathrm{ACP}^{\mathrm{cc}}$ is a set of *recursive* equations $E = \{X = t_X \mid X \in V\}$ where V is a set of variables and each t_X is a term of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$ that only contains variables from V. We write $\mathrm{V}(E)$ for the set of all variables that occur on the left-hand side of an equation in E. A *solution* of a recursive specification E is a set of processes (in some model of $\mathrm{ACP}^{\mathrm{cc}}$) $\{P_X \mid X \in \mathrm{V}(E)\}$ such that the equations of E hold if, for all $X \in \mathrm{V}(E)$, X stands for P_X.

Let t be a term of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$ containing a variable X. We call an occurrence of X in t *guarded* if t has a subterm of the form $a \cdot t'$ containing this occurrence of X. A recursive specification over $\mathrm{ACP}^{\mathrm{cc}}$ is called a *guarded* recursive specification if all occurrences of variables in the right-hand sides of its equations are guarded or it can be rewritten to such a recursive specification using the axioms of $\mathrm{ACP}^{\mathrm{cc}}$ and the equations of the recursive specification. We are only interested in models of $\mathrm{ACP}^{\mathrm{cc}}$ in which guarded recursive specifications have unique solutions.

For each guarded recursive specification E and each variable $X \in \mathrm{V}(E)$, we introduce a constant of sort $\mathbf{P}$ standing for the unique solution of E for X. This constant is denoted by $\langle X|E\rangle$. We often write X for $\langle X|E\rangle$ if E is clear from the context. In such cases, it should also be clear from the context that we use X as a constant.

We will also use the following notation. Let t be a term of sort $\mathbf{P}$ from the language of $\mathrm{ACP}^{\mathrm{cc}}$ and E be a guarded recursive specification over $\mathrm{ACP}^{\mathrm{cc}}$. Then we write $\langle t|E\rangle$ for t with, for all $X \in \mathrm{V}(E)$, all occurrences of X in t replaced by $\langle X|E\rangle$.

The additional axioms for recursion are given in Table 5. Both RDP and

Table 5
Axioms for recursion

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

RSP are axiom schemas. Side conditions are added to restrict the variables, terms and guarded recursive specifications for which X, t_X and E stand. The additional axioms for recursion are known as the recursive definition principle (RDP) and the recursive specification principle (RSP). The equations $\langle X|E\rangle = \langle t_X|E\rangle$ for a fixed E express that the constants $\langle X|E\rangle$ make up a solution of E. The conditional equations $E \Rightarrow X = \langle X|E\rangle$ express that this solution is the only one.

In the full splitting bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$, guarded recursive specifications over $\mathrm{ACP}^{\mathrm{cc}}$ have unique solutions.

Theorem 22 (Unique solutions) *For each infinite cardinal κ, guarded recursive specifications over* $\mathrm{ACP}^{\mathrm{cc}}$ *have unique solutions in* $\mathfrak{P}^{\mathrm{cc}}_{\kappa}$.

PROOF. In Ref. [30], a proof of uniqueness of solutions of guarded recursive specifications in the graph models of ACP_{τ} is given. That proof can easily be adapted to the full bisimulation models of ACP introduced in Ref. [28]. The proof consists of the following three steps: (i) proving that two transition systems are bisimilar if at least one of them is finitely branching and all their finite projections are bisimilar; (ii) proving, using the result of step (i), that every guarded recursive specification has a solution that is finitely branching; (iii) proving, using the result of step (i), that the solution from step (ii) is bisimilar to any other solution. Steps (ii) and (iii) remain essentially the same in the case of conditional transition systems and splitting bisimilarity. It is straightforward to define a normal form of elements of $\mathbb{CTS}_{\kappa}$ such that: (a) each element of $\mathbb{CTS}_{\kappa}$ is splitting bisimilar to its normal form and (b) two elements of $\mathbb{CTS}_{\kappa}$ are splitting bisimilar iff their normal forms are bisimilar (cf. the last two paragraphs of Section 3). This enables us to adapt step (i) easily to the case of conditional transition systems and splitting bisimilarity. □

