Probabilistic process algebra and strategic interleaving

Middelburg, C.A.

DOI
10.7561/SACS.2020.2.205

Publication date
2020

Document Version
Final published version

Published in
Scientific Annals of Computer Science

License
CC BY-ND

Scientific Annals of Computer Science vol. 30 (2), 2020, pp. 205–243
doi: 10.7561/SACS.2020.2.205

Probabilistic Process Algebra and Strategic Interleaving

C.A. Middelburg[1]

Abstract

We first present a probabilistic version of ACP that rests on the principle that probabilistic choices are always resolved before choices involved in alternative composition and parallel composition are resolved and then extend this probabilistic version of ACP with a form of interleaving in which parallel processes are interleaved according to what is known as a process-scheduling policy in the field of operating systems. We use the term strategic interleaving for this more constrained form of interleaving. The extension covers probabilistic process-scheduling policies.
Keywords: process algebra, probabilistic choice, parallel composition, arbitrary interleaving, strategic interleaving.

1 Introduction

First of all, we present a probabilistic version of ACP [9, 13], called pACP (probabilistic ACP). pACP is a minor variant of the subtheory of pACP_τ [4] in which the operators for abstraction from some set of actions are lacking. It is a minor variant of that subtheory because we take functions whose range is the carrier of a signed cancellation meadow instead of a field as probability measures, add probabilistic choice operators for the probabilities 0

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

and 1, and have an additional axiom because of the inclusion of these operators. The probabilistic choice operators for the probabilities 0 and 1 cause no problem because a meadow has a total multiplicative inverse operation where the multiplicative inverse of zero is zero. Because of this property, we could also improve the operational semantics of pACP. In particular, we could reduce the number of rules for the operational semantics and replace all negative premises by positive premises in the remaining rules.

We also extend pACP with a form of interleaving in which parallel processes are interleaved according to what is known as a process-scheduling policy in the field of operating systems (see e.g. [30, 31]). In [16], we have extended ACP with this more constrained form of interleaving. In that paper, we introduced the term strategic interleaving for this form of interleaving and the term interleaving strategy for process-scheduling policy. Unlike in the extension presented in [16], probabilistic interleaving strategies are covered in the extension presented in the current paper. More precisely, the latter extension assumes a generic interleaving strategy that can be instantiated with different specific interleaving strategies, including probabilistic ones.

A main contribution of this paper to the area of probabilistic process algebra is a semantics of pACP for which the axioms of pACP are sound and complete. For pACP_τ, such a semantic is not available. For pTCP_τ, a variant of pACP_τ, an erroneous semantics is given in [23] (see Section 3.5 for details). This rules out the possibility to derive a semantics of pACP or pACP_τ from this semantics of pTCP_τ. Another contribution of this paper is an extension of pACP with strategic interleaving that covers probabilistic interleaving strategies. The work presented in [16] and this paper is the only work on strategic interleaving in the setting of a general algebraic theory of processes like ACP, CCS and CSP.

The motivation for elaborating upon the work on pACP_τ presented in [4] is that it introduces a parallel composition operator characterized by remarkably simple and natural axioms — axioms that should be backed up by an appropriate semantics. The motivation for considering strategic interleaving in the setting of ACP originates from an important feature of many contemporary programming languages, namely multi-threading (see Section 4.1 for details).

The rest of this paper is organized as follows. First, the theory of signed cancellation meadows is briefly summarized (Section 2). Next, pACP and its extension with guarded recursion, called $\mathrm{pACP}_{\mathrm{rec}}$, is presented (Section 3).

After that, the extension of $\mathrm{pACP}_{\mathrm{rec}}$ with strategic interleaving is presented (Section 4). Finally, we make some concluding remarks (Section 5).

2 Signed Cancellation Meadows

Later in this paper, we will take functions whose range is the carrier of a signed cancellation meadow as probability measures. Therefore, we briefly summarize the theory of signed cancellation meadows in this section.

In [19], meadows are proposed as alternatives for fields with a purely equational axiomatization. Meadows are commutative rings with a multiplicative identity element and a total multiplicative inverse operation where the multiplicative inverse of zero is zero. Fields whose multiplicative inverse operation is made total by imposing that the multiplicative inverse of zero is zero are called zero-totalized fields. All zero-totalized fields are meadows, but not conversely.

Cancellation meadows are meadows that satisfy the *cancellation axiom* $x \neq 0 \land x \cdot y = x \cdot z \Rightarrow y = z$. The cancellation meadows that satisfy in addition the *separation axiom* $0 \neq 1$ are exactly the zero-totalized fields.

Signed cancellation meadows are introduced in [12]. They are cancellation meadows expanded with a signum operation. The signum operation makes it possible that the ordering relations $<$ and $\leq$ of ordered fields are defined (see below).

The signature of signed cancellation meadows consists of the following constants and operators:

- the *additive identity* constant 0;
- the *multiplicative identity* constant 1;
- the binary *addition* operator $+$;
- the binary *multiplication* operator $\cdot$;
- the unary *additive inverse* operator $-$;
- the unary *multiplicative inverse* operator ${}^{-1}$;
- the unary *signum* operator s.

Terms are build as usual. We use prefix notation, infix notation, and postfix notation as usual. We also use the usual precedence convention. We introduce subtraction and division as abbreviations: $t - t'$ abbreviates $t + (-t')$ and t/t' abbreviates $t \cdot (t'^{-1})$.

Signed cancellation meadows are axiomatized by the equations in Tables 1 and 2 and the above-mentioned cancellation axiom.

Table 1: Axioms of a meadow

$(x + y) + z = x + (y + z)$	$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	$(x^{-1})^{-1} = x$
$x + y = y + x$	$x \cdot y = y \cdot x$	$x \cdot (x \cdot x^{-1}) = x$
$x + 0 = x$	$x \cdot 1 = x$	
$x + (-x) = 0$	$x \cdot (y + z) = x \cdot y + x \cdot z$	

Table 2: Additional axioms for the signum operator

$\mathsf{s}(x/x) = x/x$	$\mathsf{s}(x^{-1}) = \mathsf{s}(x)$
$\mathsf{s}(1 - x/x) = 1 - x/x$	$\mathsf{s}(x \cdot y) = \mathsf{s}(x) \cdot \mathsf{s}(y)$
$\mathsf{s}(-1) = -1$	$(1 - \frac{\mathsf{s}(x)-\mathsf{s}(y)}{\mathsf{s}(x)-\mathsf{s}(y)}) \cdot (\mathsf{s}(x + y) - \mathsf{s}(x)) = 0$

The ordering relations $<$ and $\leq$ of ordered fields are defined in signed cancellation meadows as follows:

$$x < y \Leftrightarrow \mathsf{s}(y - x) = 1 ,$$
$$x \leq y \Leftrightarrow \mathsf{s}(\mathsf{s}(y - x) + 1) = 1 .$$

Since $\mathsf{s}(\mathsf{s}(y-x)+1) \neq -1$, we have $0 \leq x \leq 1 \Leftrightarrow \mathsf{s}(\mathsf{s}(x)+1)\cdot\mathsf{s}(\mathsf{s}(1-x)+1) = 1$. We will use this equivalence below to describe the set of probabilities.

In [18], Kolmogorov's probability axioms for finitely additive probability spaces are rephrased for the case where probability measures are functions whose range is the carrier of a signed cancellation meadow.

3 pACP with Guarded Recursion

In this section, we introduce pACP (probabilistic Algebra of Communicating Processes) and guarded recursion in the setting of pACP. The algebraic theory pACP is a minor variant of the subtheory of pACP_τ [4] in which the operators for abstraction from some set of actions are lacking. pACP is a variant of that subtheory because: (a) the range of the functions that are taken as probability measures is the carrier of a signed cancellation meadow in pACP and the carrier of a field in pACP$_\tau$; (b) probabilistic choice operators for the probabilities 0 and 1, together with an axiom concerning these two operators, are found in pACP, but not in pACP$_\tau$. Moreover, a semantics is available for pACP, but not really for pACP$_\tau$.[2]

3.1 pACP

In pACP, it is assumed that a fixed but arbitrary set A of *actions*, with $\delta \notin \mathsf{A}$, has been given. We write A_δ for $\mathsf{A} \cup \{\delta\}$. Related to this, it is assumed that a fixed but arbitrary commutative and associative *communication* function $\gamma : \mathsf{A}_\delta \times \mathsf{A}_\delta \to \mathsf{A}_\delta$, with $\gamma(\delta, a) = \delta$ for all $a \in \mathsf{A}_\delta$, has been given. The function γ is regarded to give the result of synchronously performing any two actions for which this is possible, and to give δ otherwise.

It is also assumed that a fixed but arbitrary signed cancellation meadow $\mathfrak{M}$ has been given. We denote the interpretations of the constants and operators of signed cancellation meadows in $\mathfrak{M}$ by the constants and operators themselves. We write $\mathcal{P}$ for the set $\{\pi \in \mathfrak{M} \mid \mathsf{s}(\mathsf{s}(\pi)+1) \cdot \mathsf{s}(\mathsf{s}(1-\pi)+1) = 1\}$ of *probabilities*.

The signature of pACP consists of the following constants and operators:

- for each $a \in \mathsf{A}$, the *action* constant a ;
- the *inaction* constant δ ;
- the binary *alternative composition* operator $+$;
- the binary *sequential composition* operator $\cdot$;
- for each $\pi \in \mathcal{P}$, the binary *probabilistic choice* operator $\mathbin{+\!\!\!\!\sqcup}_\pi$;
- the binary *parallel composition* operator $\parallel$;
- the binary *left merge* operator $⌊\!⌊$;
- the binary *communication merge* operator $\mid$;
- for each $H \subseteq \mathsf{A}$, the unary *encapsulation* operator ∂_H .

[2]Issues with the semantics of pACP$_\tau$ are discussed in Section 3.5.

We assume that there is a countably infinite set $\mathcal{X}$ of variables, which contains x, y and z, with and without subscripts. Terms are built as usual. We use infix notation for the binary operators. The precedence conventions used with respect to the operators of pACP are as follows: $+$ binds weaker than all others, $\cdot$ binds stronger than all others, and the remaining operators bind equally strong.

The constants and operators of pACP can be explained as follows:

- the constant a denotes the process that can only perform action a and after that terminate successfully;
- the constant δ denotes the process that cannot do anything;
- a closed term of the form $t + t'$ denotes the process that can behave as the process denoted by t or as the process denoted by t', where the choice between the two is resolved exactly when the first action of one of them is performed;
- a closed term of the form $t \cdot t'$ denotes the process that can first behave as the process denoted by t and can next behave as the process denoted by t';
- a closed term of the form $t \uplus_{\pi} t'$ denotes the process that will behave as the process denoted by t with probability π and as the process denoted by t' with probability $1-\pi$, where the choice between the two processes is resolved before the first action of one of them is performed;
- a closed term of the form $t \parallel t'$ denotes the process that can behave as the process that proceeds with the processes denoted by t and t' in parallel;
- a closed term of the form $t \mathbin{⌊\!⌊} t'$ denotes the process that can behave the same as the process denoted by $t \parallel t'$, except that it starts with performing an action of the process denoted by t;
- a closed term of the form $t \mid t'$ denotes the process that can behave the same as the process denoted by $t \parallel t'$, except that it starts with performing an action of the process denoted by t and an action of the process denoted by t' synchronously;
- a closed term of the form $\partial_H(t)$ denotes the process that can behave the same as the process denoted by t, except that actions from H are blocked.

Processes in parallel are considered to be arbitrarily interleaved. With that, probabilistic choices are resolved before interleaving steps are enacted.

The operators $\mathbin{\|\!\lfloor}$ and $\mid$ are of an auxiliary nature. They are needed to axiomatize pACP.

