Remarks on Testing Probabilistic Processes

Yuxin Deng[B,1] Rob van Glabbeek[A,B] Matthew Hennessy[C,A,2]
Carroll Morgan[B,1] Chenyi Zhang[A,B]

[A] *National ICT Australia, Locked Bag 6016, Sydney, NSW 1466, Australia*
[B] *School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia*
[C] *Department of Informatics, University of Sussex, Falmer, Brighton, BN1 9QN, UK*

Abstract

We develop a general testing scenario for probabilistic processes, giving rise to two theories: *probabilistic may testing* and *probabilistic must testing*. These are applied to a simple probabilistic version of the process calculus CSP. We examine the algebraic theory of probabilistic testing, and show that many of the axioms of standard testing are no longer valid in our probabilistic setting; even for non-probabilistic CSP processes, the distinguishing power of probabilistic tests is much greater than that of standard tests. We develop a method for deriving inequations valid in probabilistic may testing based on a probabilistic extension of the notion of *simulation*. Using this, we obtain a complete axiomatisation for non-probabilistic processes subject to probabilistic may testing.

Keywords: Probabilistic processes, nondeterminism, CSP, transition systems, testing equivalences, simulation, complete axiomatisations, structural operational semantics

1 Introduction

Operational semantics has played a useful role in computer science since the very inception of the subject [Lan63,ER64,Luc71,BJ78]. But with the publication of [Plo81] (republished as [Plo04]) came the realisation that, properly structured, operational semantics provides an elegant compositional method for specifying the semantics of programming languages. Because of its mathematical foundations, structural operational semantics can also be used to reason about the behavioural properties of programs, to the extent it even brings into question the necessity of denotational semantics.

Nowhere is the power of operational semantics more evident than in the development of process calculi: the semantic theory underlying CCS [Mil89], bisimulation

[1] We acknowledge the support of the Australian Research Council (ARC) Grant DP034557.
[2] This research was carried out while on a Royal Society/Leverhulme Trust Senior Research Fellowship.

equivalence, is founded entirely on operational semantics. It provides powerful co-inductive proof methods for establishing process equivalences; it also supports compositional and algebraic reasoning techniques.

With CSP [Hoa85] the story is somewhat different: the *failures model* [BHR84] is denotational and was used to justify the algebraic laws so characteristic of the subsequent development of CSP as a specification language for processes. However, it later became apparent that, just as for CCS, this model and its algebraic theory could be fully justified using purely operational concepts based on notions of process testing [OH86,DH84]. So with CSP we have the best of all possible worlds:

- a denotational model,
- an operational model and
- an algebraic theory,

all of which are in some sense equivalent.

The topic of this paper is probabilistic process calculi. The various semantic approaches for standard process calculi are essentially theories of *nondeterministic* processes, where the nondeterminism represents possible choices that can be resolved in a wholly unpredictable way. With probabilistic constructs the resolution becomes predictable up to a point, in that it is quantified; but the interaction between the quantified and unquantified forms of choice then requires close attention. The issue is mainly that the two forms of choice differ, not that one is quantified and the other is not: for example, a similar effect occurs when considering demonic and angelic choice together. Because they differ it is necessary to consider carefully the order in which they occur, and how they might or might not distribute over each other.

The first papers on probabilistic process calculi [GJS90,Chr90,LS92] proceed by *replacing* nondeterministic with probabilistic choice. The reconciliation of nondeterministic and probabilistic choice starts with [HJ90] and has received a lot of attention in the literature [JL91,WL92,JHW94,SL94,Low93,Low95,Seg95,MMSS96, PLS00,BS01,JW02,SV03], even in the sequential world [HSM97,MMS96], and, more recently, within general semantic domains [MM01,Mis00,MOW04,TKP05,VW06]; as such it could be argued that it is one of the central problems of the area. And CSP makes the issue more interesting still, by having two forms of choice already (both internal and external), so that probabilistic choice becomes the third.

The emphasis in this paper is on the development of an algebraic theory. Following [WL92] we adapt the original idea of testing [DH84] to probabilistic processes, arriving at two refinement relations between processes, the *probabilistic may preorder* and the *probabilistic must preorder*. For example $P \sqsubseteq_{\mathrm{pmay}} Q$ means that the probability that Q might pass a test is at least as good as the probability that P might pass it. We then apply this general framework of probabilistic testing to a simple finite probabilistic process algebra, pCSP, obtained by adding a probabilistic choice operator to a cut-down version of CSP. In order to do so we first need to interpret pCSP as a *probabilistic labelled transition system* [JWL01,HJ90,Seg95], a generalisation of the well known concept of *labelled transition system*, which describes the interactions which processes may have with their users.

With these generalisations it turns out that very few of the attractive algebraic laws underlying the algebraic theory of CSP, and indeed its denotational model, remain valid in the presence of probabilistic choice; this is demonstrated by a series of counterexamples, consisting of tests which distinguish process expressions previously identified. The main result of the paper is the development of a method for demonstrating positive algebraic identities. We develop a notion of *simulation* for probabilistic processes, writing $P \sqsubseteq_S Q$ to mean that process Q can simulate the behaviour or P. We then go on to show

- the simulation relation $\sqsubseteq_S$ is preserved by all the operators in pCSP
- the simulation relation is included in the probabilistic may preorder: $P \sqsubseteq_S Q$ implies $P \sqsubseteq_{\text{pmay}} Q$.

This enables us to develop an algebraic theory for probabilistic may testing, which we briefly outline. The concept of simulation can also be adapted, by introducing a notion of *failure*, to obtain similar results for the probabilistic must preorder; but in order to keep the paper concise, the details are omitted. We do however show that, as expected, the theory based on *must* testing is more discriminating than that based on *may* testing; specifically $P \sqsubseteq_{\text{pmust}} Q$ implies $Q \sqsubseteq_{\text{pmay}} P$.

In the final section we re-examine the behaviour of standard (non-probabilistic) CSP, using probabilistic tests. We show that these are in general more discriminating than non-probabilistic tests, and we give a complete equational characterisation for the resulting *may* theory.

Although this paper develops an algebraic theory of probabilistic testing in terms of pCSP, we could have obtained similar results using probabilistic versions of CCS or ACP, because processes defined in these calculi can be interpreted likewise.

2 Testing processes

It is natural to view the semantics of processes as being determined by their ability to pass tests [DH84,Hen88,BHR84,WL92,Seg96]; processes P_1 and P_2 are deemed to be semantically equivalent unless there is a test which can distinguish them. The actual tests used typically represent the ways in which users, or indeed other processes, can interact with P_i.

Let us first set up a general testing scenario, within which this idea can be formulated. It assumes

- a set of processes $\mathcal{P}roc$
- a set of tests $\mathcal{T}$, which can be applied to processes
- a set of outcomes $\mathcal{O}$, the possible results from applying a test to a process
- a function $\mathcal{A}pply : \mathcal{T} \times \mathcal{P}roc \rightarrow \mathcal{P}^+_{\mathit{fin}}(\mathcal{O})$, representing the possible results of applying a specific test to a specific process.

Here $\mathcal{P}^+_{\mathit{fin}}(\mathcal{O})$ denotes the collection of finite non-empty subsets of $\mathcal{O}$; so the result of applying a test T to a process P, $\mathcal{A}pply(T, P)$, is in general a *set* of outcomes, representing the fact that the behaviour of processes, and indeed tests, may be nondeterministic. A more general theory would allow $\mathcal{A}pply(T,P)$ to be an arbitrary non-empty subset of $\mathcal{O}$, but for the class of finite processes we consider in this paper, a finite set of outcomes turns out to be sufficient.

Moreover, some outcomes are considered better then others; for example the application of a test may simply succeed, or it may fail, with success being better than failure. So we can assume that $\mathcal{O}$ is endowed with a partial order, in which $o_1 \leq o_2$ means that o_2 is a better outcome than o_1.