Thus, the full splitting bisimulation models $\mathfrak{P}^{\mathrm{cc}\prime\prime}_{\kappa}$ of $\mathrm{ACP}^{\mathrm{cc}}$ with guarded recursion are simply the expansions of the full splitting bisimulation models $\mathfrak{P}^{\mathrm{cc}}_{\kappa}$ of $\mathrm{ACP}^{\mathrm{cc}}$ obtained by associating with each constant $\langle X|E\rangle$ the unique solution of E for X in the full splitting bisimulation model concerned.

Table 6
Axioms for preferential choice

$\delta \mathbin{+\!\!\!\rightarrow} x = x$	PC1
$a \cdot x \mathbin{+\!\!\!\rightarrow} y = a \cdot x + -\mathcal{E}_a :\!\rightarrow y$	PC2
$(x + y) \mathbin{+\!\!\!\rightarrow} z = x \mathbin{+\!\!\!\rightarrow} (y \mathbin{+\!\!\!\rightarrow} z) + y \mathbin{+\!\!\!\rightarrow} (x \mathbin{+\!\!\!\rightarrow} z)$	PC3
$\phi :\!\rightarrow x \mathbin{+\!\!\!\rightarrow} y = \phi :\!\rightarrow (x \mathbin{+\!\!\!\rightarrow} y) + -\phi :\!\rightarrow y$	PC4

9 Preferential Choice

In the presence of conditional expressions of which the conditions concern the enabledness of actions in the context in which a process is placed, it is easy to give defining equations for a preferential choice operator. In this section, we extend $\mathrm{ACP}^{\mathrm{cc}}$ with the binary *preferential choice* operator $\mathbin{+\!\!\!\rightarrow} : \mathbf{P} \times \mathbf{P} \to \mathbf{P}$.

Let p and q be closed terms of sort $\mathbf{P}$. Intuitively, the preferential choice operator can be explained as follows:

- $p \mathbin{+\!\!\!\rightarrow} q$ behaves as p if the context in which it is placed permits it to behave as p, and as q otherwise.

The additional axioms for preferential choice are the axioms given in Table 6. Axiom PC2 is actually an axiom schema: a stand for an arbitrary element of A.

From the axioms of $\mathrm{ACP}^{\mathrm{cc}}$ and axioms PC1–PC4, we can easily derive that the equation $(\phi{:\!\rightarrow}a{\cdot}x{+}\psi{:\!\rightarrow}b{\cdot}y){\mathbin{+\!\!\!\rightarrow}}z = \phi{:\!\rightarrow}a{\cdot}x{+}\psi{:\!\rightarrow}b{\cdot}y{+}{-}((\phi{\sqcap}\mathcal{E}_a){\sqcup}(\psi{\sqcap}\mathcal{E}_b)){:\!\rightarrow}z$. The following generalization of this result gives a full picture of the preferential choice operator.

Proposition 23 (Characterization) *From the axioms of* $\mathrm{ACP}^{\mathrm{cc}}$ *and axioms PC1–PC4, the following is derivable for all* $n, m \geq 0$*, for all* $a_0, \ldots, a_{n-1}$*,* $b_0, \ldots, b_{m-1} \in \mathsf{A}$*:*

$$\begin{aligned}&\Big(\sum_{i<n} \phi_i :\!\rightarrow a_i \cdot x_i + \sum_{i<m} \psi_i :\!\rightarrow b_i\Big) \mathbin{+\!\!\!\rightarrow} y \\ &\quad = \sum_{i<n} \phi_i :\!\rightarrow a_i \cdot x_i + \sum_{i<m} \psi_i :\!\rightarrow b_i \\ &\qquad + -(\textstyle\bigsqcup_{i<n}(\phi_i \sqcap \mathcal{E}_{a_i}) \sqcup \bigsqcup_{i<m}(\psi_i \sqcap \mathcal{E}_{b_i})) :\!\rightarrow y\ .\end{aligned}$$

PROOF. Easy, by induction on $n + m$. $\square$

A corollary of Proposition 23 is that the preferential choice operator is asso-

ciative.