The axioms of pACP are the equations given in Table 3. In these

Table 3: Axioms of pACP

$x + y = y + x$	A1	$x = x + x \wedge y = y + y \Rightarrow$	
$(x + y) + z = x + (y + z)$	A2	$x \parallel y = x \mathbin{\|\!\lfloor} y + y \mathbin{\|\!\lfloor} x + x \mid y$	CM1′
$a + a = a$	A3′	$a \mathbin{\|\!\lfloor} x = a \cdot x$	CM2
$(x + y) \cdot z = x \cdot z + y \cdot z$	A4	$a \cdot x \mathbin{\|\!\lfloor} y = a \cdot (x \parallel y)$	CM3
$(x \cdot y) \cdot z = x \cdot (y \cdot z)$	A5	$(x + y) \mathbin{\|\!\lfloor} z = x \mathbin{\|\!\lfloor} z + y \mathbin{\|\!\lfloor} z$	CM4
$x + \delta = x$	A6	$a \cdot x \mid b = \gamma(a, b) \cdot x$	CM5
$\delta \cdot x = \delta$	A7	$a \mid b \cdot x = \gamma(a, b) \cdot x$	CM6
		$a \cdot x \mid b \cdot y = \gamma(a, b) \cdot (x \parallel y)$	CM7
		$(x + y) \mid z = x \mid z + y \mid z$	CM8
$\partial_H(a) = a$ if $a \notin H$	D1	$x \mid (y + z) = x \mid y + x \mid z$	CM9
$\partial_H(a) = \delta$ if $a \in H$	D2	$\delta \mid x = \delta$	CM10
$\partial_H(x + y) = \partial_H(x) + \partial_H(y)$	D3	$x \mid \delta = \delta$	CM11
$\partial_H(x \cdot y) = \partial_H(x) \cdot \partial_H(y)$	D4	$a \mid b = \gamma(a, b)$	CM12
....		....	
$x \uplus_\pi y = y \uplus_{1-\pi} x$	pA1	$(x \uplus_\pi y) \parallel z = (x \parallel z) \uplus_\pi (y \parallel z)$	pCM1
$(x \uplus_\pi y) \uplus_\rho z =$		$x \parallel (y \uplus_\pi z) = (x \parallel y) \uplus_\pi (x \parallel z)$	pCM2
$x \uplus_{\pi \cdot \rho} (y \uplus_{\frac{(1-\pi) \cdot \rho}{1-\pi \cdot \rho}} z)$	pA2	$(x \uplus_\pi y) \mathbin{\|\!\lfloor} z = (x \mathbin{\|\!\lfloor} z) \uplus_\pi (y \mathbin{\|\!\lfloor} z)$	pCM3
$x \uplus_\pi x = x$	pA3	$x \mathbin{\|\!\lfloor} (y \uplus_\pi z) = (x \mathbin{\|\!\lfloor} y) \uplus_\pi (x \mathbin{\|\!\lfloor} z)$	pCM4
$(x \uplus_\pi y) \cdot z = x \cdot z \uplus_\pi y \cdot z$	pA4	$(x \uplus_\pi y) \mid z = (x \mid z) \uplus_\pi (y \mid z)$	pCM5
$(x \uplus_\pi y) + z = (x + z) \uplus_\pi (y + z)$	pA5	$x \mid (y \uplus_\pi z) = (x \mid y) \uplus_\pi (x \mid z)$	pCM6
$x \uplus_1 y = x$	pA6	$\partial_H(x \uplus_\pi y) = \partial_H(x) \uplus_\pi \partial_H(y)$	pD

equations, a and b stand for arbitrary constants of pACP (which include the action constants and the inaction constant), H stands for an arbitrary subset of A, and π and ρ stand for arbitrary probabilities from $\mathcal{P}$. Moreover, $\gamma(a, b)$ stands for the action constant for the action $\gamma(a, b)$. In D1 and D2, side conditions restrict what a and H stand for.

The equations in Table 3 above the dotted lines, with A3′ replaced by the equation $x + x = x$ and CM1′ replaced by its consequent, constitute an axiomatization of ACP. In presentations of ACP, $\gamma(a, b)$ is regularly replaced by $a \mid b$ in CM5–CM7. By CM12, which is more often called CF, these replacements give rise to an equivalent axiomatization. Moreover, CM10 and CM11 are usually absent. These equations are not derivable from the other axioms, but all their closed substitution instances are derivable from the other axioms and they hold in all models that have been considered for ACP in the literature.

With regard to axioms A3′ and CM1′, we remark that, for each closed term t of pACP, $t = t + t$ is derivable iff t is not derivably equal to a term of the form $t' \uplus_\pi t''$ with $\pi \in \mathcal{P} \setminus \{0, 1\}$. By the completeness result that will be established in Section 3.4, this roughly means that $t = t + t$ is derivable iff the process denoted by t does not have to resolve a probabilistic choice before it can perform its first action.

pACP has pA1, pA3–pA5, pCM1–pCM2, and pD in common with pACP$_\tau$ as presented in [4]. Replacement of axiom pA2 of pACP by axiom pA2 of pACP$_\tau$, that is $x \uplus_\pi (y \uplus_\rho z) = (x \uplus_{\frac{\pi}{\pi+\rho-\pi\cdot\rho}} y) \uplus_{\pi+\rho-\pi\cdot\rho} z$, gives rise to an equivalent axiomatization. In [23], axioms pCM3–pCM6 are presented as axioms of pTCP$_\tau$, a variant of pACP$_\tau$ in which the action constants have been replaced by action prefixing operators and a constant for the process that is only capable of terminating successfully. Therefore, axioms pCM3–pCM6 may be absent in [4] by mistake.

Axiom pA6 is new. Notice that $(x \uplus_0 y) \uplus_0 z = z$ and $x \uplus_0 (y \uplus_0 z) = z$ are derivable from pA1 and pA6. This is consistent with the instance of pA2 where $\pi = \rho = 0$ because in meadows $0/0 = 0$.

In the sequel, we will use the notation $\sum_{i=1}^{n} t_i$, where $n \geq 1$, for right-nested alternative compositions. For each $n \in \mathbb{N}_1$, the term $\sum_{i=1}^{n} t_i$ is defined by induction on n as follows:[3]

$$\textstyle\sum_{i=1}^{1} t_i = t_1 \quad \text{and} \quad \sum_{i=1}^{n+1} t_i = t_1 + \sum_{i=1}^{n} t_{i+1} \; .$$

In addition, we will use the convention that $\sum_{i=1}^{0} t_i = \delta$.

In the sequel, we will also use the notation $\biguplus_{i=1}^{n} [\pi_i]\, t_i$ where $n \geq 1$ and $\sum_{i<n} \pi_i = 1$, for right-nested probabilistic choices. For each $n \in \mathbb{N}_1$, the term $\biguplus_{i=1}^{n} [\pi_i]\, t_i$ is defined by induction on n as follows:

$$\textstyle\biguplus_{i=1}^{1} [\pi_i]\, t_i = t_1 \quad \text{and} \quad \biguplus_{i=1}^{n+1} [\pi_i]\, t_i = t_1 \uplus_{\pi_1} (\biguplus_{i=1}^{n} [\frac{\pi_{i+1}}{1-\pi_1}]\, t_{i+1}) \; .$$

The process denoted by $\biguplus_{i=1}^{n+1} [\pi_i]\, t_i$ will behave like the process denoted by t_1 with probability π_1, ..., and like the process denoted by t_{n+1} with probability π_{n+1}.

[3] We write $\mathbb{N}_1$ for the set $\{n \in \mathbb{N} \mid n \geq 1\}$ of positive natural numbers.

In the next definition, the following summand notation is used. Let t and t' be closed pACP terms. Then we write $t \leq_{+} t'$ for the assertion that $t \equiv t'$ or there exists a closed pACP term t'' such that $t + t'' = t'$ is derivable from axioms A1 and A2 and we write $t \leq_{\boxplus} t'$ for the assertion that $t \equiv t'$ or there exists a closed pACP term t'' and a $\pi \in \mathcal{P} \setminus \{0, 1\}$ such that $t \boxplus_{\pi} t'' = t'$ is derivable from axioms pA1 and pA2.[4]

Each closed pACP term is derivably equal to a proper basic term of pACP. The set $\mathcal{B}$ of *proper basic terms of* pACP is inductively defined, simultaneously with auxiliary sets $\mathcal{B}^0$, $\mathcal{B}^1$, $\mathcal{B}^2$, and $\mathcal{B}^3$, by the following rules:

- $\delta \in \mathcal{B}^0$;
- if $a \in \mathsf{A}$, then $a \in \mathcal{B}^1$;
- if $a \in \mathsf{A}$ and $t \in \mathcal{B}$, then $a \cdot t \in \mathcal{B}^1$;
- if $t \in \mathcal{B}^1$, then $t \in \mathcal{B}^2$;
- if $t \in \mathcal{B}^1$, $t' \in \mathcal{B}^2$, and not $t \leq_{+} t'$, then $t + t' \in \mathcal{B}^2$;
- if $t \in \mathcal{B}^2$, then $t \in \mathcal{B}^3$;
- if $t \in \mathcal{B}^2$, $t' \in \mathcal{B}^3$, not $t \leq_{\boxplus} t'$, and $\pi \in \mathcal{P} \setminus \{0, 1\}$, then $t \boxplus_{\pi} t' \in \mathcal{B}^3$;
- if $t \in \mathcal{B}^0$, then $t \in \mathcal{B}$;
- if $t \in \mathcal{B}^3$, then $t \in \mathcal{B}$.

Proposition 1 *For each* pACP *term t, there exists a proper basic term t' of* pACP *such that $t = t'$ is derivable from the axioms of* pACP*.*

Proof: The proof is straightforward by induction on the structure of t. The case where t is of the form δ and the case where t is of the form a ($a \in \mathsf{A}$) are trivial. The case where t is of the form $t_1 \cdot t_2$ follows immediately from the induction hypothesis (applied to t_1 and t_2) and the claim that, for all proper basic terms t'_1 and t'_2 of pACP, there exists a proper basic term t' of pACP such that $t'_1 \cdot t'_2 = t'$ is derivable from the axioms of pACP. This claim is straightforwardly proved by induction on the structure of t'_1. The cases where t is of the form $t_1 + t_2$, $t_1 \boxplus_{\pi} t_2$, $t_1 \mathbin{\|\!\_} t_2$, $t_1 \mid t_2$ or $\partial_H(t_1)$ are proved in the same vein as the case where t is of the form $t_1 \cdot t_2$. In the case that t is of the form $t_1 \mid t_2$, each of the cases to be considered in the

[4]We write $t \equiv t'$ to indicate that t is syntactically equal to t'.

inductive proof of the claim demands a (nested) proof by induction on the structure of t_2'. The case that t is of the form $t_1 \parallel t_2$ follows immediately from the case that t is of the form $t_1 \mathbin{⌊\!⌊} t_2$ and the case that t is of the form $t_1 \mid t_2$. $\square$

3.2 Guarded Recursion

A closed pACP term denotes a process with a finite upper bound to the number of actions that it can perform. Guarded recursion allows the description of processes without a finite upper bound to the number of actions that it can perform.

The current subsection applies to both pACP and its extension pACP+pSI introduced in Section 4. Therefore, in the current subsection, let *PPA* be pACP or pACP+pSI.

Let t be a *PPA* term containing a variable X. Then an occurrence of X in t is *guarded* if t has a subterm of the form $a \cdot t'$ where $a \in \mathsf{A}$ and t' is a *PPA* term containing this occurrence of X. A *PPA* term t is a *guarded PPA* term if all occurrences of variables in t are guarded.

A *recursive specification* over *PPA* is a set $\{X_i = t_i \mid i \in I\}$, where I is a finite or countably infinite set, each X_i is a variable from $\mathcal{X}$, each t_i is a *PPA* term in which only variables from $\{X_i \mid i \in I\}$ occur, and $X_i \neq X_j$ for all $i, j \in I$ with $i \neq j$. A recursive specification $\{X_i = t_i \mid i \in I\}$ over *PPA* is a *guarded* recursive specification over *PPA* if each t_i is rewritable to a guarded *PPA* term using the axioms of *PPA* in either direction and the equations in $\{X_j = t_j \mid j \in I \wedge i \neq j\}$ from left to right.

We write $\mathrm{V}(E)$, where E is a guarded recursive specification, for the set of all variables that occur in E. The equations occurring in a guarded recursive specification are called *recursion equations*.

A solution of a guarded recursive specification E in some model of *PPA* is a set $\{P_X \mid X \in \mathrm{V}(E)\}$ of elements of the carrier of that model such that the equations of E hold if, for all $X \in \mathrm{V}(E)$, X is assigned P_X. We are only interested in models of *PPA* in which guarded recursive specifications have unique solutions — such as the model presented in Section 3.3.

We extend *PPA* with guarded recursion by adding constants for solutions of guarded recursive specifications over *PPA* and axioms concerning these additional constants. For each guarded recursive specification E over *PPA* and each $X \in \mathrm{V}(E)$, we add a constant standing for the unique solution of E for X to the constants of *PPA*. The constant standing for the unique solution of E for X is denoted by $\langle X|E\rangle$. We use the following notation. Let t be a *PPA* term and E be a guarded recursive specification over *PPA*. 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 $\langle X|E\rangle$. We add the equation RDP and the conditional equation RSP given in Table 4 to the axioms of *PPA*. In RDP and

Table 4: Axioms for guarded recursion

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

RSP, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary *PPA* term, and E stands for an arbitrary guarded recursive specification over *PPA*. Side conditions restrict what X, t and E stand for. We write PPA_{rec} for the resulting theory.

The equations $\langle X|E\rangle = \langle t|E\rangle$ 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.

Because we have to deal with conditional equational formulas with a countably infinite number of premises in PPA_{rec}, it is understood that infinitary conditional equational logic is used in deriving equations from the axioms of PPA_{rec}. A complete inference system for infinitary conditional equational logic can be found in, for example, [33]. It is noteworthy that in the case of infinitary conditional equational logic derivation trees may be infinitely branching (but they may not have infinite branches).

3.3 Semantics of pACP with Guarded Recursion

In this subsection, we present a structural operational semantics of $\mathrm{pACP}_{\mathrm{rec}}$ and define a notion of bisimulation equivalence based on this semantics.