When comparing the result of applying tests to processes we need to compare subsets of $\mathcal{O}$. There are two standard approaches to make this comparison, based on viewing these sets as elements of either the Hoare or Smyth powerdomain [Hen82,AJ94] of $\mathcal{O}$. For $O_1, O_2 \in \mathcal{P}^+_{\mathit{fin}}(\mathcal{O})$ we let

(i) $O_1 \sqsubseteq_{\mathrm{Ho}} O_2$ if for every $o_1 \in O_1$ there exists some $o_2 \in O_2$ such that $o_1 \leq o_2$

(ii) $O_1 \sqsubseteq_{\mathrm{Sm}} O_2$ if for every $o_2 \in O_2$ there exists some $o_1 \in O_1$ such that $o_1 \leq o_2$.

Using these two comparison methods we obtain two different semantic preorders for processes:

(i) For $P, Q \in \mathcal{P}roc$ let $P \sqsubseteq_{\mathrm{may}} Q$ if $\mathcal{A}pply(T,P) \sqsubseteq_{\mathrm{Ho}} \mathcal{A}pply(T,Q)$ for every test T

(ii) Similarly, let $P \sqsubseteq_{\mathrm{must}} Q$ if $\mathcal{A}pply(T,P) \sqsubseteq_{\mathrm{Sm}} \mathcal{A}pply(T,Q)$ for every test T.

We use $P \simeq_{\mathrm{may}} Q$ and $P \simeq_{\mathrm{must}} Q$ to denote the associated equivalences.

The terminology *may* and *must* refers to the following reformulation of the same idea. Let $\mathit{Pass} \subseteq \mathcal{O}$ be an upwards-closed subset of $\mathcal{O}$, i.e. satisfying $o' \geq o \in \mathit{Pass} \Rightarrow o' \in \mathit{Pass}$, thought of as the set of outcomes that can be regarded as *passing* a test. Then we say that a process P *may* pass a test T with an outcome in Pass, notation "P **may** Pass T", if there is an outcome $o \in \mathcal{A}pply(P,T)$ with $o \in \mathit{Pass}$, and likewise P *must* pass a test T with an outcome in Pass, notation "P **must** Pass T", if for all $o \in \mathcal{A}pply(P,T)$ one has $o \in \mathit{Pass}$. Now

$$P \sqsubseteq_{\mathrm{may}} Q \;\text{ iff }\; \forall T \in \mathcal{T}\, \forall \mathit{Pass} \in \mathcal{P}^{\uparrow}(\mathcal{O})\, (P \textbf{ may } \mathit{Pass}\ T \;\Rightarrow\; Q \textbf{ may } \mathit{Pass}\ T)$$
$$P \sqsubseteq_{\mathrm{must}} Q \;\text{ iff }\; \forall T \in \mathcal{T}\, \forall \mathit{Pass} \in \mathcal{P}^{\uparrow}(\mathcal{O})\, (P \textbf{ must } \mathit{Pass}\ T \;\Rightarrow\; Q \textbf{ must } \mathit{Pass}\ T)$$

where $\mathcal{P}^{\uparrow}(\mathcal{O})$ is the set of upwards-closed subsets of $\mathcal{O}$.

The original theory of testing [DH84,Hen88] is obtained by using as the set of outcomes $\mathcal{O}$ the two-point lattice

$$\begin{array}{c} \top \\ | \\ \bot \end{array}$$

with $\top$ representing the success of a test application, and $\bot$ failure.

However, for probabilistic processes we consider an application of a test to a process to succeed with a given probability. Thus we take as the set of outcomes the unit interval $[0,1]$, with the standard mathematical ordering; if $0 \leq p < q \leq 1$ then succeeding with probability q is considered better than succeeding with probability p. This yields two preorders for probabilistic processes, which for convenience we rename $\sqsubseteq_{\rm pmay}$ and $\sqsubseteq_{\rm pmust}$, with the associated equivalences $\simeq_{\rm pmay}$ and $\simeq_{\rm pmust}$ respectively. These preorders, and their associated equivalences, were first defined by Wang and Larsen [WL92]. The purpose of the current paper is to apply them to a simple probabilistic process algebra.

Before doing so let us first point out a useful simplification: the Hoare and Smyth preorders on finite subsets of $[0,1]$ (and more generally on *closed* subsets of $[0,1]$) are determined by their maximum and minimum elements respectively.

Proposition 2.1 For $O_1,\ O_2 \in \mathcal{P}^+_{\mathit{fin}}(\mathcal{O}_{\rm prob})$ we have

(i) $O_1 \sqsubseteq_{\rm Ho} O_2$ if and only if $\mathit{max}(O_1) \le \mathit{max}(O_2)$

(ii) $O_1 \sqsubseteq_{\rm Sm} O_2$ if and only if $\mathit{min}(O_1) \le \mathit{min}(O_2)$.

Proof. Straightforward calculations. $\square$

As in the non-probabilistic case [DH84], we could also define a testing preorder combining the may- must-preorders; we will not study this combination here.

3 Finite probabilistic CSP

We first define the language and its operational semantics. Then we show how the general probabilistic testing theory just outlined can be applied to processes from this language.

3.1 The language

Let Act be a set of actions, ranged over by $a, b, c, \ldots$, which processes can perform. Then the finite probabilistic CSP processes are given by the following syntax:

$$P ::= \mathbf{0} \ \mid\ a.P \ \mid\ P \sqcap P \ \mid\ P \,\Box\, P \ \mid\ P \,|_A\, P \ \mid\ P \,{}_p\!\oplus P$$

The intuitive meaning of the various constructs is straightforward:

(i) $\mathbf{0}$ represents the stopped process.

(ii) $a.P$, where a is in Act, is a process which first performs the action a, and then proceeds as P.

(iii) $P \sqcap Q$ is the *internal choice* between the processes P and Q; it will act either like P or like Q, but a user is unable to influence which.

(iv) $P \,\Box\, Q$ is the *external choice* between P and Q; again it will act either like P or like Q but, in this case, according to the demands of a user.

(v) $P \,|_A\, Q$, where A is a subset of Act, represents processes P and Q running in parallel. They may synchronise by performing the same action from A simultaneously; such a synchronisation results in an internal action $\tau \notin \mathsf{Act}$. In addition P and Q may independently do any action from $(\mathsf{Act} \setminus A) \cup \{\tau\}$.

(vi) $P \,{}_p\!\oplus Q$, where p is an arbitrary probability, a real number strictly between 0 and 1, is the *probabilistic choice* between P and Q. With probability p it will act like P and with probability $(1{-}p)$ it will act like Q.

We use pCSP to denote the set of terms defined by this grammar, and CSP denotes the subset of that which does not use the probabilistic choice. Of course the language CSP in all its glory [BHR84,Hoa85,OH86] has many more operators; we have simply chosen a representative selection, adding probabilistic choice to obtain an elementary probabilistic version of CSP. Our parallel operator is not a CSP primitive, but it can easily be expressed in terms of the CSP primitives—in particular $P \mid_A Q = (P \|_A Q) \backslash A$, where $\|_A$ and $\backslash A$ are the parallel composition and hiding operators of [OH86]. It can also be expressed in terms of the parallel composition, renaming and restriction operators of CCS. We have chosen this (non-associative) operator for its convenience in defining the application of tests to processes.

As usual we tend to omit occurrences of $\mathbf{0}$; the action prefixing operator $a.\_$ binds stronger than any of the binary operators, and precedence between the binary operators will be indicated via brackets or spacing. We will also sometimes use n-ary versions of the binary operators, such as $\bigoplus_{i\in I} p_i P_i$ with $\sum_{i\in I} p_i = 1$, and $\Box_{i\in I} P_i$.

3.2 Operational Semantics of pCSP

The above intuitive reading of the various operators can be formalised by an *operational semantics* which associates with each process term a graph-like structure representing the manner in which it may react to users' requests. Let us briefly recall this procedure for (non-probabilistic) CSP.