Corollary 24 (Associativity) *For all closed terms p, q, r of sort* $\mathbf{P}$ *from the language of* $\mathrm{ACP}^{\mathrm{cc}}$ *extended with preferential choice, $p \pri (q \pri r) = (p \pri q) \pri r$ is derivable from the axioms of* $\mathrm{ACP}^{\mathrm{cc}}$ *and axioms PC1–PC4.*

Another corollary of Proposition 23 is that all occurrences of the preferential choice operator can be eliminated from closed terms.

Corollary 25 (Elimination) *For all closed terms p of sort* $\mathbf{P}$ *from the language of* $\mathrm{ACP}^{\mathrm{cc}}$ *extended with preferential choice, there exists a closed term q of sort* $\mathbf{P}$ *from the language of* $\mathrm{ACP}^{\mathrm{cc}}$ *such that $p = q$ is derivable from the axioms of* $\mathrm{ACP}^{\mathrm{cc}}$ *and axioms PC1–PC4.*

Remark 26 *The priority choice operator added to CCS in Ref. [12] is similar to the preferential choice operator of* $\mathrm{ACP}^{\mathrm{cc}}$*, but subject to the restrictions that CCS-style actions must be taken and the initial actions of the operands of the priority choice operator must be input actions. Take CCS-style actions, i.e. $\mathsf{A} = A \cup \{\overline{a} \mid a \in A\} \cup \{t\}$ for some set A such that $t \notin A$. An action $a \in \mathsf{A}$ is called an input action if $a \in A$. Let $a, b \in \mathsf{A} \setminus \{t\}$, and let p and q be terms of sort* $\mathbf{P}$*. If both a and b are input actions, then $a \cdot p \pri b \cdot q$ is written $a \mathbin{.} p \pri b \mathbin{.} q$ with priority choice as well. Otherwise, $a \cdot p \pri b \cdot q$ cannot be written with priority choice. Note that $a \cdot p \pri b \cdot q$ can be written $a \mathbin{.} p + \{a\} : b \mathbin{.} q$ with priority guards, also if not both a and b are input actions.*

Remark 27 *Preferential choices do not need the full expressiveness of* $\mathrm{ACP}^{\mathrm{cc}}$*. The additional expressiveness is among other things found in terms of the form $\mathcal{E}_a \sqcap \mathcal{E}_b :\rightarrow (a \cdot p + b \cdot q)$. Terms of this form express that there is a proper choice: if not both a and b are enabled, then there is not really a choice. The additional expressiveness is also found in terms of the form $a \cdot p + \mathcal{E}_b :\rightarrow c \cdot q$. An example using a term of this form is given in Section 10.*

The full bisimulation models of $\mathrm{ACP}^{\mathrm{cc}}$ with preferential choice are the expansions of the full bisimulation models $\mathfrak{P}^{\mathrm{cc}}$ of $\mathrm{ACP}^{\mathrm{cc}}$ obtained by first associating with the operator $\pri$ a corresponding operation on $\mathbb{CTS}_\kappa$ and then lifting the result of this to $\mathbb{CTS}_\kappa/\leftrightarrows$. This calls for extraction of the initial guarded actions of a conditional transition system.

Let $T = (S, \rightarrow, \rightarrow\surd, s^0) \in \mathbb{CTS}_\kappa$. Then the *initial guarded actions* of T, written $\mathrm{I}(T)$, is the set $\{(\alpha, a) \in \mathcal{C}^- \times \mathsf{A} \mid \exists s \in S \bullet s^0 \xrightarrow{[\alpha]\,a} s \vee s^0 \xrightarrow{[\alpha]\,a} \surd\}$.