We start with the presentation of a structural operational semantics of $\mathrm{pACP}_{\mathrm{rec}}$. The following relations on closed $\mathrm{pACP}_{\mathrm{rec}}$ terms are used:

- for each $a \in \mathsf{A}$, a unary relation $\xrightarrow{a}\surd$;
- for each $a \in \mathsf{A}$, a binary relation $\xrightarrow{a}$;
- for each $\pi \in \mathcal{P}$, a binary relation $\overset{\pi}{\longmapsto}$.

We write $t \xrightarrow{a}\surd$ for the assertion that $t \in \xrightarrow{a}\surd$, $t \xrightarrow{a} t'$ for the assertion that $(t,t') \in \xrightarrow{a}$, $t \overset{\pi}{\longmapsto} t'$ for the assertion that $(t,t') \in \overset{\pi}{\longmapsto}$, and $t \overset{(0,1]}{\not\longmapsto} t'$ for the assertion that, for all $\pi \in \mathcal{P} \setminus \{0\}$, not $(t,t') \in \overset{\pi}{\longmapsto}$. These assertions can be explained as follows:

- $t \xrightarrow{a} \surd$ indicates that t can perform action a and then terminate successfully;
- $t \xrightarrow{a} t'$ indicates that t can perform action a and then behave as t';
- $t \overset{\pi}{\longmapsto} t'$ indicates that t will behave as t' with probability π;
- $t \overset{(0,1]}{\not\longmapsto} t'$ indicates that t will not behave as t' with a probability greater than zero.

The structural operational semantics of $\mathrm{pACP}_{\mathrm{rec}}$ is described by the rules given in Tables 5 and 6. The rules in Table 5 describe the relations $\xrightarrow{a} \surd$ and the relations $\xrightarrow{a}$ and the rules in Table 6 describe the relations $\overset{\pi}{\longmapsto}$. In these tables, a and b stand for arbitrary actions from A, π, ρ, and ρ' stand for arbitrary probabilities from $\mathcal{P}$, X stands for an arbitrary variable from $\mathcal{X}$, t stands for an arbitrary pACP term, and E stands for an arbitrary guarded recursive specification over pACP.

We could have excluded the relation $\overset{0}{\longmapsto}$ and by that obviated the need for the last rule in Table 6. In that case, however, 11 additional rules concerning the relations $\overset{\pi}{\longmapsto}$, all with negative premises, would be needed instead.

Notice that, if t is not derivably equal to a term whose outermost operator is a probabilistic choice operator, then t can only behave as itself and consequently we have that $t \overset{1}{\longmapsto} t$ and $t \overset{0}{\longmapsto} t'$ for each term t' other than t.

The next two propositions express properties of the relations $\overset{\pi}{\longmapsto}$.

Proposition 2 *For all closed* $\mathrm{pACP}_{\mathrm{rec}}$ *terms* t *and* t', $t \overset{1}{\longmapsto} t'$ *only if* $t \equiv t'$.

Proof: This is easy to prove by induction on the structure of t. □

Proposition 3 *For all closed* $\mathrm{pACP}_{\mathrm{rec}}$ *terms* t *and* t', *there exists a* $\pi \in \mathcal{P}$ *such that* $t \overset{\pi}{\longmapsto} t'$.

Proof: This is easy to prove by induction on the structure of t. □

We define a probability distribution function P from the set of all pairs of closed $\mathrm{pACP}_{\mathrm{rec}}$ terms to $\mathcal{P}$ as follows:

$$P(t,t') = \sum_{\pi \in \Pi(t,t')} \pi \;, \quad \text{where } \Pi(t,t') = \{\pi \mid t \overset{\pi}{\longmapsto} t'\} \;.$$

Table 5: Rules for the operational semantics of $\mathrm{pACP_{rec}}$ (part 1)

$$\frac{}{a \xrightarrow{a} \surd}$$

$$\frac{x \xrightarrow{a} \surd,\ y \overset{1}{\longmapsto} y'}{x + y \xrightarrow{a} \surd} \qquad \frac{x \overset{1}{\longmapsto} x',\ y \xrightarrow{a} \surd}{x + y \xrightarrow{a} \surd} \qquad \frac{x \xrightarrow{a} x',\ y \overset{1}{\longmapsto} y'}{x + y \xrightarrow{a} x'} \qquad \frac{x \overset{1}{\longmapsto} x',\ y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

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

$$\frac{x \xrightarrow{a} \surd,\ y \overset{1}{\longmapsto} y'}{x \parallel y \xrightarrow{a} y} \qquad \frac{x \overset{1}{\longmapsto} x',\ y \xrightarrow{a} \surd}{x \parallel y \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x',\ y \overset{1}{\longmapsto} y'}{x \parallel y \xrightarrow{a} x' \parallel y} \qquad \frac{x \overset{1}{\longmapsto} x',\ y \xrightarrow{a} y'}{x \parallel y \xrightarrow{a} x \parallel y'}$$

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd}{x \parallel y \xrightarrow{\gamma(a,b)} \surd}\ \gamma(a,b) \in \mathsf{A} \qquad \frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} y'}{x \parallel y \xrightarrow{\gamma(a,b)} y'}\ \gamma(a,b) \in \mathsf{A}$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd}{x \parallel y \xrightarrow{\gamma(a,b)} x'}\ \gamma(a,b) \in \mathsf{A} \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y'}{x \parallel y \xrightarrow{\gamma(a,b)} x' \parallel y'}\ \gamma(a,b) \in \mathsf{A}$$

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

$$\frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} \surd}{x \mid y \xrightarrow{\gamma(a,b)} \surd}\ \gamma(a,b) \in \mathsf{A} \qquad \frac{x \xrightarrow{a} \surd,\ y \xrightarrow{b} y'}{x \mid y \xrightarrow{\gamma(a,b)} y'}\ \gamma(a,b) \in \mathsf{A}$$

$$\frac{x \xrightarrow{a} x',\ y \xrightarrow{b} \surd}{x \mid y \xrightarrow{\gamma(a,b)} x'}\ \gamma(a,b) \in \mathsf{A} \qquad \frac{x \xrightarrow{a} x',\ y \xrightarrow{b} y'}{x \mid y \xrightarrow{\gamma(a,b)} x' \parallel y'}\ \gamma(a,b) \in \mathsf{A}$$

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

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

This function can be explained as follows: $P(t, t')$ is the total probability that t will behave as t'.

We write $P(t, T)$, where t is a closed $\mathrm{pACP_{rec}}$ term and T is a set of closed $\mathrm{pACP_{rec}}$ terms, for $\sum_{t' \in T} P(t, t')$.

The well-definedness of P is a corollary of Proposition 3.

Table 6: Rules for the operational semantics of $\mathrm{pACP}_{\mathrm{rec}}$ (part 2)

$$\frac{}{a \overset{1}{\longmapsto} a} \qquad \frac{}{\delta \overset{1}{\longmapsto} \delta}$$

$$\frac{x \overset{\pi}{\longmapsto} x',\ y \overset{\rho}{\longmapsto} y'}{x + y \overset{\pi \cdot \rho}{\longmapsto} x' + y'} \qquad \frac{x \overset{\pi}{\longmapsto} x'}{x \cdot y \overset{\pi}{\longmapsto} x' \cdot y} \qquad \frac{x \overset{\rho}{\longmapsto} z,\ y \overset{\rho'}{\longmapsto} z}{x \sqcup\!\!\!\!\!\;\,{}^{-}_{\pi}\, y \overset{\pi\cdot\rho+(1-\pi)\cdot\rho'}{\longmapsto} z}$$

$$\frac{x \overset{\pi}{\longmapsto} x',\ y \overset{\rho}{\longmapsto} y'}{x \parallel y \overset{\pi \cdot \rho}{\longmapsto} x' \parallel y'} \qquad \frac{x \overset{\pi}{\longmapsto} x',\ y \overset{\rho}{\longmapsto} y'}{x \mathbin{\|\!\_} y \overset{\pi \cdot \rho}{\longmapsto} x' \mathbin{\|\!\_} y'} \qquad \frac{x \overset{\pi}{\longmapsto} x',\ y \overset{\rho}{\longmapsto} y'}{x \mid y \overset{\pi \cdot \rho}{\longmapsto} x' \mid y'}$$

$$\frac{x \overset{\pi}{\longmapsto} x'}{\partial_H(x) \overset{\pi}{\longmapsto} \partial_H(x')} \qquad \frac{\langle t|E\rangle \overset{\pi}{\longmapsto} z}{\langle X|E\rangle \overset{\pi}{\longmapsto} z}\ X = t\ \in\ E$$

$$\frac{x \overset{(0,1]}{\not\longmapsto} x'}{x \overset{0}{\longmapsto} x'}$$

Corollary 1 *Let t and t' be closed* $\mathrm{pACP}_{\mathrm{rec}}$ *terms. Then there exists a unique $\pi \in \mathcal{P}$ such that $P(t, t') = \pi$.*

Moreover, P is actually a probability distribution function.

Proposition 4 *Let T be the set of all closed* $\mathrm{pACP}_{\mathrm{rec}}$ *terms. Then, for all closed* $\mathrm{pACP}_{\mathrm{rec}}$ *terms t, $P(t, T) = 1$.*

Proof: This is easy to prove by induction on the structure of t. □

It follows from Propositions 2 and 4 that the behaviour of t does not start with a probabilistic choice if $t \overset{1}{\longmapsto} t'$. This explains the premises $x \overset{1}{\longmapsto} x'$ and $y \overset{1}{\longmapsto} y'$ in Table 5: they guarantee that probabilistic choices are always resolved before choices involved in alternative composition and parallel composition are resolved.

The relations used in an operational semantics are often called transition relations. It is questionable whether the relations $\overset{\pi}{\longmapsto}$ deserve this name. Recall that $t \overset{\pi}{\longmapsto} t'$ means that t will behave as t' with probability π. It is rather far-fetched to suppose that a transition from t to t' has taken place at the time that t starts to behave as t'. The relations $\overset{\pi}{\longmapsto}$ primarily constitute a representation of the probability distribution function P defined above. This representation turns out to be a convenient one in the setting of structural operational semantics.

In the next paragraph, we write $[t]_R$, where t is a closed $\mathrm{pACP_{rec}}$ term and R is an equivalence relation on closed $\mathrm{pACP_{rec}}$ terms, for the equivalence class of t with respect to R.

A *probabilistic bisimulation* is an equivalence relation R on closed $\mathrm{pACP_{rec}}$ terms such that, for all closed $\mathrm{pACP_{rec}}$ terms t_1, t_2 with $R(t_1, t_2)$, the following conditions hold:

- if $t_1 \xrightarrow{a} t_1'$ for some closed $\mathrm{pACP_{rec}}$ term t_1' and $a \in \mathsf{A}$, then there exists a closed $\mathrm{pACP_{rec}}$ term t_2' such that $t_2 \xrightarrow{a} t_2'$ and $R(t_1', t_2')$;
- if $t_1 \xrightarrow{a} \surd$ for some $a \in \mathsf{A}$, then $t_2 \xrightarrow{a} \surd$;
- $P(t_1, [t]_R) = P(t_2, [t]_R)$ for all closed $\mathrm{pACP_{rec}}$ terms t.

Two closed $\mathrm{pACP_{rec}}$ terms t_1, t_2 are *probabilistic bisimulation equivalent*, written $t_1 \leftrightarrow t_2$, if there exists a probabilistic bisimulation R such that $R(t_1, t_2)$. Let R be a probabilistic bisimulation such that $R(t_1, t_2)$. Then we say that R is a probabilistic bisimulation *witnessing* $t_1 \leftrightarrow t_2$.

The next two propositions state some useful results about $\leftrightarrow$.

Proposition 5 *For all closed* $\mathrm{pACP_{rec}}$ *terms t, $t \leftrightarrow t + t$ only if $t \overset{1}{\mapsto} t$.*

Proof: This follows immediately from the rules for the operational semantics of $\mathrm{pACP_{rec}}$, using that, for all $\pi \in \mathcal{P}$, $\pi \cdot \pi = 1$ iff $\pi = 1$. $\square$

Proposition 6 $\leftrightarrow$ *is the maximal probabilistic bisimulation.*

Proof: It follows from the definition of $\leftrightarrow$ that it is sufficient to prove that $\leftrightarrow$ is a probabilistic bisimulation.

We start with proving that $\leftrightarrow$ is an equivalence relation. The proofs of reflexivity and symmetry are trivial. Proving transitivity amounts to showing that the conditions from the definition of a probabilistic bisimulation hold for the composition of two probabilistic bisimulations. The proofs that the conditions concerning the relations $\xrightarrow{a}$ and $\xrightarrow{a} \surd$ hold are trivial. The proof that the condition concerning the function P holds is also easy using the following easy-to-check property of P: if I is an index set and, for each $i \in I$, T_i is a set of closed $\mathrm{pACP_{rec}}$ terms such that, for all $i, j \in I$ with $i \neq j$, $T_i \cap T_j = \emptyset$, then $P(t, \bigcup_{i \in I} T_i) = \sum_{i \in I} P(t, T_i)$.