Definition 3.1 A *labelled transition system* (LTS) is a triple $\langle S, \mathsf{Act}_\tau, \rightarrow\rangle$, where

(i) S is a set of states

(ii) Act_τ is a set of actions Act, augmented with a new action symbol τ to represent synchronisations, and more generally internal unobservable activity

(iii) $\rightarrow \;\subseteq\; S \times \mathsf{Act}_\tau \times S$ represents the effect of performing actions.

It is usual to use the more intuitive notation $s \xrightarrow{\alpha} s'$ instead of $(s, \alpha, s') \in \rightarrow$.

The operational semantics of CSP is obtained by endowing the set of terms with the structure of an LTS. Specifically

(i) the set of states S is taken to be all terms from the language CSP

(ii) the action relations $P \xrightarrow{\alpha} Q$ are defined inductively on the syntax of terms.

A precise definition may be found in [OH86].

In order to interpret the full pCSP operationally we need to generalise the notion of LTS to take probabilities into account. First we need some notation for *probability distributions*. A (discrete) probability distribution over a set S is a mapping $\Delta : S \rightarrow [0,1]$ with $\sum_{s\in S} \Delta(s) = 1$. The *support* of Δ is given by $\lceil \Delta \rceil := \{\, s \in S \mid \Delta(s) > 0 \,\}$. For our simple setting we only require distributions with finite support; let $\mathcal{D}(S)$, ranged over by Δ, Θ, Φ, denote the collection of all such distributions over S. We use $\overline{s}$ to denote the point distribution, satisfying

$$\overline{s}(t) = \begin{cases} 1 & \text{if } t = s, \\ 0 & \text{otherwise} \end{cases}$$

while if $p_i \geq 0$ and Δ_i is a distribution for each i in some finite index set I, then $\sum_{i \in I} p_i \cdot \Delta_i$ is given by

$$(\sum_{i \in I} p_i \cdot \Delta_i)(s) \quad = \quad \sum_{i \in I} p_i \cdot \Delta_i(s)$$

If $\sum_{i \in I} p_i = 1$ then this is easily seen to be a distribution in $\mathcal{D}(S)$, and we will sometimes write it as $p_1 \cdot \Delta_1 + \ldots + p_n \cdot \Delta_n$ when the index set I is $\{1, \ldots, n\}$.

We can now present the probabilistic generalisation of an LTS:

Definition 3.2 A *probabilistic labelled transition system* (pLTS) is a triple $\langle S, \mathsf{Act}_\tau, \rightarrow\rangle$, where

(i) S is a set of states

(ii) Act_τ is a set of actions Act, augmented by a new action τ, as above

(iii) $\rightarrow \; \subseteq \; S \times \mathsf{Act}_\tau \times \mathcal{D}(S)$.

As with LTSs, we usually write $s \xrightarrow{\alpha} \Delta$ in place of $(s, \alpha, \Delta) \in \rightarrow$. An LTS may be viewed as a degenerate pLTS, one in which only point distributions are used.

We now mimic the operational interpretation of CSP as an LTS by associating with pCSP a particular pLTS $\langle S_p, \mathsf{Act}_\tau, \rightarrow\rangle$. However there are two major differences:

(i) only a subset of terms in pCSP will be used as the set of states S_p in the pLTS

(ii) terms in pCSP will be interpreted as distributions over S_p, rather than as elements of S_p.

First we define the subset S_p of states that we use: it is the least set satisfying

(i) $\mathbf{0} \in S_p$

(ii) $a.P \in S_p$

(iii) $P \sqcap Q \in S_p$

(iv) $s_1, s_2 \in S_p$ implies $s_1 \,\Box\, s_2 \in S_p$

(v) $s_1, s_2 \in S_p$ implies $s_1 \mid_A s_2 \in S_p$.

Thus, S_p is the set of pCSP expressions in which every occurrence of the probabilistic choice operator ${}_p\!\oplus$ is *weakly guarded*, i.e. is within a subexpression of the form $a.P$ or $P \sqcap Q$. The interpretation of terms in pCSP as distributions in $\mathcal{D}(S_p)$ is as follows:

(i) $[\![\, \mathbf{0} \,]\!] = \overline{\mathbf{0}}$

(ii) $[\![a.P]\!] = \overline{a.P}$

(iii) $[\![P \sqcap Q]\!] = \overline{P \sqcap Q}$

(iv) $[\![P \,{}_p\!\oplus Q]\!] = p \cdot [\![P]\!] + (1{-}p) \cdot [\![Q]\!]$

(v) $[\![P \,\Box\, Q]\!] = [\![P]\!] \,\Box\, [\![Q]\!]$

(vi) $[\![P \mid_A Q]\!] = [\![P]\!] \mid_A [\![Q]\!]$.

In the last two cases we take advantage of the fact that both $\Box$ and $\mid_A$ can be viewed

(ACTION)
$$a.P \xrightarrow{a} [\![P]\!]$$

(INT.L)
$$P \sqcap Q \xrightarrow{\tau} [\![P]\!]$$

(INT.R)
$$P \sqcap Q \xrightarrow{\tau} [\![Q]\!]$$

(EXT.I.L)
$$\frac{s_1 \xrightarrow{\tau} \Delta}{s_1 \,\square\, s_2 \xrightarrow{\tau} \Delta \,\square\, \overline{s_2}}$$

(EXT.I.R)
$$\frac{s_2 \xrightarrow{\tau} \Delta}{s_1 \,\square\, s_2 \xrightarrow{\tau} \overline{s_1} \,\square\, \Delta}$$

(EXT.L)
$$\frac{s_1 \xrightarrow{a} \Delta}{s_1 \,\square\, s_2 \xrightarrow{a} \Delta}$$

(EXT.R)
$$\frac{s_2 \xrightarrow{a} \Delta}{s_1 \,\square\, s_2 \xrightarrow{a} \Delta}$$

(PAR.L)
$$\frac{s_1 \xrightarrow{\alpha} \Delta}{s_1 \mid_A s_2 \xrightarrow{\alpha} \Delta \mid_A \overline{s_2}} \quad \alpha \notin A$$

(PAR.R)
$$\frac{s_2 \xrightarrow{\alpha} \Delta}{s_1 \mid_A s_2 \xrightarrow{\alpha} \overline{s_1} \mid_A \Delta} \quad \alpha \notin A$$

(PAR.I)
$$\frac{s_1 \xrightarrow{a} \Delta_1, \; s_2 \xrightarrow{a} \Delta_2}{s_1 \mid_A s_2 \xrightarrow{\tau} \Delta_1 \mid_A \Delta_2} \quad a \in A$$

Fig. 1. Operational semantics of pCSP. Here a ranges over Act and α over Act_τ.

as binary operators over S_p, and can therefore be lifted to $\mathcal{D}(S_p)$ in the standard manner. Namely we define

$$(\Delta_1 \,\square\, \Delta_2)(s) \quad = \quad \begin{cases} \Delta_1(t_1) \cdot \Delta_2(t_2) & \text{if } s = t_1 \,\square\, t_2, \\ 0 & \text{otherwise} \end{cases}$$

with $\Delta_1 \mid_A \Delta_2$ given similarly; note that as a result we have $[\![P]\!] = \overline{P}$ for all $P \in S_p$. Finally the definition of the relations $\xrightarrow{\alpha}$ is given in Figure 1. These rules are very similar to the standard ones used to interpret CSP as an LTS [OH86], modified to take into account that the result of an action must be a distribution. For example (INT.L) and (INT.R) say that $P \sqcap Q$ makes an internal unobservable choice to act either like P or like Q. Similarly the four rules (EXT.L), (EXT.R), (EXT.I.L) and (EXT.I.R) can be read as giving the standard interpretation to the external choice operator: the process $P \,\square\, Q$ may perform any external action of its components P and Q, which resolves the choice; it may also perform any of their internal actions, but when these are performed the choice is not resolved.