We proceed with associating with the operator $\pri$ an operation $\widehat{\pri}$ on $\mathbb{CTS}_\kappa$ as follows.

- Let $T_i = (S_i, \rightarrow_i, \rightarrow\surd_i, s_i^0) \in \mathbb{CTS}_\kappa$ for $i = 1, 2$. Then

$$T_1 \mathbin{\widehat{+\!\!\!\rightarrow}} T_2 = \Gamma(S, \rightarrow, \rightarrow\surd, s^0) \ ,$$

where

$$\begin{aligned} s^0 &= \mathrm{ch}_\kappa(\mathcal{S}_\kappa \setminus (S_1 \uplus S_2)) \ , \\ S &= \{s^0\} \cup (S_1 \uplus S_2) \ , \end{aligned}$$

and for every $(\alpha, a) \in \mathcal{C}^- \times \mathsf{A}$:

$$\begin{aligned} \xrightarrow{(\alpha,a)} &= \Big\{(s^0, \mu_1(s)) \;\Big|\; s_1^0 \xrightarrow{[\alpha]\,a}_1 s\Big\} \\ &\quad \cup \Big\{(s^0, \mu_2(s)) \;\Big|\; \exists \beta \bullet \Big(s_2^0 \xrightarrow{[\beta]\,a}_2 s \wedge \\ &\qquad\qquad \alpha = -(\textstyle\bigsqcup_{(\alpha',a') \in \mathrm{I}(T_1)} (\alpha' \sqcap \mathcal{E}_{a'})) \sqcap \beta\Big)\Big\} \\ &\quad \cup \Big\{(\mu_1(s), \mu_1(s')) \;\Big|\; s \xrightarrow{[\alpha]\,a}_1 s'\Big\} \\ &\quad \cup \Big\{(\mu_2(s), \mu_2(s')) \;\Big|\; s \xrightarrow{[\alpha]\,a}_2 s'\Big\} \ , \\ \xrightarrow{(\alpha,a)}\surd &= \Big\{s^0 \;\Big|\; s_1^0 \xrightarrow{[\alpha]\,a} \surd_1\Big\} \\ &\quad \cup \Big\{s^0 \;\Big|\; \exists \beta \bullet \Big(s_2^0 \xrightarrow{[\alpha]\,a} \surd_2 \wedge \\ &\qquad\qquad \alpha = -(\textstyle\bigsqcup_{(\alpha',a') \in \mathrm{I}(T_1)} (\alpha' \sqcap \mathcal{E}_{a'})) \sqcap \beta\Big)\Big\} \\ &\quad \cup \Big\{\mu_1(s) \;\Big|\; s \xrightarrow{[\alpha]\,a} \surd_1\Big\} \\ &\quad \cup \Big\{\mu_2(s) \;\Big|\; s \xrightarrow{[\alpha]\,a} \surd_2\Big\} \ . \end{aligned}$$

We can show that splitting bisimilarity is a congruence with respect to preferential choice.

Lemma 28 (Congruence) *Let κ be an infinite cardinal. Then for all $T_1, T_2, T_1', T_2' \in \mathbb{CTS}_\kappa$, $T_1 \leftrightarroweq T_1'$ and $T_2 \leftrightarroweq T_2'$ imply $T_1 \mathbin{\widehat{+\!\!\!\rightarrow}} T_2 \leftrightarroweq T_1' \mathbin{\widehat{+\!\!\!\rightarrow}} T_2'$.*

PROOF. Although preferential choice differs from alternative composition (being a non-preferential choice), a witnessing splitting bisimulation can be constructed in the same way as in the proof of Lemma 9. It is straightforward to show that the constructed relation is a splitting bisimulation indeed. As compared with alternative composition, all we have to do more is to show that $T_1 \leftrightarroweq T_1'$ implies $\bigsqcup_{(\alpha',a') \in \mathrm{I}(T_1)} (\alpha' \sqcap \mathcal{E}_{a'}) = \bigsqcup_{(\alpha',a') \in \mathrm{I}(T_1')} (\alpha' \sqcap \mathcal{E}_{a'})$. $\Box$