We also have to prove that the conditions from the definition of a probabilistic bisimulation hold for $\leftrightarrow$. The proofs that the conditions concerning the relations $\xrightarrow{a}$ and $\xrightarrow{a} \surd$ hold are trivial. The proof that the condition concerning the function P holds is easy knowing the above-mentioned property of P. $\square$

3.4 Soundness and Completeness Results

In this subsection, we present a soundness theorem for $\mathrm{pACP}_{\mathrm{rec}}$ and a completeness theorem for pACP.

We write R^{e}, where R is a binary relation, for the equivalence closure of R.

The following proposition will be used below in the proof of a soundness theorem for $\mathrm{pACP}_{\mathrm{rec}}$.

Proposition 7 $\leftrightarrows$ *is a congruence with respect to the operators of* $\mathrm{pACP}_{\mathrm{rec}}$.

Proof: In this proof, we write $R_1 \diamond R_2$, where R_1 and R_2 are probabilistic bisimulations and $\diamond$ is a binary operator of $\mathrm{pACP}_{\mathrm{rec}}$, for the equivalence relation $\{(t_1 \diamond t_2, t_1' \diamond t_2') \mid R_1(t_1, t_1') \wedge R_2(t_2, t_2')\}$.

Let t_1, t_1', t_2, t_2' be closed $\mathrm{pACP}_{\mathrm{rec}}$ terms such that $t_1 \leftrightarrows t_1'$ and $t_2 \leftrightarrows t_2'$, and let R_1 and R_2 be probabilistic bisimulations witnessing $t_1 \leftrightarrows t_1'$ and $t_2 \leftrightarrows t_2'$, respectively.

For each binary operator $\diamond$ of $\mathrm{pACP}_{\mathrm{rec}}$, we construct an equivalence relation $R_\diamond$ on closed $\mathrm{pACP}_{\mathrm{rec}}$ terms as follows:

in the case that $\diamond$ is $\cdot$: $R_\diamond = ((R_1 \diamond R_2) \cup R_2)^{\mathsf{e}}$;
in the case that $\diamond$ is $+$, $\boxplus_\pi$ or $\|$: $R_\diamond = ((R_1 \diamond R_2) \cup R_1 \cup R_2)^{\mathsf{e}}$;
in the case that $\diamond$ is $\lfloor\!\lfloor$ or $\mid$: $R_\diamond = ((R_1 \diamond R_2) \cup (R_1 \parallel R_2) \cup R_1 \cup R_2)^{\mathsf{e}}$

and for each encapsulation operator ∂_H, we construct an equivalence relation R_{∂_H} on closed $\mathrm{pACP}_{\mathrm{rec}}$ terms as follows:

$$R_{\partial_H} = (\{(\partial_H(t_1), \partial_H(t_1')) \mid R_1(t_1, t_1')\} \cup R_1)^{\mathsf{e}} .$$

For each operator $\diamond$ of $\mathrm{pACP}_{\mathrm{rec}}$, we have to show that the conditions from the definition of a probabilistic bisimulation hold for the constructed relation $R_\diamond$.

The proofs that the conditions concerning the relations $\xrightarrow{a}$ and $\xrightarrow{a}\surd$ hold are easy. The proof that the condition concerning the function P holds is straightforward using the property of P mentioned in the proof of

Proposition 6 and the following easy-to-check properties of P:

$$
\begin{aligned}
P(t \cdot t', T \cdot T') &= 0 && \text{if } t' \notin T' \;, \\
P(t \cdot t', T \cdot T') &= P(t, T) && \text{if } t' \in T' \;, \\
P(t + t', T + T') &= P(t, T) \cdot P(t', T') \;, \\
P(t \uplus_\pi t', T) &= \pi \cdot P(t, T) + (1 - \pi) \cdot P(t', T) \;, \\
P(t \parallel t', T \parallel T') &= P(t, T) \cdot P(t', T') \;, \\
P(t \mathbin{\lfloor\!\lfloor} t', T \mathbin{\lfloor\!\lfloor} T') &= P(t, T) \cdot P(t', T') \;, \\
P(t \mid t', T \mid T') &= P(t, T) \cdot P(t', T') \;, \\
P(\partial_H(t), \partial_H(T)) &= P(t, T) \;,
\end{aligned}
$$

where we write $T \diamond T'$, where T and T' are sets of closed $\mathrm{pACP_{rec}}$ terms and $\diamond$ is a binary operator of $\mathrm{pACP_{rec}}$, for the set $\{t \diamond t' \mid t \in T \wedge t' \in T'\}$ and we write $\partial_H(T)$, where T is a set of closed $\mathrm{pACP_{rec}}$ terms, for the set $\{\partial_H(t) \mid t \in T\}$. □

$\mathrm{pACP^+}$ is the variant of pACP with a different parallel composition operator that is presented in [2, 3].[5] A detailed proof of Proposition 7 is to a large extent a simplified version of the detailed proof of the fact that $\leftrightarroweq$ is a congruence with respect to the operators of $\mathrm{pACP^+}$ that is given in [3]. This is because of the fact that, except for the parallel composition operator, the structural operational semantics of pACP presented in this paper can essentially be obtained from the structural operational semantics of $\mathrm{pACP^+}$ that is presented in [3] by removing unnecessary complexity.

In [27], constraints have been proposed on the form of operational semantics rules which ensure that probabilistic bisimulation equivalence is a congruence. Both the reactive and generative models of probabilistic processes (see [32]) are covered in that paper. While $\mathrm{pACP_{rec}}$ is based on the generative model, virtually all other work in this area covers the reactive model only. Unfortunately, the relations used for the structural operational semantics of $\mathrm{pACP_{rec}}$ differ from the ones used in [27]. The chances are that the structural operational semantics of $\mathrm{pACP_{rec}}$ can be adapted such that the results from that paper can be used to prove Proposition 7. However, it seems quite likely that such a proof requires much more effort than the proof sketched above.

$\mathrm{pACP_{rec}}$ is sound with respect to probabilistic bisimulation equivalence for equations between closed terms.

[5] $\mathrm{pACP^+}$ is called $\mathrm{ACP^+_\pi}$ in [2].

Theorem 1 (Soundness) *For all closed* pACP$_{\mathrm{rec}}$ *terms* t *and* t', $t = t'$ *is derivable from the axioms of* pACP$_{\mathrm{rec}}$ *only if* $t \leftrightarrow t'$.

Proof: Since $\leftrightarrow$ is a congruence for pACP$_{\mathrm{rec}}$, we only need to verify the soundness of each axiom of pACP$_{\mathrm{rec}}$.

For each equational axiom e of pACP$_{\mathrm{rec}}$ (all axioms of pACP$_{\mathrm{rec}}$ except CM1′ and RSP are equational), we construct an equivalence relation R_e on closed pACP$_{\mathrm{rec}}$ terms as follows:

$$R_e = \{(t, t') \mid t = t' \text{ is a closed substitution instance of } e\}^{\mathsf{e}} .$$

For axiom CM1′, we construct an equivalence relation R' on closed pACP$_{\mathrm{rec}}$ terms as follows:

$$R' = \{(t, t') \mid t = t' \text{ is a closed substitution instance of } e \wedge t \overset{1}{\mapsto} t \wedge t' \overset{1}{\mapsto} t'\}^{\mathsf{e}} ,$$

where e is the consequent of CM1′.

For axiom RSP, we take an arbitrary instance $\{X_i = t_i \mid i \in I\} \Rightarrow X_j = \langle X_j | \{X_i = t_i \mid i \in I\}\rangle$ $(j \in I)$ and construct an equivalence relation R'' on closed pACP$_{\mathrm{rec}}$ terms as follows:

$$R'' = \{(\theta(X_j), \langle X_j | \{X_i = t_i \mid i \in I\}\rangle) \mid j \in I \wedge \theta \in \Theta \wedge \textstyle\bigwedge_{i \in I} \theta(X_i) \leftrightarrow \theta(t_i)\}^{\mathsf{e}} ,$$

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

For each equational axiom e of pACP$_{\mathrm{rec}}$, we have to check whether the conditions from the definition of a probabilistic bisimulation hold for the constructed relation R_e. For axiom CM1′, we have to check whether the conditions from the definition of a probabilistic bisimulation hold for the constructed relation R'. That this is sufficient for the soundness of axiom CM1′ follows from Proposition 5. For the instances of axiom RSP, we have to check whether the conditions from the definition of a probabilistic bisimulation hold for the constructed relation R''.

All these checks are straightforward, for the condition concerning the function P, using the following easy-to-check property of P: if β is a bijection on T and $P(t', t) = P(t'', \beta(t))$ for all $t \in T$, then $P(t', T) =$

$P(t'', T)$. □

In versions of ACP where RSP follows from RDP and AIP (Approximation Induction Principle), soundness of RSP follows from soundness of RDP and AIP (see e.g. [9]).

The following three lemmas will be used below in the proof of a completeness theorem for pACP. For convenience, we introduce the notion of a rigid closed pACP term.

A closed pACP term t is *rigid* if, for all probabilistic bisimulations R, $R(t, t)$ only if the restriction of R to the set of all subterms of t is the identity relation on that set.

Lemma 1 *All proper basic terms t of* pACP *are rigid.*

Proof: This is easily proved by induction on the structure of t. □

Lemma 2 *For all rigid closed* pACP *terms t and t', for all probabilistic bisimulations R with $R(t, t')$, the restriction of R to the set of all subterms of t is a bijection.*

Proof: Suppose there exist subterms t_1 and t_2 of t and a subterm t'' of t' such that $R(t_1, t'')$ and $R(t_2, t'')$. Because $R(t, t')$, R^{-1} is a probabilistic bisimulation such that $R^{-1}(t', t)$ and $R^{-1} \circ R$ is a probabilistic bisimulation such that $R^{-1} \circ R(t, t)$. We also have that $R^{-1} \circ R(t_1, t_2)$. Because t is rigid, it follows that $t_1 = t_2$. □

Lemma 3 *For all proper basic term t and t' of* pACP*, there exists a probabilistic bisimulation R with $R(t, t')$ such that the restriction of R to the set of all subterms of t is a bijection only if $t = t'$ is derivable from axioms A1, A2, pA1, and pA2.*

Proof: This is straightforwardly proved by induction on the structure of t. □

Theorem 2 (Completeness) *For all closed* pACP *terms t and t', $t \leftrightarrow t'$ only if $t = t'$ is derivable from the axioms of* pACP*.*

Proof: By Proposition 1 and Theorem 1, it is sufficient to prove the theorem for proper basic terms t and t' of pACP. Assume that $t \leftrightarrow t'$. Then, there exists a probabilistic bisimulation R such that $R(t, t')$. By Lemma 1, t and t' are rigid. So, by Lemma 2, the restriction of R to the set of all subterms of t is a bijection. From this, by Lemma 3, it follows that $t = t'$ is derivable from axioms A1, A2, pA1, and pA2. □

3.5 Remarks Relating to the Semantics of $\mathrm{pACP_{rec}}$

In this subsection, we make some remarks, relating to the operational semantics of $\mathrm{pACP_{rec}}$, that did not fit in very well at an earlier point.

pACP is a minor variant of the subtheory of pACP_τ from [4] in which the operators for abstraction from some set of actions are lacking. Soundness and completeness results with respect to branching bisimulation equivalence of an unspecified operational semantics of pACP_τ are claimed in [4]. In principle, the operational semantics concerned should be derivable from the operational semantics of pTCP_τ given in [23].[6] However, it turns out that a mistake has been made in the rules for the probabilistic choice operators that concern the relations $\overset{\pi}{\longmapsto}$. The mistake concerned manifests only in closed terms of the form $t \mathbin{+\!\!\!\!+}_{1/2} t$. For example, if t is not derivably equal to a term whose outermost operator is a probabilistic choice operator, then both the left-hand side and the right-hand side of $t \mathbin{+\!\!\!\!+}_{1/2} t$ give rise to $t \mathbin{+\!\!\!\!+}_{1/2} t \overset{1/2}{\longmapsto} t$. Consequently, the total probability that $t \mathbin{+\!\!\!\!+}_{1/2} t$ behaves as t is $1/2$ instead of 1. This is counterintuitive and inconsistent with axiom pA3.

A meadow has a total multiplicative inverse operation where the multiplicative inverse of zero is zero. This is why there is no reason to exclude the probabilistic choice operators $\mathbin{+\!\!\!\!+}_\pi$ for $\pi \in \{0,1\}$ if a meadow is used instead of a field. Because we have included these operators, we also have included relations $\overset{\pi}{\longmapsto}$ for $\pi \in \{0,1\}$. As a bonus of the inclusion of these relations, we could achieve that for all pairs (t,t') of closed $\mathrm{pACP_{rec}}$ terms, there exists a $\pi \in \mathcal{P}$ such that $t \overset{\pi}{\longmapsto} t'$. Due to this, we could at the same time reduce the number of rules for the operational semantics that concern the relations $\overset{\pi}{\longmapsto}$, replace all negative premises by positive premises in rules for the operational semantics that concern the relations $\overset{a}{\longrightarrow}$ and $\overset{a}{\longrightarrow}\surd$, and correct the above-mentioned mistake in the rules for the probabilistic choice operators that concern the relations $\overset{\pi}{\longmapsto}$.