3.3 The precedence of probabilistic choice

Our operational semantics entails that $\square$ and $\mid_A$ distribute over probabilistic choice:

$$[\![P \,\square\, (Q \,{}_p\!\oplus\, R)]\!] = [\![(P \,\square\, Q) \,{}_p\!\oplus\, (P \,\square\, R)]\!]$$
$$[\![P \mid_A (Q \,{}_p\!\oplus\, R)]\!] = [\![(P \mid_A Q) \,{}_p\!\oplus\, (P \mid_A R)]\!]$$

In Section 6.5, these identities will return as axioms of may testing. However, this is not so much a consequence of our testing methodology: it is hardwired in our interpretation $[\![\_]\!]$ of pCSP expressions as distributions. We could have obtained the same effect by introducing pCSP as a two-sorted language, given by the grammar

$$P ::= S \mid P\,{}_p\!\oplus P$$
$$S ::= \mathbf{0} \mid a.P \mid P \sqcap P \mid S \,\Box\, S \mid S \mid_A S$$

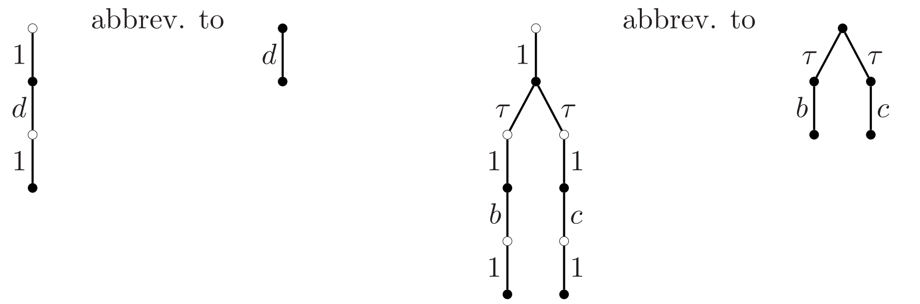
and introducing expressions like $P \,\Box\, (Q\,{}_p\!\oplus R)$ and $P \mid_A (Q\,{}_p\!\oplus R)$ as syntactic sugar for the grammatically correct expressions obtained by distributing $\Box$ and $\mid_A$ over ${}_p\!\oplus$. In that case, the S-expressions would constitute the set S_p of states in the pLTS of pCSP, and $[\![s]\!] = \overline{s}$ for any $s \in S_p$.

A consequence of our operational semantics is that in the process $a.(b\,{}_{\frac{1}{2}}\!\oplus c) \mid_\emptyset d$ the action d can be scheduled either before a, or after the probabilistic choice between b and c—but it can *not* be scheduled after a and before this probabilistic choice. We justify this by thinking of $P\,{}_p\!\oplus Q$ not as a process that starts with making a probabilistic choice, but rather as one that *has* just made such a choice, and with probability p *is* no more and no less than the process P. Thus $a.(P\,{}_p\!\oplus Q)$ is a process that in doing the a-step makes a probabilistic choice between the alternative targets P and Q.

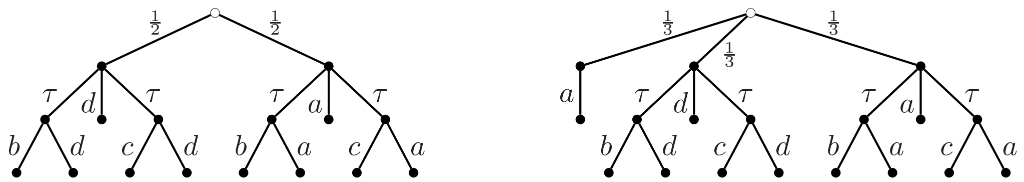
This design decision is in full agreement with previous work featuring nondeterminism, probabilistic choice and parallel composition [HJ90,WL92,Seg95]. Moreover, a probabilistic choice between processes P and Q that does not take precedence over actions scheduled in parallel can simply be written as $\tau.(P\,{}_p\!\oplus Q)$. Here $\tau.P$ is an abbreviation for $P \sqcap P$. Using the operational semantics of $\sqcap$ in Figure 1, $\tau.P$ is a process whose sole initial transition is $\tau.P \xrightarrow{\tau} P$, hence $\tau.(P\,{}_p\!\oplus Q)$ is a process that starts with making a probabilistic choice, modelled as an internal action, and with probability p proceeds as P. Any activity scheduled in parallel with $\tau.(P\,{}_p\!\oplus Q)$ can now be scheduled before or after this internal action, hence before or after the making of the choice. In particular, $a.\tau.(b\,{}_{\frac{1}{2}}\!\oplus c) \mid_\emptyset d$ allows d to happen between a and the probabilistic choice.

3.4 Graphical representation of pCSP *processes*

We graphically depict the operational semantics of a pCSP expression P by drawing the part of the pLTS defined above that is reachable from $[\![P]\!]$ as a finite acyclic directed graph. Given that in a pLTS transitions go from states to distributions, we need to introduce additional edges to connect distributions back to states, thereby obtaining a bipartite graph. States are represented by nodes of the form $\bullet$ and distributions by nodes of the form $\circ$. For any state s and distribution Δ with $s \xrightarrow{\alpha} \Delta$ we draw an edge from s to Δ, labelled with α. Consequently, the edges leaving a $\bullet$-node are all labelled with actions from Act_τ. For any distribution Δ and state s in $\lceil \Delta \rceil$, the support of Δ, we draw an edge from Δ to s, labelled with $\Delta(s)$. Consequently, the edges leaving a $\circ$-node are labelled with positive real numbers that sum up to 1. Because for our simple language the resulting directed bipartite graphs are acyclic, we leave out arrowheads on edges and we assume control to flow



i) $d.\mathbf{0}$ *ii)* $b.\mathbf{0} \sqcap c.\mathbf{0}$



iii) $(b \sqcap c) \,\Box\, (d \,{}_{\frac{1}{2}}\!\oplus a)$ *iv)* $a \,{}_{\frac{1}{3}}\!\oplus ((b \sqcap c) \,\Box\, (d \,{}_{\frac{1}{2}}\!\oplus a))$

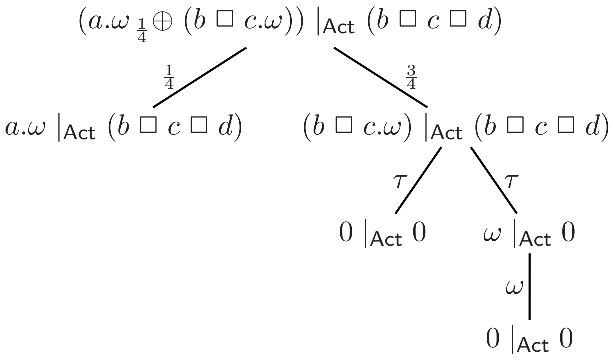
Fig. 2. Example pLTSs

from top to bottom.

A few examples are described in Figure 2. In *i)* we find $[\![d.\mathbf{0}]\!]$ as the point distribution $\overline{d.\mathbf{0}}$, represented by a tree with one edge from the root, labelled with the probability 1, to the state $d.\mathbf{0}$. In turn this state is represented as the subtree with one outgoing edge, labelled by the only possible action d to $[\![\mathbf{0}]\!]$. Finally this is also a point distribution, represented as a subtree with one edge leading to a leaf, labelled by the probability 1.

These edges labelled by probability 1 occur so frequently that it makes sense to omit them, together with the associated nodes $\circ$ representing point distributions. With this convention $[\![d.\mathbf{0}]\!]$ turns out to be a simple tree with one edge labelled by the action d. The same convention simplifies considerably the representation of $b \sqcap c$ in *ii)*, resulting in an LTS detailing an internal choice between the two actions. Officially, the endnodes of this graph ought to be merged, as both of them represent the process $\mathbf{0}$. However, for reasons of clarity, in graphical representations we often unwind the underlying graph into a tree, thus representing the same state or distribution multiple times.