The operation $\mathbin{\widetilde{+\!\!\!\rightarrow}}$ on $\mathbb{CTS}_\kappa/\!\leftrightarroweq$ is defined as follows:

$$[\,T_1\,]_{\leftrightarroweq} \mathbin{\widetilde{+\!\!\!\rightarrow}} [\,T_2\,]_{\leftrightarroweq} = [\,T_1 \mathbin{\widehat{+\!\!\!\rightarrow}} T_2\,]_{\leftrightarroweq} \ .$$

10 Examples

In this section, we give an example of the use of the preferential choice operator and an example of the use of coordination conditions where the preferential choice operator is inadequate.

The first example is adapted from Ref. [5]. The example concerns a printer with a control panel. The printer will print an infinite sequence $\langle c_0, c_1, c_2, \ldots \rangle$ of characters from a finite set of characters C, but can be interrupted by means of the control panel. The control panel has two buttons, a start button to indicate that the printing must be started and a stop button to indicate that the printing must be stopped. Whenever a button is pushed, a corresponding message is sent to the printer. The printer prints characters from the infinite sequence, but can also receive messages from the control panel. The recursive specification of the control panel is as follows:

$$C = b(start) \cdot s(start) \cdot C + b(stop) \cdot s(stop) \cdot C \;;$$

and the recursive specification of the printer is as follows ($i \geq 0$):

$$\begin{aligned} P &= W_0 \;, \\ W_i &= r(start) \cdot P_i + r(stop) \cdot W_i \;, \\ P_i &= (r(start) \cdot P_i + r(stop) \cdot W_i) \mathbin{+\!\!\rightarrow} p(c_i) \cdot P_{i+1} \;. \end{aligned}$$

In this example, we take $| : \mathsf{A}_\delta \times \mathsf{A}_\delta \to \mathsf{A}_\delta$ such that $s(start) \mid r(start) = c(start)$ and $s(stop) \mid r(stop) = c(stop)$, and in all other cases it yields δ. The whole system is described by $t_I(\partial_H(C \parallel P))$, where $H = \{s(start), r(start), s(stop), r(stop)\}$ and $I = \{c(start), c(stop)\}$.

In the recursive specification of the printer, the preferential choice operator is used in the equation for P_i to describe that receiving a message from the control panel must be given preference to printing a character. It follows from the axioms for preferential choice that the preferential choice operator can be eliminated from this equation. This yields the following equation:

$$P_i = r(start) \cdot P_i + r(stop) \cdot W_i + -(\mathcal{E}_{r(start)} \sqcup \mathcal{E}_{r(stop)}) :\!\rightarrow p(c_i) \cdot P_{i+1} \;.$$

From the axioms of $\mathrm{ACP}^{\mathrm{cc}}$, the additional axioms for preferential choice and RSP, we can derive that $t_I(\partial_H(C \,\|\, P))$ is the solution of the following recursive

specification ($i \geq 0$):

$$\begin{aligned}
Z &= Z_0^w \ , \\
Z_i^w &= b(start) \cdot t \cdot Z_i^p + b(stop) \cdot t \cdot Z_i^w \ , \\
Z_i^p &= b(start) \cdot t \cdot Z_i^p + b(stop) \cdot t \cdot Z_i^w + p(c_i) \cdot Z_{i+1}^p \ .
\end{aligned}$$

This shows that the reaction to button pushing is immediate: printing of characters never takes place between button pushing and the reaction to button pushing.