We already mentioned that a variant of pACP, called pACP^+, is presented in [2, 3]. pACP, just like pACP_τ from [4], differs from pACP^+ with respect to the parallel composition operator. Moreover, in [2, 3], the probability distribution function is defined directly instead of via the operational semantics. However, except for parallel composition and left merge, the probability distribution function corresponds to the probability distribution function P defined above. The direct definition of the probability distribution function removes the root of the above-mentioned mistake made in [23].

[6] Recall that pTCP_τ is pACP_τ with the action constants replaced by action prefixing operators and a constant for the process that is only capable of terminating successfully.

4 Probabilistic Strategic Interleaving

In this section, we extend pACP with probabilistic strategic interleaving, i.e. interleaving according to some probabilistic interleaving strategy. Interleaving strategies are known as process-scheduling policies in the field of operating systems. A well-known probabilistic process-scheduling policy is lottery scheduling [34]. In the presented extension of pACP deterministic interleaving strategies are special cases of probabilistic interleaving strategies: they are the ones obtained by restriction to the trivial probabilities 0 and 1.

4.1 Motivation for Strategic Interleaving

In this subsection, the motivation for taking strategic interleaving into consideration is given.

The interest in strategic interleaving originates from an important feature of many contemporary programming languages, namely multi-threading. In algebraic theories of processes, such as ACP [9], CCS [28], and CSP [26], processes are discrete behaviours that proceed by doing steps in a sequential fashion. In these theories, parallel composition of two processes is usually interpreted as arbitrary interleaving of the steps of the processes concerned. Arbitrary interleaving turns out to be appropriate for many applications and to facilitate formal algebraic reasoning. Multi-threading as found in programming languages such as Java [24] and C# [25], gives rise to parallel composition of processes. In the case of multi-threading, however, the steps of the processes concerned are interleaved according to what is known as a process-scheduling policy in the field of operating systems.

Arbitrary interleaving and strategic interleaving are quite different. The following points illustrate this: (a) whether the interleaving of certain processes leads to inactiveness depends on the interleaving strategy used; (b) sometimes inactiveness occurs with a particular interleaving strategy whereas arbitrary interleaving would not lead to inactiveness and vice versa. Nowadays, multi-threading is often used in the implementation of systems. Because of this, in many systems, for instance hardware/software systems, we have to do with parallel processes that may best be considered to be interleaved in an arbitrary way as well as parallel processes that may best be considered to be interleaved according to some interleaving strategy. Such applications potentially ask for a process algebra that supports both arbitrary interleaving and strategic interleaving.

4.2 pACP with Probabilistic Strategic Interleaving

In the extension of pACP with probabilistic strategic interleaving presented below, it is expected that an interleaving strategy uses the interleaving history in one way or another to make process-scheduling decisions.

The sets $\mathcal{H}_n$ of *interleaving histories for n processes*, for $n \in \mathbb{N}_1$, are the subsets of $(\mathbb{N}_1 \times \mathbb{N}_1)^*$ that are inductively defined by the following rules:[7]

- $\langle\,\rangle \in \mathcal{H}_n$;
- if $i \leq n$, then $(i, n) \in \mathcal{H}_n$;
- if $h \frown (i, n) \in \mathcal{H}_n$, $j \leq n$, and $n-1 \leq m \leq n+1$, then $h \frown (i, n) \frown (j, m) \in \mathcal{H}_m$.

The intuition concerning interleaving histories is as follows: if the kth pair of an interleaving history is (i, n), then the ith process got a turn in the kth interleaving step and after its turn there were n processes to be interleaved. The number of processes to be interleaved may increase due to process creation (introduced below) and decrease due to successful termination of processes.

The presented extension of pACP is called pACP+pSI (pACP with probabilistic Strategic Interleaving). It covers a generic probabilistic interleaving strategy that can be instantiated with different specific probabilistic interleaving strategies that can be represented in the way that is explained below.

In pACP+pSI, it is assumed that the following has been given:[8]

- a fixed but arbitrary set S;
- a fixed but arbitrary partial function $\sigma_n : \mathcal{H}_n \times S \nrightarrow (\{1, \ldots, n\} \to \mathcal{P})$ for each $n \in \mathbb{N}_1$;
- a fixed but arbitrary total function $\vartheta_n : \mathcal{H}_n \times S \times \{1, \ldots, n\} \times \mathsf{A} \times \{0, 1\} \to S$ for each $n \in \mathbb{N}_1$;
- a fixed but arbitrary set $C \subset \mathsf{A}$;

where, for each $n \in \mathbb{N}_1$:

- for each $h \in \mathcal{H}_n$ and $s \in S$, $\sum_{i=1}^{n} \sigma_n(h, s)(i) = 1$;

[7] The special sequence notation used in this paper is explained in an appendix.

[8] We write $f : A \nrightarrow B$ to indicate that f is a partial function from A to B.

- for each $h \in \mathcal{H}_n$, $s \in S$, $i \in \{1, \ldots, n\}$, and $a \in \mathsf{A} \setminus C$, $\vartheta_n(h, s, i, a, 0) = s$;
- for each $c \in C$, $\overline{c} \in \mathsf{A} \setminus C$ and, for each $a, b \in \mathsf{A}$, $\gamma(a, b) \neq c$, $\gamma(a, b) \neq \overline{c}$, $\gamma(a, c) = \delta$, and $\gamma(a, \overline{c}) = \delta$.

The elements of S are called *control states*, σ_n is called an *abstract scheduler* (*for n processes*), ϑ_n is called a *control state transformer* (*for n processes*), and the elements of C are called *control actions*. The intuition concerning S, σ_n, ϑ_n, and C is as follows:

- the control states from S encode data that are relevant to the interleaving strategy, but not derivable from the interleaving history;
- if $\sigma_n(h, s) = i$, then the ith process gets the next turn after interleaving history h in control state s;
- if $\sigma_n(h, s)$ is undefined, then no process gets the next turn after interleaving history h in control state s;
- if $\vartheta_n(h, s, i, a, 0) = s'$, then s' is the control state that arises from the ith process doing a after interleaving history h in control state s in the case that doing a does not bring the ith process to successful termination;
- if $\vartheta_n(h, s, i, a, 1) = s'$, then s' is the control state that arises from the ith process doing a after interleaving history h in control state s in the case that doing a brings the ith process to successful termination;
- if $a \in C$, then a is an explicit means to bring about a control state change and $\overline{a}$ is left as a trace after a has been dealt with.

Thus, S, $\langle \sigma_n \rangle_{n \in \mathbb{N}_1}$, $\langle \vartheta_n \rangle_{n \in \mathbb{N}_1}$, and C together represent an interleaving strategy. This way of representing an interleaving strategy is engrafted on [29].

Consider the case where S is a singleton set, for each $n \in \mathbb{N}_1$, σ_n is defined by

$$
\begin{aligned}
&\sigma_n(\langle\,\rangle, s)(i) = 1 && \text{if } i = 1\ ,\\
&\sigma_n(\langle\,\rangle, s)(i) = 0 && \text{if } i \neq 1\ ,\\
&\sigma_n(h \frown (j, n), s)(i) = 1 && \text{if } i = (j \bmod n) + 1\ ,\\
&\sigma_n(h \frown (j, n), s)(i) = 0 && \text{if } i \neq (j \bmod n) + 1
\end{aligned}
$$

and, for each $n \in \mathbb{N}_1$, ϑ_n is defined by

$$\vartheta_n(h, s, i, a, f) = s \; .$$

In this case, the interleaving strategy corresponds to the round-robin scheduling algorithm. This deterministic interleaving strategy is called cyclic interleaving in our work on interleaving strategies in the setting of thread algebra (see e.g. [15]). In the current setting, an interleaving strategy is deterministic if, for all $n \in \mathbb{N}_1$, for all $h \in \mathcal{H}_n$, $s \in S$, and $i \in \{1, \dots, n\}$, $\sigma_n(h, s)(i) \in \{0, 1\}$. In the case that S and ϑ_n are as above, but σ_n is defined by

$$\sigma_n(h, s)(i) = 1/n \; ,$$

the interleaving strategy is a purely probabilistic one. The probability distribution used is a uniform distribution.

More advanced strategies can be obtained if the scheduling makes more advanced use of the interleaving history and the control state. The interleaving history may, for example, be used to factor the individual lifetimes of the processes to be interleaved or their creation hierarchy into the process-scheduling decision making. Individual properties of the processes to be interleaved that depend on actions performed by them can be taken into account by making use of the control state. The control state may, for example, be used to factor whether a process is currently waiting to acquire a lock from a process that manages a shared resource into the process-scheduling decision making. An example of a probabilistic interleaving strategy supporting mutual exclusion of critical subprocesses is given in Section 4.5.

In pACP+pSI, it is also assumed that a fixed but arbitrary set D of *data* and a fixed but arbitrary function $\phi : D \to P$, where P is the set of all closed terms over the signature of pACP+pSI (given below), have been given and that, for each $d \in D$ and $a, b \in \mathsf{A}$, $\mathsf{cr}(d), \overline{\mathsf{cr}}(d) \in \mathsf{A}$, $\gamma(\mathsf{cr}(d), a) = \delta$, and $\gamma(a, b) \neq \mathsf{cr}(d)$. The action $\mathsf{cr}(d)$ can be considered a process creation request and the action $\overline{\mathsf{cr}}(d)$ can be considered a process creation act. They represent the request to start the process denoted by $\phi(d)$ in parallel with the requesting process and the act of carrying out that request, respectively.

The signature of pACP+pSI consists of the constants and operators from the signature of pACP and in addition the following operators:

- for each $n \in \mathbb{N}_1$, $h \in \mathcal{H}_n$, and $s \in S$, the n-ary *strategic interleaving* operator $\|^n_{h,s}$;

- for each $n, i \in \mathbb{N}_1$ with $i \leq n$, $h \in \mathcal{H}_n$, and $s \in S$, the n-ary *positional strategic interleaving* operator $\rfloor\!\lfloor^{n,i}_{h,s}$.

The strategic interleaving operators can be explained as follows:

- a closed term of the form $\|^n_{h,s}(t_1, \ldots, t_n)$ denotes the process that results from interleaving of the n processes denoted by $t_1, \ldots, t_n$ after interleaving history h in control state s, according to the interleaving strategy represented by S, $\langle \sigma_n \rangle_{n \in \mathbb{N}_1}$, and $\langle \vartheta_n \rangle_{n \in \mathbb{N}_1}$.

The positional strategic interleaving operators are auxiliary operators used to axiomatize the strategic interleaving operators. The role of the positional strategic interleaving operators in the axiomatization is similar to the role of the left merge operator found in pACP.

The axioms of pACP+pSI are the axioms of pACP and in addition the equations given in Table 7. In the additional equations, n and i stand for

Table 7: Axioms for strategic interleaving

$x_1 = x_1 + x_1 \land \ldots \land x_1 = x_n + x_n \Rightarrow$		
$\quad \|^n_{h,s}(x_1, \ldots, x_n) = \delta$	if $\sigma_n(h, s)$ is undefined	SI0′
$x_1 = x_1 + x_1 \land \ldots \land x_1 = x_n + x_n \Rightarrow$		
$\quad \|^n_{h,s}(x_1, \ldots, x_n) = \biguplus^n_{i=1} [\sigma_n(h, s)(i)]\ \rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_n)$	if $\sigma_n(h, s)$ is defined	SI1′
$\rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, \delta, x_{i+1}, \ldots, x_n) = \delta$		SI2
$\rfloor\!\lfloor^{1,i}_{h,s}(a) = a$		SI3
$\rfloor\!\lfloor^{n+1,i}_{h,s}(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_{n+1}) =$		
$\quad a \cdot \|^n_{h \frown (i,n), \vartheta_{n+1}(h,s,i,a,1)}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{n+1})$		SI4
$\rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, a \cdot x'_i, x_{i+1}, \ldots, x_n) =$		
$\quad a \cdot \|^n_{h \frown (i,n), \vartheta_n(h,s,i,a,0)}(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n)$		SI5
$\rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, \mathsf{cr}(d), x_{i+1}, \ldots, x_n) =$		
$\quad \overline{\mathsf{cr}}(d) \cdot \|^n_{h \frown (i,n), \vartheta_n(h,s,i,\mathsf{cr}(d),1)}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n, \phi(d))$		SI6
$\rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, \mathsf{cr}(d) \cdot x'_i, x_{i+1}, \ldots, x_n) =$		
$\quad \overline{\mathsf{cr}}(d) \cdot \|^{n+1}_{h \frown (i,n+1), \vartheta_n(h,s,i,\mathsf{cr}(d),0)}(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n, \phi(d))$		SI7
$\rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, x'_i + x''_i, x_{i+1}, \ldots, x_n) =$		
$\quad \rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n) + \rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, x''_i, x_{i+1}, \ldots, x_n)$		SI8
$\|^n_{h,s}(x_1, \ldots, x_{i-1}, x'_i \uplus_\pi x''_i, x_{i+1}, \ldots, x_n) =$		
$\quad \|^n_{h,s}(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n) \uplus_\pi \|^n_{h,s}(x_1, \ldots, x_{i-1}, x''_i, x_{i+1}, \ldots, x_n)$		pSI1
$\rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, x'_i \uplus_\pi x''_i, x_{i+1}, \ldots, x_n) =$		
$\quad \rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n) \uplus_\pi \rfloor\!\lfloor^{n,i}_{h,s}(x_1, \ldots, x_{i-1}, x''_i, x_{i+1}, \ldots, x_n)$		pSI2

arbitrary numbers from $\mathbb{N}_1$, h stands for an arbitrary interleaving history from $\mathcal{H}$, s stands for an arbitrary control state from S, a stands for an arbitrary action constant that is not of the form $\mathsf{cr}(d)$ or $\overline{\mathsf{cr}}(d)$, and d stands for an arbitrary datum d from D.