The interpretation of $(b \sqcap c) \,\Box\, (d \,{}_{\frac{1}{2}}\!\oplus a)$ is more interesting. This requires clause (v) above in the definition of $[\![\_]\!]$, resulting in the distribution $\overline{(b \sqcap c)} \,\Box\, \Delta$, where Δ is the distribution resulting from the interpretation of $(d \,{}_{\frac{1}{2}}\!\oplus a)$, itself a two-point distribution mapping both the states $d.\mathbf{0}$ and $a.\mathbf{0}$ to the probability $\frac{1}{2}$. The result is again a two-point distribution, this time mapping the two states $(b \sqcap c) \,\Box\, d$ and $(b \sqcap c) \,\Box\, a$ to $\frac{1}{2}$. The end result in *iii)* is obtained by further interpreting these two states using the rules in Figure 1. Finally in *iv)* we show the graphical



$$\mathcal{A}pply((a.\omega \,{}_{\frac{1}{4}}\!\oplus (b \mathrel{\Box} c.\omega)), (b \mathrel{\Box} c \mathrel{\Box} d)) = \tfrac{1}{4} \cdot \{0\} + \tfrac{3}{4} \cdot \{0, 1\} = \{0, \tfrac{3}{4}\}$$

Fig. 3. Example of testing

representation which results when this term is combined probabilistically with the simple process $a.\mathbf{0}$.

To sum up, the operational semantics endows pCSP with the structure of a pLTS, and the function $[\![\_]\!]$ interprets process terms in pCSP as distributions in this pLTS, which can be represented by finite acyclic directed graphs (typically drawn as trees), with edges labelled either by probabilities or actions, so that edges labelled by actions and probabilities alternate (although in pictures we may suppress 1-labelled edges and point distributions).

3.5 Testing pCSP *processes*

Let us now turn to applying the testing theory from Section 2 to pCSP. As with the standard theory [DH84,Hen88], we use as tests any process from pCSP itself, which in addition can use a special symbol ω to report success. For example, $a.\omega \,{}_{\frac{1}{4}}\!\oplus (b \mathrel{\Box} c.\omega)$ is a probabilistic test, which 25% of the time requests an a action, and 75% requests that c can be performed. If it is used as *must* test the 75% that requests a c action additionally requires that b is not possible. As in [DH84,Hen88], it is not the execution of ω that constitutes success, but the arrival in a state where ω is possible. The introduction of the ω-action is simply a method of defining a success predicate on states without having to enrich the language of processes explicitly with such predicates.

Formally, let $\omega \notin \mathsf{Act}_\tau$ and write Act^ω for $\mathsf{Act} \cup \{\omega\}$ and Act^ω_τ for $\mathsf{Act} \cup \{\tau, \omega\}$. In Figure 1 we now let a range over Act^ω and α over Act^ω_τ. Tests may have subterms $\omega.P$, but other processes may not. To apply the test T to the process P we run them in parallel, tightly synchronised; that is, we run the combined process $T \mathrel{|_{\mathsf{Act}}} P$. Here P can only synchronise with T, and in turn the test T can only perform the success action ω, in addition to synchronising with the process being tested; of course both tester and testee can also perform internal actions. An example is provided in Figure 3, where the test $a.\omega \,{}_{\frac{1}{4}}\!\oplus (b \mathrel{\Box} c.\omega)$ is applied to the process $b \mathrel{\Box} c \mathrel{\Box} d$. We see that 25% of the time the test is unsuccessful, in that it does not reach a state where ω is possible, and 75% of the time it *may* be successful, depending on how the now internal choice between the actions b and c is resolved, but it is not the case that it *must* be successful.

$[\![T \mid_{\mathsf{Act}} P]\!]$ is representable as a finite graph which encodes all possible interactions of the test T with the process P. It only contains the actions τ and ω. Each occurrence of τ represents a nondeterministic choice, either in T or P themselves, or as a nondeterministic response by P to a request from T, while the distributions represent the resolution of underlying probabilities in T and P. We use the structure $[\![T \mid_{\mathsf{Act}} P]\!]$ to define $\mathcal{A}pply(T, P)$, the non-empty finite subset of $[0, 1]$ representing the set of probabilities that applying T to P will be a success.

First we define a function $\mathcal{R}esults(\_)$, which when applied to any state in S_p returns a finite subset of $[0, 1]$. The definition will require that it be also applied to distributions, and to do so we need to use *choice functions* for collecting elements from subsets of $[0, 1]$. Suppose $R : S_p \to \mathcal{P}^+_{\mathit{fin}}([0, 1])$, and $c : X \to [0, 1]$, where X is a subset of S_p. Then we write $c \in_X R$ if $c(s) \in R(s)$ for every s in X, and the results-collecting function can be defined as follows:

$$\mathcal{R}esults(s) \quad = \quad \begin{cases} \{1\} & \text{if } s \xrightarrow{\omega}, \\ \bigcup\{\, \mathcal{R}esults(\Delta) \mid s \xrightarrow{\tau} \Delta \,\} & \text{if } s \not\xrightarrow{\omega},\ s \xrightarrow{\tau}, \\ \{0\} & \text{otherwise} \end{cases}$$

where

$$\mathcal{R}esults(\Delta) \quad = \quad \{ \sum_{s \in \lceil \Delta \rceil} \Delta(s) \cdot c(s) \mid c \in_{\lceil \Delta \rceil} \mathcal{R}esults \}$$

However, instead of the explicit use of choice functions, we will tend to use the more convenient notation

$$\mathcal{R}esults(\Delta) \quad = \quad \Delta(s_1) \cdot \mathcal{R}esults(s_1) + \ldots + \Delta(s_n) \cdot \mathcal{R}esults(s_n)$$

where $\lceil \Delta \rceil = \{s_1, \ldots s_n\}$. Note that $\mathcal{R}esults(\_)$ is indeed a well-defined function, because the pLTS $\langle S_p, \mathsf{Act}_\tau, \rightarrow \rangle$ is finitely branching and well-founded.

For example consider the transition systems in Figure 4, where for reference we have labelled the nodes. Then $\mathcal{R}esults(s_1) = \{1, 0\}$ while $\mathcal{R}esults(s_2) = \{1\}$, and therefore $\mathcal{R}esults(\Delta_s) = \frac{1}{2} \cdot \{1, 0\} + \frac{1}{2} \cdot \{1\}$ which, since there are only two possible choice functions $c \in_{\lceil \Delta_s \rceil} \mathcal{R}esults$, evaluates further to $\{\frac{1}{2}, 1\}$. Similarly $\mathcal{R}esults(t_1) = \mathcal{R}esults(t_2) = \{0, 1\}$ and using the four choice functions $c \in_{\lceil \Delta_t \rceil} \mathcal{R}esults$, the calculation of $\mathcal{R}esults(\Delta_t) = \frac{1}{4} \cdot \{0, 1\} + \frac{3}{4} \cdot \{0, 1\}$ leads to $\{0, \frac{1}{4}, \frac{3}{4}, 1\}$.

Definition 3.3 For any $P \in \mathsf{pCSP}$ and $T \in \mathcal{T}$ let $\mathcal{A}pply(T, P) = \mathcal{R}esults([\![T|_{\mathsf{Act}}P]\!])$.

With this definition we now have two testing preorders for pCSP, one based on *may* testing, $P \sqsubseteq_{\mathrm{pmay}} Q$, and the other on *must* testing, $P \sqsubseteq_{\mathrm{pmust}} Q$.

$\mathcal{R}esults(\Delta_s) = \{\frac{1}{2},\ 1\}$ $\mathcal{R}esults(\Delta_t) = \{0,\ \frac{1}{4},\ \frac{3}{4},\ 1\}$

Fig. 4. Collecting results

4 Counterexamples

We will see in this section that many of the standard testing axioms are not valid in the presence of probabilistic choice. We also provide counterexamples for a few distributive laws involving probabilistic choice that may appear plausible at first sight. In all cases we establish a statement $P \not\simeq_{\text{pmay}} Q$ by exhibiting a test T such that $max(\mathcal{A}pply(T,P)) \neq max(\mathcal{A}pply(T,Q))$ and a statement $P \not\simeq_{\text{pmust}} Q$ by exhibiting a test T such that $min(\mathcal{A}pply(T,P)) \neq min(\mathcal{A}pply(T,Q))$. In case $max(\mathcal{A}pply(T,P)) > max(\mathcal{A}pply(T,Q))$ we find in particular that $P \not\sqsubseteq_{\text{pmay}} Q$, and in case $min(\mathcal{A}pply(T,P)) > min(\mathcal{A}pply(T,Q))$ we obtain $P \not\sqsubseteq_{\text{pmust}} Q$.