The second example concerns a sender of messages and two receivers. The sender will send an infinite sequence $\langle m_0, m_1, m_2, \ldots \rangle$ of messages from a finite set of messages M. Each message from M is either secure or not. One receiver, say R, receives only the messages that are secure and files them after receipt. The other receiver, say R', receives only the messages that are not secure and discards them after receipt. Whenever receiver R is ready to receive a message, but the message that the sender is ready to send is not secure, receiver R sends a request to receiver R' to take over the receipt of the message. Receiver R' receives messages only after receipt of a request from receiver R. The recursive specification of the sender is as follows ($i \geq 0$):

$$\begin{aligned}
S &= S_0 \ , \\
S_i &= s(m_i) \cdot S_{i+1} \ ;
\end{aligned}$$

the recursive specifications of the two receivers are as follows:

$$\begin{aligned}
R &= \sum_{m \in SM} r(m) \cdot f(m) \cdot R + \bigsqcup_{m' \in M \setminus SM} \mathcal{E}_{r(m')} :\rightarrow sr \cdot R \ , \\
R' &= rr \cdot \sum_{m' \in M \setminus SM} r(m') \cdot R' \ .
\end{aligned}$$

In this example, we take $| : \mathsf{A}_\delta \times \mathsf{A}_\delta \to \mathsf{A}_\delta$ such that $s(m) \mid r(m) = c(m)$ for all $m \in M$ and $sr \mid rr = cr$, and in all other cases it yields δ. Moreover, we write SM for the set of all messages from M that are secure. The whole system is described by $t_I(\partial_H(S \parallel (R \parallel R')))$, where $H = \{s(m), r(m) \mid m \in M\} \cup \{sr, rr\}$ and $I = \{c(m) \mid m \in M\} \cup \{cr\}$.

From the axioms of $\mathrm{ACP}^{\mathrm{cc}}$ and RSP, we can derive that $t_I(\partial_H(S \parallel (R \parallel R')))$

is the solution of the following recursive specification ($i \geq 0$):

$$\begin{aligned} Z &= Z_0 \ , \\ Z_i &= t \cdot f(m_i) \cdot Z_{i+1} \quad \mathsf{if}\ m_i \in SM \ , \\ Z_i &= t \cdot Z_{i+1} \qquad\qquad \mathsf{if}\ m_i \notin SM \ . \end{aligned}$$

This shows that all messages from the infinite sequence $\langle m_0, m_1, m_2, \ldots \rangle$ are either filed or discarded. In other words, no deadlock occurs.

Note that R cannot be specified by means of the preferential choice operator: the coordination condition needed concerns the enabledness of actions that do not belong to the initial actions of R, whereas the preferential choice operator concerns the disabledness of actions that belong to the initial actions of a process.

11 Concluding Remarks

We have presented ACP$^{\mathrm{cc}}$, an extension of ACP with conditional expressions of which the conditions concern the enabledness of actions in the context in which a process is placed. The presentation includes the axioms of ACP$^{\mathrm{cc}}$ and the main models of ACP$^{\mathrm{cc}}$. To the best of our knowledge, there is almost no work in the field of process algebra on conditions of this kind. The closest related are the priority guards added to CCS in Ref. [13]. However, priority guards are much simpler, are restricted to CCS-style communication, and can only be used in combination with action prefixing. There are several extensions of ACP that include conditional expressions of some kind. ACP$^{\mathrm{cc}}$ is a variant of a recent extension of ACP with conditional expressions called ACP$^{\mathrm{c}}$ [1,31], which in turn is based on earlier extensions of ACP with conditional expressions that can be found in Refs. [20,32,33].

A striking point of ACP$^{\mathrm{cc}}$ is that its axioms do not include axioms similar to the axioms of ACP for parallel composition (axioms CM1–CM9, see e.g. Ref. [3]). Such axioms are too much to expect: the mutual enabling of actions involved in the parallel composition of two processes is a matter which can only be resolved by looking at the processes as a whole. The same need of a global approach makes it troublesome to associate transition systems with closed terms in the style of structural operational semantics.

We do not have a clear notion of the applications of ACP$^{\mathrm{cc}}$. We have treated an application of ACP$^{\mathrm{cc}}$ that remains within the domain of process algebra: the extension of ACP$^{\mathrm{cc}}$ with a preferential choice operator. This operator

generalizes the priority choice operator added to CCS in Ref. [12].