The equations in Table 7 above the dotted line are similar to the axioms for strategic interleaving presented in [16] for the deterministic case. The difference between SI1 from that paper and the consequent of SI1′ is unavoidable because probabilistic interleaving strategies are not covered there. The other differences are due to the finding that the generic interleaving strategy from [16] cannot be instantiated with: (a) interleaving strategies where the data relevant to the process-scheduling decision making may be such that none of the processes concerned can be given a turn, (b) interleaving strategies where the data relevant to the process-scheduling decision making must be updated on successful termination of one of the processes concerned, and (c) interleaving strategies where the process-scheduling decision making may be adjusted by steps of the processes concerned that are solely intended to change the data relevant to the process-scheduling decision making.

Axiom SI2 expresses that, in the event of inactiveness of the process whose turn it is, the whole becomes inactive immediately. A plausible alternative is that, in the event of inactiveness of the process whose turn it is, the whole becomes inactive only after all other processes have terminated or become inactive. In that case, the functions $\vartheta_n{:}\mathcal{H}\times S\times\{1,\dots,n\}\times\mathsf{A}\times\{0,1\}\to S$ must be extended to functions $\vartheta_n{:}\mathcal{H}\times S\times\{1,\dots,n\}\times(\mathsf{A}\cup\{\delta\})\times\{0,1\}\to S$ and axiom SI2 must be replaced by the axioms in Table 8.

Table 8: Alternative axioms for SI2

$\rrbracket^{1,i}_{h,s}(\delta)=\delta$	SI2a
$\rrbracket^{n+1,i}_{h,s}(x_1,\dots,x_{i-1},\delta,x_{i+1},\dots,x_{n+1})=$ $\quad\Vert^{n}_{h\frown(i,n),\vartheta_{n+1}(h,s,i,\delta,0)}(x_1,\dots,x_{i-1},x_{i+1},\dots,x_{n+1})\cdot\delta$	SI2b

In $(\text{pACP+pSI})_{\text{rec}}$, i.e. pACP+pSI extended with guarded recursion in the way described in Section 3.2, the processes that can be created are restricted to the ones denotable by a closed pACP+pSI term. This restriction stems from the requirement that ϕ is a function from D to the set of all closed pACP+pSI terms. The restriction can be removed by relaxing this requirement to the requirement that ϕ is a function from D to the set of all closed $(\text{pACP+pSI})_{\text{rec}}$ terms. We write $(\text{pACP+pSI})^{+}_{\text{rec}}$ for the theory resulting from this relaxation. In other words, $(\text{pACP+pSI})^{+}_{\text{rec}}$ differs from

$(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$ in that it is assumed that a fixed but arbitrary function $\phi : D \to P$, where P is the set of all closed terms over the signature of $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$, has been given.

4.3 Semantics of pACP+pSI with Guarded Recursion

In this subsection, we present a structural operational semantics of pACP+pSI with guarded recursion.

The structural operational semantics of $(\mathrm{pACP{+}pSI})^{+}_{\mathrm{rec}}$ is described by the rules for the operational semantics of $\mathrm{pACP}_{\mathrm{rec}}$ (given in Tables 5 and 6) and in addition the rules given in Table 9. In the additional rules, n and i

Table 9: Additional rules for the operational semantics of $(\mathrm{pACP{+}pSI})^{+}_{\mathrm{rec}}$

$$\frac{x \xrightarrow{a} \surd}{\rfloor\!\lfloor^{1,1}_{h,s}(x) \xrightarrow{a} \surd}$$

$$\frac{x_1 \xmapsto{1} x'_1,\ \ldots,\ x_{i-1} \xmapsto{1} x'_{i-1},\ x_i \xrightarrow{a} \surd,\ x_{i+1} \xmapsto{1} x'_{i+1},\ \ldots,\ x_{n+1} \xmapsto{1} x'_{n+1}}{\rfloor\!\lfloor^{n+1,i}_{h,s}(x_1,\ldots,x_{n+1}) \xrightarrow{a} \|^{n}_{h\frown(i,n),\vartheta_{n+1}(h,s,i,a,1)}(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_{n+1})}$$

$$\frac{x_1 \xmapsto{1} x'_1,\ \ldots,\ x_{i-1} \xmapsto{1} x'_{i-1},\ x_i \xrightarrow{a} x'_i,\ x_{i+1} \xmapsto{1} x'_{i+1},\ \ldots,\ x_n \xmapsto{1} x'_n}{\rfloor\!\lfloor^{n,i}_{h,s}(x_1,\ldots,x_n) \xrightarrow{a} \|^{n}_{h\frown(i,n),\vartheta_n(h,s,i,a,0)}(x_1,\ldots,x_{i-1},x'_i,x_{i+1},\ldots,x_n)}$$

$$\frac{x_1 \xmapsto{1} x'_1,\ \ldots,\ x_{i-1} \xmapsto{1} x'_{i-1},\ x_i \xrightarrow{\mathsf{cr}(d)} \surd,\ x_{i+1} \xmapsto{1} x'_{i+1},\ \ldots,\ x_n \xmapsto{1} x'_n}{\rfloor\!\lfloor^{n,i}_{h,s}(x_1,\ldots,x_n) \xrightarrow{\overline{\mathsf{cr}}(d)} \|^{n}_{h\frown(i,n),\vartheta_n(h,s,i,\mathsf{cr}(d),1)}(x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_n,\phi(d))}$$

$$\frac{x_1 \xmapsto{1} x'_1,\ \ldots,\ x_{i-1} \xmapsto{1} x'_{i-1},\ x_i \xrightarrow{\mathsf{cr}(d)} x'_i,\ x_{i+1} \xmapsto{1} x'_{i+1},\ \ldots,\ x_n \xmapsto{1} x'_n}{\rfloor\!\lfloor^{n,i}_{h,s}(x_1,\ldots,x_n) \xrightarrow{\overline{\mathsf{cr}}(d)} \|^{n+1}_{h\frown(i,n+1),\vartheta_n(h,s,i,\mathsf{cr}(d),0)}(x_1,\ldots,x_{i-1},x'_i,x_{i+1},\ldots,x_n,\phi(d))}$$

$$\frac{x_1 \xmapsto{\pi_1} x'_1,\ \ldots,\ x_n \xmapsto{\pi_n} x'_n}{\|^{n}_{h,s}(x_1,\ldots,x_n) \xmapsto{\sigma_n(h,s)(i)\cdot\pi_1\cdot\ldots\cdot\pi_n} \rfloor\!\lfloor^{n,i}_{h,s}(x'_1,\ldots,x'_n)}\quad \sigma_n(h,s) \text{ is defined}$$

$$\frac{x_1 \xmapsto{\pi_1} x'_1,\ \ldots,\ x_n \xmapsto{\pi_n} x'_n}{\rfloor\!\lfloor^{n,i}_{h,s}(x_1,\ldots,x_n) \xmapsto{\pi_1\cdot\ldots\cdot\pi_n} \rfloor\!\lfloor^{n,i}_{h,s}(x'_1,\ldots,x'_n)}$$

stand for arbitrary numbers from $\mathbb{N}_1$, h stands for an arbitrary interleaving history from $\mathcal{H}$, s stands for an arbitrary control state from S, a stands for an arbitrary action from A that is not of the form $\mathsf{cr}(d)$ or $\overline{\mathsf{cr}}(d)$, d stands for an arbitrary datum d from D, and $\pi_1, \ldots, \pi_n$ stand for arbitrary probabilities from $\mathcal{P}$.

Proposition 8 $\leftrightarrows$ *is a congruence w.r.t. the operators of* $(\mathrm{pACP{+}pSI})^{+}_{\mathrm{rec}}$.

Proof: The proof goes along the same line as the proof of Proposition 7 $\square$

$(\mathrm{pACP{+}pSI})^{+}_{\mathrm{rec}}$ is sound with respect to probabilistic bisimulation equivalence for equations between closed terms.

Theorem 3 (Soundness) *For all closed* $(\mathrm{pACP{+}pSI})^{+}_{\mathrm{rec}}$ *terms* t *and* t', $t = t'$ *is derivable from the axioms of* $(\mathrm{pACP{+}pSI})^{+}_{\mathrm{rec}}$ *only if* $t \leftrightarrows t'$.

Proof: The proof goes along the same line as the proof of Theorem 1. $\square$

4.4 Reduction of Guarded Recursive Specifications over pACP+pSI

In this subsection, we show that each guarded recursive specifications over pACP+pSI can be reduced to a guarded recursive specification over pACP. We make use of the fact that each guarded pACP+pSI term has a head normal form.

Let T be pACP+pSI or $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$. The set *HNF* of *head normal forms of* T is inductively defined by the following rules:

- $\delta \in \mathit{HNF}$;
- if $a \in \mathsf{A}$, then $a \in \mathit{HNF}$;
- if $a \in \mathsf{A}$ and t is a T term, then $a \cdot t \in \mathit{HNF}$;
- if $t, t' \in \mathit{HNF}$, then $t + t' \in \mathit{HNF}$;
- if $t, t' \in \mathit{HNF}$ and $\pi \in \mathcal{P}$, then $t \uplus_{\pi} t' \in \mathit{HNF}$.

Each head normal form of T is derivably equal to a head normal form of the form $\biguplus_{i=1}^{n} [\pi_i]\, s_i$, where $n \in \mathbb{N}_1$ and, for each $i \in \mathbb{N}_1$ with $i \leq n$, s_i is of the form $\sum_{j=1}^{n_i} a_{ij} \cdot t_{ij} + \sum_{k=1}^{m_i} b_{ik}$, where $n_i, m_i \in \mathbb{N}_1$ and, for all $j \in \mathbb{N}_1$ with $j \leq n_i$, $a_{ij} \in \mathsf{A}$ and t_{ij} is a T term, and, for all $k \in \mathbb{N}_1$ with $k \leq m_i$, $b_{ik} \in \mathsf{A}$.

Each guarded $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$ term is derivably equal to a head normal form of $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$.

Proposition 9 *For each guarded* $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$ *term* t*, there exists a head normal form* t' *of* $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$ *such that* $t = t'$ *is derivable from the axioms of* $(\mathrm{pACP{+}pSI})_{\mathrm{rec}}$*.*

Proof: First we prove the following weaker result about head normal forms:

> *For each guarded* pACP+pSI *term* t*, there exists a head normal form* t' *of* pACP+pSI *such that* $t = t'$ *is derivable from the axioms of* pACP+pSI.

The proof is straightforward by induction on the structure of t. The case where t is of the form δ and the case where t is of the form a ($a \in \mathsf{A}$) are trivial. The case where t is of the form $t_1 \cdot t_2$ follows immediately from the induction hypothesis (applied to t_1) and the claim that, for all head normal forms t'_1 and t'_2 of pACP+pSI, there exists a head normal form t' of pACP+pSI such that $t'_1 \cdot t'_2 = t'$ is derivable from the axioms of pACP+pSI. This claim is easily proved by induction on the structure of t'_1. The cases where t is of the form $t_1 + t_2$ or $t_1 \uplus_\pi t_2$ follow immediately from the induction hypothesis. The cases where t is of one of the forms $t_1 \mathbin{⌊\!⌊} t_2$, $t_1 \mid t_2$ or $\partial_H(t_1)$ are proved in the same vein as the case where t is of the form $t_1 \cdot t_2$. In the case that t is of the form $t_1 \mid t_2$, each of the cases to be considered in the inductive proof of the claim demands a (nested) proof by induction on the structure of t'_2. The case that t is of the form $t_1 \parallel t_2$ follows immediately from the case that t is of the form $t_1 \mathbin{⌊\!⌊} t_2$ and the case that t is of the form $t_1 \mid t_2$. The case where t is of the form $\rfloor\!\lfloor^{n,i}_{h,s}(t_1, \ldots, t_n)$ is proved in the same vein as the case where t is of the form $t_1 \cdot t_2$, but the claim is of course proved by induction on the structure of t'_i instead of t'_1. The case that t is of the form $\parallel^n_{h,s}(t_1, \ldots, t_n)$ follows immediately from the case that t is of the form $\rfloor\!\lfloor^{n,i}_{h,s}(t_1, \ldots, t_n)$. Because t is a guarded pACP+pSI term, the case where t is a variable cannot occur.