Example 4.1 The axiom $a.(P\,{}_p\!\oplus Q) = a.P\,{}_p\!\oplus a.Q$ is unsound.

Consider the example in Figure 5. In R_1 the probabilistic choice between b and c is taken after the action a, while in R_2 the choice is made before the action has happened. These processes can be distinguished by the nondeterministic test $T = a.b.\omega \sqcap a.c.\omega$. First consider running this test on R_1. There is an immediate choice made by the test, effectively running either the test $a.b.\omega$ on R_1 or the test $a.c.\omega$; in fact the effect of running either test is exactly the same. Consider $a.b.\omega$. When run on R_1 the a action immediately happens, and there is a probabilistic choice between running $b.\omega$ on either b or c, giving as possible results $\{1\}$ or $\{0\}$; combining these according to the definition of the function $\mathcal{R}esults(\_)$ we get $\frac{1}{2}\cdot\{0\}+\frac{1}{2}\cdot\{1\} = \{\frac{1}{2}\}$. Since running the test $a.c.\omega$ has the same effect, $\mathcal{A}pply(T,R_1)$ turns out to be the same set $\{\frac{1}{2}\}$.

Now consider running the test T on R_2. Because R_2, and hence also $T\ |_{\mathsf{Act}}\ R_2$, starts with a probabilistic choice, due to the definition of the function $\mathcal{R}esults(\_)$, the test must first be applied to the probabilistic components, $a.b$ and $a.c$, respectively, and the results subsequently combined probabilistically. When the test is run on $a.b$, immediately a nondeterministic choice is made in the test, to run either $a.b.\omega$ or $a.c.\omega$. With the former we get the result $\{1\}$, with the latter $\{0\}$, so overall, for running T on $a.b$, we get the possible results $\{0,1\}$. The same is true when we run it on $a.c$, and therefore $\mathcal{A}pply(T,R_2) = \frac{1}{2}\cdot\{0,1\}+\frac{1}{2}\cdot\{0,1\} = \{0,\frac{1}{2},1\}$.

So we have $R_2 \not\sqsubseteq_{\text{pmay}} R_1$ and $R_1 \not\sqsubseteq_{\text{pmust}} R_2$.

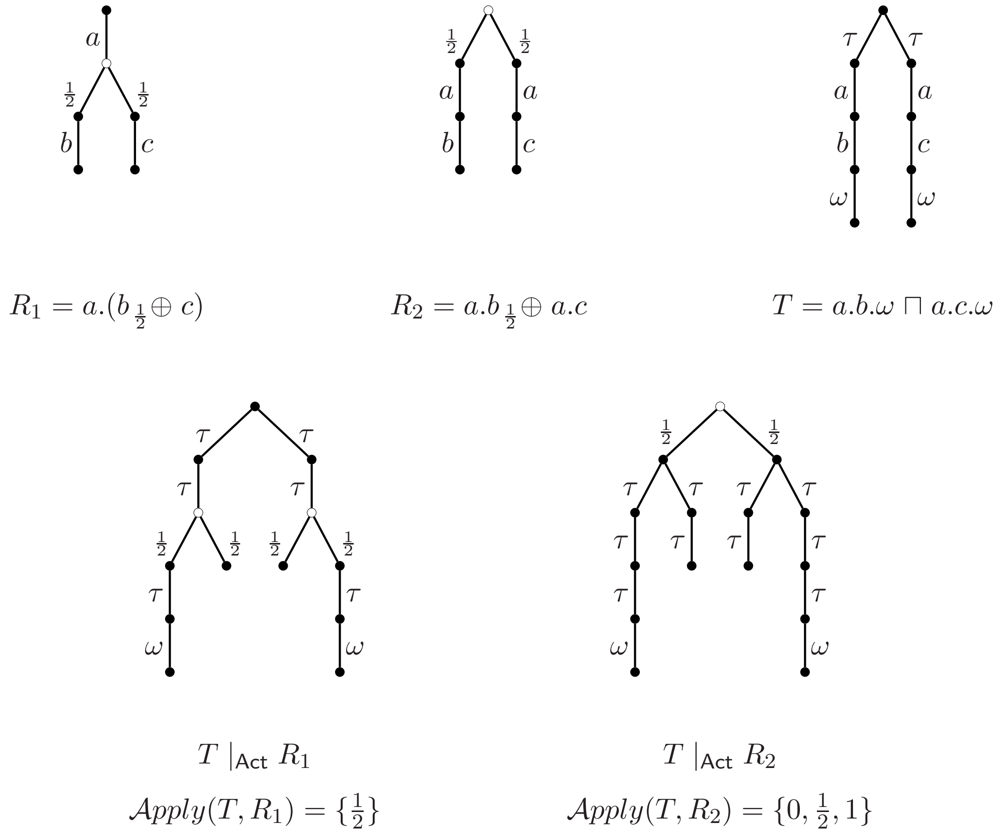


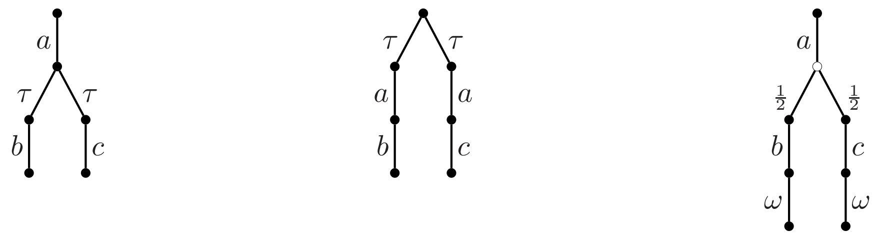
Fig. 5. Counterexample: action prefix does not distribute over probabilistic choice

Example 4.2 The axiom $a.(P \sqcap Q) = a.P \sqcap a.Q$ is unsound.

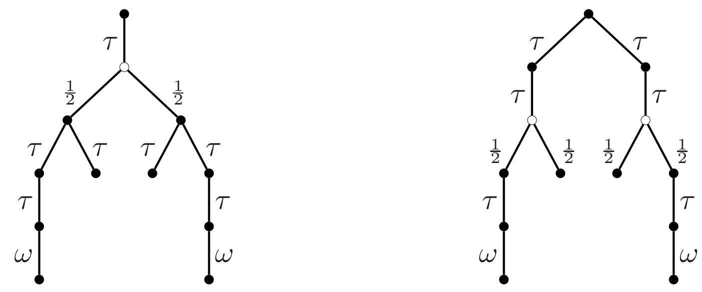
It is well known that this axiom is valid in the standard theory of testing, for non-probabilistic processes. However, consider the instance R_1 and R_2 in Figure 6, and note that these processes do not contain any probabilistic choice. But they can be differentiated by the probabilistic test $T = a.(b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)$; the details are in Figure 6. There is only one possible outcome from applying T to R_2, the probability $\frac{1}{2}$, because the nondeterministic choice is made before the probabilistic choice. On the other hand when T is applied to R_1 there are three possible outcomes, 0, $\frac{1}{2}$ and 1, because effectively the probabilistic choice takes precedence over the nondeterministic choice. So we have $R_1 \not\sqsubseteq_{\text{pmay}} R_2$ and $R_2 \not\sqsubseteq_{\text{pmust}} R_1$.

Example 4.3 The axiom $a.(P \,\Box\, Q) = a.P \,\Box\, a.Q$ is unsound.