ACP^{cc} includes pre-abstraction, but not abstraction. Abstraction is usually based on observation equivalence [11] or branching bisimulation equivalence [29], which both abstract from both the structure of finitary internal activity and its presence. That way, a process without internal actions can be equivalent to a process with internal actions. This is undesirable in the setting of ACP^{cc}. Orthogonal bisimulation equivalence [34], an equivalence introduced recently, abstracts from the structure of finitary internal activity, but not from its presence. This equivalence looks to be better suited to the setting of ACP^{cc}.

One option for future work is development of an extension of ACP^{cc} with abstraction based on orthogonal bisimulation equivalence. Another option for future work is investigation into ways to combine the kind of conditions considered in ACP^{cc} with other kinds of conditions, in particular with the retrospective conditions of ACP^{cr} [1].

Acknowledgements

We thank three anonymous referees for their helpful suggestions.

References

[1] J. A. Bergstra, C. A. Middelburg, Splitting bisimulations and retrospective conditions, to appear in *Information and Computation*. Preliminary version: Computer Science Report 05-03, Department of Mathematics and Computer Science, Eindhoven University of Technology, January 2005.

[2] J. A. Bergstra, J. W. Klop, Process algebra for synchronous communication, Information and Control 60 (1/3) (1984) 109–137.

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

[4] R. Cleaveland, G. Lüttgen, V. Natarajan, Priority in process algebra, in: J. A. Bergstra, A. Ponse, S. A. Smolka (Eds.), Handbook of Process Algebra, Elsevier, Amsterdam, 2001, pp. 711–765.

[5] J. C. M. Baeten, J. A. Bergstra, J. W. Klop, Syntax and defining equations for an interrupt mechanism in process algebra, Fundamenta Informaticae 9 (1986) 127–168.

[6] INMOS Ltd, occam Reference Manual, Prentice-Hall, 1984.

[7] S. D. Brookes, C. A. R. Hoare, A. W. Roscoe, A theory of communicating sequential processes, Journal of the ACM 31 (3) (1984) 560–599.

[8] C. A. R. Hoare, Communicating Sequential Processes, Prentice-Hall, Englewood Cliffs, 1985.

[9] C. J. Fidge, A formal definition of priority in CSP, ACM Transactions on Programming Languages and Systems 15 (4) (1993) 681–705.

[10] M. Hennessy, R. Milner, Algebraic laws for non-determinism and concurrency, Journal of the ACM 32 (1985) 137–161.

[11] R. Milner, Communication and Concurrency, Prentice-Hall, Englewood Cliffs, 1989.

[12] J. Camilleri, G. Winskel, CCS with priority choice, Information and Computation 116 (1) (1995) 26–37.

[13] I. Phillips, CCS with priority guards, in: K. G. Larsen, M. Nielsen (Eds.), CONCUR 2001, Vol. 2154 of Lecture Notes in Computer Science, Springer-Verlag, 2001, pp. 305–320.

[14] J. D. Monk, R. Bonnet (Eds.), Handbook of Boolean Algebras, Vol. 1, Elsevier, Amsterdam, 1989.

[15] J. C. M. Baeten, J. A. Bergstra, J. W. Klop, Conditional axioms and α/β-calculus in process algebra, in: M. Wirsing (Ed.), Formal Description of Programming Concepts III, North-Holland, 1987, pp. 53–75.

[16] J. C. M. Baeten, J. A. Bergstra, Global renaming operators in concrete process algebra, Information and Control 78 (3) (1988) 205–245.

[17] J. C. M. Baeten, J. A. Bergstra, S. Mauw, G. J. Veltink, A process specification formalism based on static COLD, in: J. A. Bergstra, L. M. G. Feijs (Eds.), Algebraic Methods II: Theory, Tools and Applications, Vol. 490 of Lecture Notes in Computer Science, Springer-Verlag, 1991, pp. 303–335.