The proof of the proposition itself is also straightforward by induction on the structure of t. The cases other than the case where t is of the form $\langle X|E\rangle$ is proved in the same way as in the above proof of the weaker result. The case where t is of the form $\langle X|E\rangle$ follows immediately from the weaker result and RDP. □

The following theorem refers to three process algebras. It is implicit that the same set A of actions and the same communication function γ are assumed in the process algebras referred to.

Each guarded recursive specification over pACP+pSI can be reduced to a guarded recursive specification over pACP.

Theorem 4 (Reduction) *For each guarded recursive specification E over* pACP+pSI *and each $X \in \mathrm{V}(E)$, there exists a guarded recursive specification E' over* pACP *such that $\langle X|E\rangle = \langle X|E'\rangle$ is derivable from the axioms of* $(\text{pACP+pSI})_{\text{rec}}$.

Proof: We start with devising an algorithm to construct the guarded recursive specification E'. The algorithm keeps a set V of recursion equations from E' that are already found and a sequence W of equations of the form $X_k = \langle t_k|E\rangle$ that still have to be transformed. The algorithm has a finite or countably infinite number of stages. In each stage, V and W are finite. Initially, V is empty and W contains only the equation $X_0 = \langle X|E\rangle$.

In each stage, we remove the first equation from W. Assume that this equation is $X_k = \langle t_k|E\rangle$. We bring the term $\langle t_k|E\rangle$ into head normal form. If t_k is not a guarded term, then we use RDP here to turn t_k into a guarded term first. Thus, by Proposition 9, we can always bring $\langle t_k|E\rangle$ into head normal form. Assume that the resulting head normal form is $\biguplus_{i=1}^{n} [\pi_i]\,(\sum_{j=1}^{n_i} a_{ij} \cdot t'_{ij} + \sum_{k=1}^{m_i} b_{ik})$. Then, we add the equation $X_k = \biguplus_{i=1}^{n} [\pi_i]\,(\sum_{j=1}^{n_i} a_{ij} \cdot X_{k+(\sum_{i'=1}^{i} n_{i'})+j} + \sum_{k=1}^{m_i} b_{ik})$, where the $X_{k+(\sum_{i'=1}^{i} n_{i'})+j}$ are fresh variables, to the set V. Moreover, for each i and j such that $1 \leq i \leq n$ and $1 \leq j \leq n_i$, we add the equation $X_{k+(\sum_{i'=1}^{i} n_{i'})+j} = t'_{ij}$ to the end of the sequence W. Notice that the terms t'_{ij} are of the form $\langle t_{k+(\sum_{i'=1}^{i} n_{i'})+j}|E\rangle$.

Because V grows monotonically, there exists a limit. That limit is the finite or countably infinite guarded recursive specification E'. Every equation that is added to the finite sequence W, is also removed from it. Therefore, the right-hand side of each equation from E' only contains variables that also occur as the left-hand side of an equation from E'.

Now, we want to use RSP to show that $\langle X|E\rangle = \langle X|E'\rangle$ is derivable from the axioms of $(\text{pACP+pSI})_{\text{rec}}$. The variables occurring in E' are $X_0, X_1, X_2, \ldots$. For each k, the variable X_k has been exactly once in W as the left-hand side of an equation. For each k, assume that this equation is $X_k = \langle t_k|E\rangle$. To use RSP, we have to show for each k that the equation $X_k = \biguplus_{i=1}^{n} [\pi_i]\,(\sum_{j=1}^{n_i} a_{ij} \cdot X_{k+(\sum_{i'=1}^{i} n_{i'})+j} + \sum_{k=1}^{m_i} b_{ik})$, with, for each l, all occurrences of X_l replaced by $\langle t_l|E\rangle$, is derivable from the axioms of $(\text{pACP+pSI})_{\text{rec}}$. For each k, this follows from the construction. $\square$

Theorem 4 would not hold if guarded recursive specifications were restricted to finite sets of recursion equations.

Let t be a closed pACP term or a closed pACP+pSI term, and let $X \in \mathcal{X}$. Then $\langle X | \{X = t\} \rangle = t$ is derivable from RDP. This gives rise to the following corollary of Theorem 4.

Corollary 2 *For each closed* $(\text{pACP+pSI})_{\text{rec}}$ *term* t*, there exists a closed* pACP_{rec} *term* t' *such that* $t = t'$ *is derivable from the axioms of* $(\text{pACP+pSI})_{\text{rec}}$.

4.5 An Example

In this subsection, we instantiate the generic interleaving strategy on which pACP+pSI is based with a specific interleaving strategy. The interleaving strategy concerned corresponds to a scheduling algorithm that:

- selects randomly, according to a uniform probability distribution, the next process that gets turns to perform an action;
- gives the selected process a fixed number k of consecutive turns to perform an action;
- takes care of mutual exclusion of critical subprocesses of the different processes being interleaved.

Mutual exclusion of certain subprocesses is the condition that they are not interleaved and critical subprocesses are subprocesses that possibly interfere with each other when this condition is not met. The adopted mechanism for mutual exclusion is essentially a binary semaphore mechanism [10, 20, 21]. Below binary semaphores are simply called *semaphores*.

In this section, it is assumed that a fixed but arbitrary natural number $k \in \mathbb{N}_1$ has been given. We use k as the number of consecutive turns that each process being interleaved gets to perform an action.

Moreover, it is assumed that a finite set R of semaphores has been given. We instantiate the set C of control actions as follows:

$$C = \{\mathsf{wait}(r) \mid r \in R\} \cup \{\mathsf{signal}(r) \mid r \in R\} \;,$$

hereby taking for granted that C satisfies the necessary conditions. The wait and signal actions correspond to the P and V operations from [21].

We instantiate the set S of control states as follows:

$$S = \textstyle\bigcup_{R' \subseteq R} (R' \to {\mathbb{N}_1}^*) \;.$$

The intuition concerning the connection between control states $s \in S$ and the semaphore mechanism as introduced in [21] is as follows:

- $r \notin \mathrm{dom}(s)$ indicates that semaphore r has the value 1;
- $r \in \mathrm{dom}(s)$ indicates that semaphore r has the value 0;
- $r \in \mathrm{dom}(s)$ and $s(r) = \langle\,\rangle$ indicates that no process is suspended on semaphore r;
- if $r \in \mathrm{dom}(s)$ and $s(r) \neq \langle\,\rangle$, then $s(r)$ represents a first-in, first-out queue of processes suspended on r.

As a preparation for the instantiation of the abstract schedulers σ_n and control state transformers ϑ_n, we define some auxiliary functions.

We define a total function $\mathit{turns} : \mathcal{H} \times \mathbb{N}_1 \to \mathbb{N}$ recursively as follows:

$$\begin{array}{ll} \mathit{turns}(\langle\,\rangle, i) = 0 \;, & \\ \mathit{turns}(h \frown (j, n), i) = 0 & \textsf{if } i \neq j \;, \\ \mathit{turns}(h \frown (j, n), i) = \mathit{turns}(h, i) + 1 & \textsf{if } i = j \;. \end{array}$$

If $\mathit{turns}(h, i) = l$ and $l > 0$, then the interleaving history h ends with l consecutive turns of the ith process being interleaved. If $\mathit{turns}(h, i) = 0$, then the interleaving history h does not end with turns of the ith process being interleaved.

We define a total function $\mathit{waiting} : S \to \mathcal{P}(\mathbb{N}_1)$ as follows:

$$\mathit{waiting}(s) = \textstyle\bigcup_{r \in \mathrm{dom}(s)} \mathrm{elems}(s(r)) \;.$$

If $\mathit{waiting}(s) = I$, then $i \in I$ iff the ith process being interleaved is suspended on one or more semaphores in control state s.

We define a total function $\mathit{time2switch}_n : \mathcal{H} \times S \to \{0, 1\}$, for each $n \in \mathbb{N}_1$, as follows:

$$\begin{array}{ll} \mathit{time2switch}_n(h, s) = 1 & \textsf{if } \sum_{i \in \{1,\dots,n\} \setminus \mathit{waiting}(s)} \mathit{turns}(h, i) \in \{0, k\} \;, \\ \mathit{time2switch}_n(h, s) = 0 & \textsf{if } \sum_{i \in \{1,\dots,n\} \setminus \mathit{waiting}(s)} \mathit{turns}(h, i) \notin \{0, k\} \;. \end{array}$$

If $\mathit{time2switch}_n(h, s) = b$, then $b = 1$ iff the interleaving history h ends with a number of consecutive turns of some process that equals k if that process is not suspended in control state s.

We define a partial function $\mathit{sched}_n : \mathcal{H} \times S \nrightarrow (\{1, \dots, n\} \to \mathcal{P})$, for each $n \in \mathbb{N}_1$, as follows:

$$
\begin{aligned}
&\mathit{sched}_n(h,s)(i) = 1/(n - \mathrm{card}(\mathit{waiting}(s))) \\
&\qquad \mathsf{if}\ \mathit{time2switch}_n(h,s) = 1 \land i \notin \mathit{waiting}(s) \land \mathit{waiting}(s) \neq \{1,\ldots,n\}\ , \\
&\mathit{sched}_n(h,s)(i) = 0 \\
&\qquad \mathsf{if}\ \mathit{time2switch}_n(h,s) = 1 \land i \in \mathit{waiting}(s) \land \mathit{waiting}(s) \neq \{1,\ldots,n\}\ , \\
&\mathit{sched}_n(h,s)(i) = 1 \\
&\qquad \mathsf{if}\ \mathit{time2switch}_n(h,s) = 0 \land \mathit{turns}(h,i) \neq 0 \land \mathit{waiting}(s) \neq \{1,\ldots,n\}\ , \\
&\mathit{sched}_n(h,s)(i) = 0 \\
&\qquad \mathsf{if}\ \mathit{time2switch}_n(h,s) = 0 \land \mathit{turns}(h,i) = 0 \land \mathit{waiting}(s) \neq \{1,\ldots,n\}\ .
\end{aligned}
$$

The function sched_n represents a scheduler that works as follows: when a process has been given k consecutive turns to perform an action or has been suspended, the next process that is given turns is randomly selected, according to a uniform probability distribution, from the processes being interleaved that are not suspended. Notice that $\mathit{sched}_n(h,s)(i)$ is undefined if $\mathit{waiting}(s) = \{1,\ldots,n\}$. In that case, none of the processes being interleaved can be given a turn and the whole becomes inactive.

We define a total function $\mathit{remove}_n : S \times \{1,\ldots,n\} \to S$ recursively as follows:[9]

$$
\begin{aligned}
&\mathit{remove}_n([\,],i) = [\,]\ , \\
&\mathit{remove}_n(s \dagger [r \mapsto q], i) = \mathit{remove}_n(s,i) \dagger [r \mapsto \mathit{remove}'_n(q,i)]\ ,
\end{aligned}
$$

where the total function $\mathit{remove}'_n : {\mathbb{N}_1}^* \times \{1,\ldots,n\} \to {\mathbb{N}_1}^*$ is recursively defined as follows:

$$
\begin{aligned}
&\mathit{remove}'_n(\langle\,\rangle,i) = \langle\,\rangle\ , \\
&\mathit{remove}'_n(j \curvearrowright q,i) = j \curvearrowright \mathit{remove}'_n(q,i) && \mathsf{if}\ j < i\ , \\
&\mathit{remove}'_n(j \curvearrowright q,i) = \mathit{remove}'_n(q,i) && \mathsf{if}\ j = i\ , \\
&\mathit{remove}'_n(j \curvearrowright q,i) = (j-1) \curvearrowright \mathit{remove}'_n(q) && \mathsf{if}\ j > i\ .
\end{aligned}
$$

If $\mathit{remove}_n(s,i) = s'$, then s' is s adapted to the successful termination of the ith process of the processes being interleaved.