This axiom is valid in the standard may-testing semantics. However, consider the two processes $R_1 = a.(b \,\Box\, c)$, $R_2 = a.b \,\Box\, a.c$ and the probabilistic test $T = a.(b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)$. Now $\mathcal{A}pply(T, R_1) = \{1\}$ and $\mathcal{A}pply(T, R_2) = \{\frac{1}{2}\}$. Therefore $R_1 \not\sqsubseteq_{\text{pmay}} R_2$ and $R_1 \not\sqsubseteq_{\text{pmust}} R_2$.



$R_1 = a.(b \sqcap c)$ $R_2 = a.b \sqcap a.c$ $T = a.(b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)$



$T \mid_{\mathsf{Act}} R_1$ $T \mid_{\mathsf{Act}} R_2$

$\mathcal{A}pply(T, R_1) = \{0, \frac{1}{2}, 1\}$ $\mathcal{A}pply(T, R_2) = \{\frac{1}{2}\}$

Fig. 6. Counterexample: action prefix does not distribute over internal choice

Example 4.4 The axiom $P = P \,\Box\, P$ is unsound.

Let R_1, R_2 denote $(a \,{}_{\frac{1}{2}}\!\oplus b)$ and $(a \,{}_{\frac{1}{2}}\!\oplus b) \,\Box\, (a \,{}_{\frac{1}{2}}\!\oplus b)$, respectively. It is easy to calculate that $\mathcal{A}pply(a.\omega, R_1) = \{\frac{1}{2}\}$ but, because of the way we interpret external choice as an operator over distributions of states in a pLTS, it turns out that $[\![R_2]\!] = [\![((a \,\Box\, a) \,{}_{\frac{1}{2}}\!\oplus (a \,\Box\, b)) \,{}_{\frac{1}{2}}\!\oplus ((b \,\Box\, a) \,{}_{\frac{1}{2}}\!\oplus (b \,\Box\, b))]\!]$ and so $\mathcal{A}pply(a.\omega, R_2) = \{\frac{3}{4}\}$. Therefore $R_2 \not\sqsubseteq_{\mathrm{pmay}} R_1$ and $R_2 \not\sqsubseteq_{\mathrm{pmust}} R_1$.

Example 4.5 The axiom $P \,{}_p\!\oplus (Q \sqcap R) = (P \,{}_p\!\oplus Q) \sqcap (P \,{}_p\!\oplus R)$ is unsound.

Consider the processes $R_1 = a \,{}_{\frac{1}{2}}\!\oplus (b \sqcap c)$ and $R_2 = (a \,{}_{\frac{1}{2}}\!\oplus b) \sqcap (a \,{}_{\frac{1}{2}}\!\oplus c)$, and the test $T_1 = a.\omega \sqcap (b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)$. In the best of possible worlds, when we apply T_1 to R_1 we obtain probability 1, that is $max(\mathcal{A}pply(T_1, R_1)) = 1$. Informally this is because half the time when it is applied to the subprocess a of R_1, optimistically the sub-test $a.\omega$ is actually run. The other half of the time, when it is applied to the subprocess $(b \sqcap c)$, optimistically the sub-test $T_r = (b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)$ is actually used. And here again, optimistically, we obtain probability 1: whenever the test $b.\omega$ is used it might be applied to the subprocess b, while when $c.\omega$ is used it might be applied to c. Formally we have

$$\begin{aligned}
\mathcal{A}pply(T_1, R_1) &= \tfrac{1}{2} \cdot \mathcal{A}pply(T_1, a) + \tfrac{1}{2} \cdot \mathcal{A}pply(T_1, b \sqcap c) \\
&= \tfrac{1}{2} \cdot (\mathcal{A}pply(a.\omega, a) \cup \mathcal{A}pply(T_r, a)) + \\
&\quad \tfrac{1}{2} \cdot (\mathcal{A}pply(T_1, b) \cup \mathcal{A}pply(T_1, c) \cup \mathcal{A}pply(a.\omega, b \sqcap c) \cup \mathcal{A}pply(T_r, b \sqcap c)) \\
&= \tfrac{1}{2} \cdot (\{1\} \cup \{0\}) + \tfrac{1}{2} \cdot (\{0, \tfrac{1}{2}\} \cup \{0, \tfrac{1}{2}\} \cup \{0\} \cup \{0, \tfrac{1}{2}, 1\}) \\
&= \{0, \tfrac{1}{4}, \tfrac{1}{2}, \tfrac{3}{4}, 1\}
\end{aligned}$$

However no matter how optimistic we are when applying T_1 to R_2 we can never get probability 1; the most we can hope for is $\frac{3}{4}$, which might occur when T_1 is applied to the subprocess $(a \,{}_{\frac{1}{2}}\!\oplus b)$. Specifically when the subprocess a is being tested the sub-test $a.\omega$ might be used, giving probability 1, and when the subprocess b is being tested the sub-test $(b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)$ might be used, giving probability $\frac{1}{2}$. We leave the reader to check that formally

$$\mathcal{A}pply(T_1, R_2) = \{0, \tfrac{1}{4}, \tfrac{1}{2}, \tfrac{3}{4}\}$$

from which we can conclude $R_1 \not\sqsubseteq_{\mathrm{pmay}} R_2$.

We can also show that $R_2 \not\sqsubseteq_{\mathrm{pmust}} R_1$, using the test

$$T_2 = (b.\omega \;\Box\; c.\omega) \sqcap (a.\omega \,{}_{\frac{1}{3}}\!\oplus (b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega)).$$

Reasoning pessimistically, the worst that can happen when applying T_2 to R_1 is we get probability 0. Each time the subprocess a is tested the worst probability will occur when the sub-test $(b.\omega \;\Box\; c.\omega)$ is used; this results in probability 0. Similarly when the subprocess $(b \sqcap c)$ is being tested the subtest $(a.\omega \,{}_{\frac{1}{3}}\!\oplus (b.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega))$ will give probability 0. In other words $min(\mathcal{A}pply(T_2, R_1)) = 0$. When applying T_2 to R_2, things can never be as bad. The worst probability will occur when T_2 is applied to the subprocess $(a \,{}_{\frac{1}{2}}\!\oplus b)$, namely probability $\frac{1}{6}$. We leave the reader to check that formally $\mathcal{A}pply(T_2, R_1) = \{0, \frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}\}$ and $\mathcal{A}pply(T_2, R_2) = \{\frac{1}{6}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}\}$.

Example 4.6 The axiom $P \sqcap (Q \,{}_p\!\oplus R) = (P \sqcap Q) \,{}_p\!\oplus (P \sqcap R)$ is unsound.

Let $R_1 = a \sqcap (b \,{}_{\frac{1}{2}}\!\oplus c)$, $R_2 = (a \sqcap b) \,{}_{\frac{1}{2}}\!\oplus (a \sqcap c)$ and $T = a.(\omega \,{}_{\frac{1}{2}}\!\oplus \mathbf{0}) \;\Box\; b.\omega$. One can check that $\mathcal{A}pply(T, R_1) = \{\frac{1}{2}\}$ and $\mathcal{A}pply(T, R_2) = \frac{1}{2}\{\frac{1}{2}, 1\} + \frac{1}{2}\{\frac{1}{2}, 0\} = \{\frac{1}{4}, \frac{1}{2}, \frac{3}{4}\}$. Therefore we have $R_2 \not\sqsubseteq_{\mathrm{pmay}} R_1$ and $R_1 \not\sqsubseteq_{\mathrm{pmust}} R_2$.

Example 4.7 The axiom $P \;\Box\; (Q \sqcap R) = (P \;\Box\; Q) \sqcap (P \;\Box\; R)$ is unsound.

Let $R_1 = (a \,{}_{\frac{1}{2}}\!\oplus b) \;\Box\; (c \sqcap d)$, $R_2 = ((a \,{}_{\frac{1}{2}}\!\oplus b) \;\Box\; c) \sqcap ((a \,{}_{\frac{1}{2}}\!\oplus b) \;\Box\; d)$ and $T = (a.\omega \,{}_{\frac{1}{2}}\!\oplus c.\omega) \sqcap (b.\omega \,{}_{\frac{1}{2}}\!\oplus d.\omega)$. This time we get $\mathcal{A}pply(T, R_1) = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1\}$ and $\mathcal{A}pply(T, R_2) = \{\frac{1}{4}, \frac{3}{4}\}$. So $R_1 \not\sqsubseteq_{\mathrm{pmay}} R_2$ and $R_2 \not\sqsubseteq_{\mathrm{pmust}} R_1$.