[18] M. Wirsing, Algebraic specification, in: J. van Leeuwen (Ed.), Handbook of Theoretical Computer Science, Vol. B, Elsevier, Amsterdam, 1990, pp. 675–788.

[19] D. Sannella, A. Tarlecki, Algebraic preliminaries, in: E. Astesiano, H.-J. Kreowski, B. Krieg-Brückner (Eds.), Algebraic Foundations of Systems Specification, Springer-Verlag, Berlin, 1999, pp. 13–30.

[20] J. C. M. Baeten, J. A. Bergstra, Process algebra with signals and conditions, in: M. Broy (Ed.), Programming and Mathematical Methods, Vol. F88 of NATO ASI Series, Springer-Verlag, 1992, pp. 273–323.

[21] C. A. R. Hoare, I. J. Hayes, He Jifeng, C. C. Morgan, A. W. Roscoe, J. W. Sanders, I. H. Sorensen, J. M. Spivey, B. A. Sufrin, Laws of programming, Communications of the ACM 30 (1987) 672–686.

[22] P. R. Halmos, Lectures on Boolean Algebras, Mathematical Studies, Van Nostrand, Princeton, NJ, 1963.

[23] J. W. Klop, Term rewriting systems, in: S. Abramsky, D. M. Gabbay, T. S. E. Maibaum (Eds.), Handbook of Logic in Computer Science, Vol. II, Oxford University Press, Oxford, 1992, pp. 1–116.

[24] J. A. Bergstra, J. W. Klop, Algebra of communicating processes with abstraction, Theoretical Computer Science 37 (1985) 77–121.

[25] N. Busi, R. J. van Glabbeek, R. Gorrieri, Axiomatising ST-bisimulation semantics, in: E.-R. Olderog (Ed.), PROCOMET'94, Vol. 56 of IFIP Transactions A, North-Holland, 1994, pp. 169–188.

[26] K. G. Larsen, A. Skou, Bisimulation through probabilistic testing, Information and Computation 94 (1) (1991) 1–28.

[27] M. Hennessy, H. Lin, Symbolic bisimulations, Theoretical Computer Science 138 (1995) 353–389.

[28] J. A. Bergstra, C. A. Middelburg, Model theory for process algebra, in: A. Middeldorp, V. van Oostrom, F. van Raamsdonk, R. C. de Vrijer (Eds.), Processes, Terms and Cycles: Steps on the Road to Infinity, Vol. 3838 of Lecture Notes in Computer Science, Springer-Verlag, 2005, pp. 445–495.

[29] R. J. van Glabbeek, W. P. Weijland, Branching time and abstraction in bisimulation semantics, Journal of the ACM 43 (3) (1996) 555–600.

[30] J. C. M. Baeten, J. A. Bergstra, J. W. Klop, On the consistency of Koomen's fair abstraction rule, Theoretical Computer Science 51 (1/2) (1987) 129–176.

[31] J. A. Bergstra, C. A. Middelburg, Strong splitting bisimulation equivalence, in: J. L. Fiadeiro, N. Harman, M. Roggenbach, J. Rutten (Eds.), CALCO 2005, Vol. 3629 of Lecture Notes in Computer Science, Springer-Verlag, 2005, pp. 83–97.

[32] J. A. Bergstra, A. Ponse, J. J. van Wamel, Process algebra with backtracking, in: J. W. de Bakker, W. P. de Roever, G. Rozenberg (Eds.), A Decade of Concurrency (Reflections and Perspectives), Vol. 803 of Lecture Notes in Computer Science, Springer-Verlag, 1994, pp. 46–91.

[33] J. C. M. Baeten, J. A. Bergstra, Process algebra with propositional signals, Theoretical Computer Science 177 (1997) 381–405.

[34] J. A. Bergstra, A. Ponse, M. B. van der Zwaag, Branching time and orthogonal bisimulation equivalence, Theoretical Computer Science 309 (2003) 313–355.