For each $n \in \mathbb{N}_1$, we instantiate the abstract scheduler σ_n and control

[9] The special function notation used in this paper is explained in an appendix.

state transformer ϑ_n as follows:

$$\sigma_n(h,s) = \mathit{sched}_n(h,s) \;,$$

$$\begin{array}{ll}
\vartheta_n(\langle\,\rangle, s, i, a, 0) = [\,] & \text{if } a \notin C \;, \\
\vartheta_n(h \frown (j,n), s, i, a, 0) = s & \text{if } a \notin C \;, \\
\vartheta_n(\langle\,\rangle, s, i, \mathsf{wait}(r), 0) = [r \mapsto \langle\,\rangle] \;, & \\
\vartheta_n(h \frown (j,n), s, i, \mathsf{wait}(r), 0) = s \dagger [r \mapsto \langle\,\rangle] & \text{if } r \notin \mathrm{dom}(s) \;, \\
\vartheta_n(h \frown (j,n), s, i, \mathsf{wait}(r), 0) = s \dagger [r \mapsto s(r) \frown i] & \text{if } r \in \mathrm{dom}(s) \;, \\
\vartheta_n(\langle\,\rangle, s, i, \mathsf{signal}(r), 0) = [\,] \;, & \\
\vartheta_n(h \frown (j,n), s, i, \mathsf{signal}(r), 0) = s & \text{if } r \notin \mathrm{dom}(s) \;, \\
\vartheta_n(h \frown (j,n), s, i, \mathsf{signal}(r), 0) = s \mathbin{⩤} \{r\} & \text{if } r \in \mathrm{dom}(s) \land s(r) = \langle\,\rangle \;, \\
\vartheta_n(h \frown (j,n), s, i, \mathsf{signal}(r), 0) = s \dagger [r \mapsto \mathrm{tl}(s(r))] & \text{if } r \in \mathrm{dom}(s) \land s(r) \neq \langle\,\rangle \;, \\
\vartheta_n(h, s, i, a, 1) = \mathit{remove}_n(s, i) \;. &
\end{array}$$

The following clarifies the connection between the instantiated control state transformers ϑ_n and the semaphore mechanism as introduced in [21]:

- $s = [\,]$ indicates that all semaphores have value 1;
- if $r \notin \mathrm{dom}(s)$, then the transition from s to $s \dagger [r \mapsto \langle\,\rangle]$ indicates that the value of semaphore r changes from 1 to 0;
- if $r \in \mathrm{dom}(s)$, then the transition from s to $s \dagger [r \mapsto s(r) \frown i]$ indicates that the ith process being interleaved is added to the queue of processes suspended on semaphore r;
- if $r \notin \mathrm{dom}(s)$, then the transition from s to s indicates that the value of semaphore r remains 1;
- if $r \in \mathrm{dom}(s)$ and $s(r) = \langle\,\rangle$, then the transition from s to $s \mathbin{⩤} \{r\}$ indicates that the value of semaphore r changes from 0 to 1;
- if $r \in \mathrm{dom}(s)$ and $s(r) \neq \langle\,\rangle$, then the transition from s to $s \dagger [r \mapsto \mathrm{tl}(s(r))]$ indicates that the first process in the queue of processes suspended on semaphore r is removed from that queue.

The example given above is only meant to show that the generic probabilistic interleaving strategy assumed in pACP+pSI can be instantiated with non-trivial specific probabilistic interleaving strategies. In practice, more advanced probabilistic interleaving strategies, such as strategies based on lottery scheduling [34], are more important.

5 Concluding Remarks

We have presented a probabilistic version of ACP [9, 14] that rests on the principle that probabilistic choices are always resolved before choices involved in alternative composition and parallel composition are resolved. By taking functions whose range is the carrier of a signed cancellation meadow [12, 19] instead of a field as probability measures, we could include probabilistic choice operators for the probabilities 0 and 1 without any problem and give a simple operational semantics.

We have also extended this probabilistic version of ACP with a form of interleaving in which parallel processes are interleaved according to what is known as a process-scheduling policy in the field of operating systems. This is the form of interleaving that underlies multi-threading as found in contemporary programming languages. To our knowledge, the work presented in [16] and this paper is the only work on this form of interleaving in the setting of a general algebraic theory of processes like ACP, CCS and CSP.

The main probabilistic versions of ACP introduced earlier are prACP [6], pACP^{+} [2], and pACP_{τ} [4]. Like pACP, those probabilistic versions of ACP are based on the generative model of probabilistic processes. In prACP, the alternative composition operator and the parallel composition operator are replaced by probabilistic choice operators and probabilistic parallel composition operators. In pACP^{+}, no operators are replaced, but probabilistic choice operators are added. The parallel composition operator of pACP^{+} is somewhat tricky because probabilistic choices are not resolved before choices involved in parallel composition are resolved. pACP_{τ} is, apart from abstraction, pACP^{+} with another parallel composition operator where probabilistic choices are resolved before choices involved in parallel composition are resolved. pACP is a minor variant of pACP_{τ} without abstraction operators. The differences and their consequences are described in the first and last but one paragraph of Section 3.5.

In this paper, we consider strategic interleaving where process creation is taken into account. The approach to process creation followed originates from the one first followed in [11] to extend ACP with process creation and later followed in [5, 7, 17] to extend different timed versions of ACP with process creation. The only other approach that we know of is the approach, based on [1], that has for instance been followed in [8, 22]. However, with that approach, it is most unlikely that data about the creation of processes can be made available for the decision making concerning the strategic interleaving of processes.

Appendix: Sequence and Function Notations

We use the following sequence notation:

- $\langle\,\rangle$ for the empty sequence;
- d for the sequence having d as sole element;
- $u \frown v$ for the concatenation of sequences u and v;
- $\mathrm{hd}(u)$ for the first element of non-empty sequence u;
- $\mathrm{tl}(u)$ for the subsequence of non-empty sequence u whose first element is the second element of u and whose last element is the last element of u;
- $\mathrm{elems}(u)$ is the set of all elements of sequence u.

We use the following special function notation:

- $[\,]$ for the empty function;
- $[d \mapsto e]$ for the function f with $\mathrm{dom}(f) = \{d\}$ such that $f(d) = e$;
- $f \dagger g$ for the function h with $\mathrm{dom}(h) = \mathrm{dom}(f) \cup \mathrm{dom}(g)$ such that for all $d \in \mathrm{dom}(h)$, $h(d) = f(d)$ if $d \notin \mathrm{dom}(g)$ and $h(d) = g(d)$ otherwise;
- $f \unlhd S$ for the function g with $\mathrm{dom}(g) = \mathrm{dom}(f) \setminus S$ such that for all $d \in \mathrm{dom}(g)$, $g(d) = f(d)$.

References

[1] P. America, J.W. de Bakker. Designing Equivalent Semantic Models for Process Creation. *Theoretical Computer Science* 60(2), 109–176, 1988. doi:10.1016/0304-3975(88)90048-5.

[2] S. Andova. Process Algebra with Probabilistic Choice. In J.-P. Katoen (Ed.), *Formal Methods for Real-Time and Probabilistic Systems (ARTS 1999)*, *Lecture Notes in Computer Science* 1601, 111–129, 1999. doi:10.1007/3-540-48778-6_7.

[3] S. Andova. *Probabilistic Process Algebra*. PhD thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, 2002. doi:10.6100/IR561343.

[4] S. Andova, S. Georgievska. On Compositionality, Efficiency, and Applicability of Abstraction in Probabilistic Systems. In M. Nielsen, A. Kučera, P.B. Miltersen, C. Palamidessi, P. Tůma, F. Valencia (Eds.), *Theory and Practice of Computer Science (SOFSEM 2009)*, *Lecture Notes in Computer Science* 5404, 67–78, 2009. doi:10.1007/978-3-540-95891-8_10.

[5] J.C.M. Baeten, J.A. Bergstra. Real Space Process Algebra. *Formal Aspects of Computing* 5(6), 481–529, 1993. doi:10.1007/BF01211247.

[6] J.C.M. Baeten, J.A. Bergstra, S.A. Smolka. Axiomatizing Probabilistic Processes: ACP with Generative Probabilities. *Information and Computation* 121(2), 234–255, 1995. doi:10.1006/inco.1995.1135.

[7] J.C.M. Baeten, C.A. Middelburg. *Process Algebra with Timing*. Monographs in Theoretical Computer Science, An EATCS Series. Springer-Verlag, Berlin, 2002. doi:10.1007/978-3-662-04995-2.

[8] J.C.M. Baeten, F.W. Vaandrager. An Algebra of Process Creation. *Acta Informatica* 29(4), 303–334, 1992. doi:10.1007/BF01178776.

[9] J.C.M. Baeten, W.P. Weijland. *Process Algebra*, *Cambridge Tracts in Theoretical Computer Science* 18. Cambridge University Press, Cambridge, 1990. doi:10.1017/CBO9780511624193.

[10] M. Ben-Ari. *Principles of Concurrent and Distributed Programming*. Pearson, Harlow, second edition, 2006.

[11] J.A. Bergstra. A Process Creation Mechanism in Process Algebra. In J.C.M. Baeten (Ed.), *Applications of Process Algebra*, *Cambridge Tracts in Theoretical Computer Science* 17, 81–88, 1990. doi:10.1017/CBO9780511608841.006.

[12] J.A. Bergstra, I. Bethke, A. Ponse. Cancellation Meadows: A Generic Basis Theorem and Some Applications. *Computer Journal* 56(1), 3–14, 2013. doi:10.1093/comjnl/bxs028.

[13] J.A. Bergstra, J.W. Klop. The Algebra of Recursively Defined Processes and the Algebra of Regular Processes. In J. Paredaens (Ed.), *Automata, Languages and Programming (ICALP 1984)*, *Lecture Notes in Computer Science* 172, 82–95, 1984. doi:10.1007/3-540-13345-3_7.

[14] J.A. Bergstra, J.W. Klop. Process Algebra for Synchronous Communication. *Information and Control* 60(1–3), 109–137, 1984. doi:10.1016/S0019-9958(84)80025-X.

[15] J.A. Bergstra, C.A. Middelburg. Thread Algebra for Strategic Interleaving. *Formal Aspects of Computing* 19(4), 445–474, 2007. doi:10.1007/s00165-007-0024-9.

[16] J.A. Bergstra, C.A. Middelburg. Process Algebra with Strategic Interleaving. *Theory of Computing Systems* 63(3), 488–505, 2019. doi:10.1007/s00224-018-9873-2.

[17] J.A. Bergstra, C.A. Middelburg, Y.S. Usenko. Discrete Time Process Algebra and the Semantics of SDL. In J.A. Bergstra, A.Ponse, S.A. Smolka (Eds.), *Handbook of Process Algebra*, 1209–1268, 2001. doi:10.1016/B978-044482830-9/50036-9.

[18] J.A. Bergstra, A. Ponse. Probability Functions in the Context of Signed Involutive Meadows. In P. James, M. Roggenbach (Eds.), *Recent Trends in Algebraic Development Techniques (WADT 2016)*, *Lecture Notes in Computer Science* 10644, 73–87, 2017. doi:10.1007/978-3-319-72044-9_6.

[19] J.A. Bergstra, J.V. Tucker. The Rational Numbers as an Abstract Data Type. *Journal of the ACM* 54(2), Article 7, 2007. doi:10.1145/1219092.1219095.

[20] P. Brinch Hansen. *Operating System Principles*. Prentice-Hall, Englewood Cliffs, NJ, 1973.

[21] E.W. Dijkstra. Cooperating Sequential Processes. In F. Genuys (Ed.), *Programming Languages*, 43–112, 1968.

[22] T. Gehrke, A. Rensink. Process Creation and Full Sequential Composition in a Name-Passing Calculus. *Electronic Notes in Theoretical Computer Science* 7, 141–160, 1997. doi:10.1016/S1571-0661(05)80471-2.

[23] S. Georgievska. *Probability and Hiding in Concurrent Processes*. PhD thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, Eindhoven, 2011. doi:10.6100/IR716397.

[24] J. Gosling, B. Joy, G. Steele, G. Bracha. *The Java Language Specification*. Addison-Wesley, Reading, MA, third edition, 2005.

[25] A. Hejlsberg, S. Wiltamuth, P. Golde. *C# Language Specification*. Addison-Wesley, Reading, MA, 2003.

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

[27] R. Lanotte, S. Tini. Probabilistic Bisimulation as a Congruence. *ACM Transactions on Computational Logic* 10(2), Article 9, 2009. doi:10.1145/1462179.1462181.

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

[29] A. Sabelfeld, D. Sands. Probabilistic Noninterference for Multi-Threaded Programs. In *Proceedings 13th IEEE Computer Security Foundations Workshop (CSFW-13)*, 2000. doi:10.1109/CSFW.2000.856937.

[30] A. Silberschatz, P.B. Galvin, G. Gagne. *Operating System Concepts*. John Wiley and Sons, Hoboken, NJ, tenth edition, 2018.

[31] A.S. Tanenbaum, H. Bos. *Modern Operating Systems*. Pearson, Harlow, fourth edition, 2015.

[32] R.J. van Glabbeek, S.A. Smolka, B. Steffen. Reactive, Generative and Stratified Models of Probabilistic Processes. *Information and Computation* 121(1), 59–80, 1995. doi:10.1109/LICS.1990.113740.

[33] R.J. van Glabbeek, F.W. Vaandrager. Modular Specification of Process Algebras. *Theoretical Computer Science* 113(2), 293–348, 1993. doi:10.1016/0304-3975(93)90006-F.

[34] C.A. Waldspurger, W.E. Weihl. Lottery Scheduling: Flexible Proportional-Share Resource Management. In *Proceedings of the 1st USENIX conference on Operating Systems Design and Implementation (OSDI '94)*, Article 1, 1994.