Example 4.8 The axiom $P \sqcap (Q \;\Box\; R) = (P \sqcap Q) \;\Box\; (P \sqcap R)$ is unsound.

Let $R_1 = (a \,{}_{\frac{1}{2}}\!\oplus b) \sqcap ((a \,{}_{\frac{1}{2}}\!\oplus b) \;\Box\; \mathbf{0})$ and $R_2 = ((a \,{}_{\frac{1}{2}}\!\oplus b) \sqcap (a \,{}_{\frac{1}{2}}\!\oplus b)) \;\Box\; ((a \,{}_{\frac{1}{2}}\!\oplus b) \sqcap \mathbf{0})$. One obtains $\mathcal{A}pply(a.\omega, R_1) = \{\frac{1}{2}\}$ and $\mathcal{A}pply(a.\omega, R_2) = \{\frac{1}{2}, \frac{3}{4}\}$. So $R_2 \not\sqsubseteq_{\mathrm{pmay}} R_1$. Let R_3 and R_4 result from substituting $a \,{}_{\frac{1}{2}}\!\oplus b$ for each of P, Q and R in the axiom above. Now $\mathcal{A}pply(a.\omega, R_3) = \{\frac{1}{2}, \frac{3}{4}\}$ and $\mathcal{A}pply(a.\omega, R_4) = \{\frac{3}{4}\}$. So $R_4 \not\sqsubseteq_{\mathrm{pmust}} R_3$.

Example 4.9 The axiom $P\,{}_p\!\oplus (Q \,\Box\, R) = (P\,{}_p\!\oplus Q) \,\Box\, (P\,{}_p\!\oplus R)$ is unsound.

Let $R_1 = a\,{}_{\frac{1}{2}}\!\oplus (b \,\Box\, c)$, $R_2 = (a\,{}_{\frac{1}{2}}\!\oplus b) \,\Box\, (a\,{}_{\frac{1}{2}}\!\oplus c)$ and $R_3 = (a \,\Box\, b)\,{}_{\frac{1}{2}}\!\oplus (a \,\Box\, c)$. R_1 is an instance of the left-hand side of the axiom, and R_2 an instance of the right-hand side. Here we use R_3 as a tool to reason about R_2, but in Section 8 we need R_3 in its own right. Note that $[\![R_2]\!] = \frac{1}{2}\cdot[\![R_1]\!] + \frac{1}{2}\cdot[\![R_3]\!]$. Let $T = a.\omega$. It is easy to see that $\mathcal{A}pply(T, R_1) = \{\frac{1}{2}\}$ and $\mathcal{A}pply(T, R_3) = \{1\}$. Therefore $\mathcal{A}pply(T, R_2) = \{\frac{3}{4}\}$. So we have $R_2 \not\sqsubseteq_{\text{pmay}} R_1$ and $R_2 \not\sqsubseteq_{\text{pmust}} R_1$.

Of all the examples in this section, this is the only one for which we can show that $\sqsubseteq_{\text{pmay}}$ and $\sqsupseteq_{\text{pmay}}$ both fail, i.e. both inequations that can be associated with the axiom are unsound for *may* testing. Let $T = a.(\omega\,{}_{\frac{1}{2}}\!\oplus \mathbf{0}) \sqcap (b.\omega\,{}_{\frac{1}{2}}\!\oplus c.\omega)$. It is not hard to check that $\mathcal{A}pply(T, R_1) = \{0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}\}$ and $\mathcal{A}pply(T, R_3) = \{\frac{1}{2}\}$. Thus $\mathcal{A}pply(T, R_2) = \{\frac{1}{4}, \frac{3}{8}, \frac{1}{2}, \frac{5}{8}\}$. Therefore, we have $R_1 \not\sqsubseteq_{\text{pmay}} R_2$.

For future reference, we also observe that $R_1 \not\sqsubseteq_{\text{pmay}} R_3$ and $R_3 \not\sqsubseteq_{\text{pmay}} R_1$.

5 Must versus may testing

On pCSP there are two differences between the preorders $\sqsubseteq_{\text{pmay}}$ and $\sqsubseteq_{\text{pmust}}$:

- Must testing is more discriminating
- The preorders $\sqsubseteq_{\text{pmay}}$ and $\sqsubseteq_{\text{pmust}}$ are oriented in opposite directions.

In this section we substantiate these claims by proving that $P \sqsubseteq_{\text{pmust}} Q$ implies $Q \sqsubseteq_{\text{pmay}} P$, and by providing a counterexample that shows the implication is strict. We are only able to obtain the implication since our language does not feature *divergence*, infinite sequences of τ-actions. It is well known from the non-probabilistic theory of testing [DH84,Hen88] that in the presence of divergence $\simeq_{\text{may}}$ and $\simeq_{\text{must}}$ are incomparable.

To establish a relationship between must testing and may testing, we define the context $C[\_] = \_\;|_{\{\omega\}}\;(\omega \,\Box\, (\nu \sqcap \nu))$ so that for every test T we obtain a new test $C[T]$, by considering ν instead of ω as success action.

Lemma 5.1 For any process P and test T, it holds that

(i) if $p \in \mathcal{A}pply(T, P)$ then $(1-p) \in \mathcal{A}pply(C[T], P)$

(ii) if $p \in \mathcal{A}pply(C[T], P)$ then there exists a $q \in \mathcal{A}pply(T, P)$ such that $1-q \leq p$.

Proof. A state of the form $C[s]\;|_{\mathsf{Act}}\;t$ can always do a τ-move, and never directly a success action ν. The τ-steps that $C[s]\;|_{\mathsf{Act}}\;t$ can do fall into three classes: the resulting distribution is either

- a point distribution $\overline{v}$ with $v \xrightarrow{\nu}$; we call this a *successful* τ-step, because it contributes 1 to the set $\mathcal{R}esults(C[s]\;|_{\mathsf{Act}}\;t)$
- a point distribution $\overline{u}$ with u a state from which the success action ν is unreachable; we call this an *unsuccessful* τ-step, because it contributes 0 to the set $\mathcal{R}esults(C[s]\;|_{\mathsf{Act}}\;t)$
- or a distribution of form $C[\Theta]\;|_{\mathsf{Act}}\;\Delta$.

Note that

- $C[s] \mid_{\mathsf{Act}} t$ can always do a successful τ-step
- $C[s] \mid_{\mathsf{Act}} t$ can do an unsuccessful τ-step iff $s \mid_{\mathsf{Act}} t$ can do a ω-step
- and $C[s] \mid_{\mathsf{Act}} t \xrightarrow{\tau} C[\Theta] \mid_{\mathsf{Act}} \Delta$ iff $s \mid_{\mathsf{Act}} t \xrightarrow{\tau} \Theta \mid_{\mathsf{Act}} \Delta$.

Using this, both claims follow by a straightforward induction on T and P. □

Proposition 5.2 If $P \sqsubseteq_{\mathrm{pmust}} Q$ then $Q \sqsubseteq_{\mathrm{pmay}} P$.

Proof. Suppose $P \sqsubseteq_{\mathrm{pmust}} Q$. We must show that, for any test T, if $p \in \mathcal{A}pply(T, Q)$ then there exists a $q \in \mathcal{A}pply(T, P)$ such that $p \leq q$. So suppose $p \in \mathcal{A}pply(T, Q)$. By the first clause of Lemma 5.1, we have $(1-p) \in \mathcal{A}pply(C[T], Q)$. Given that $P \sqsubseteq_{\mathrm{pmust}} Q$, there must be an $x \in \mathcal{A}pply(C[T], P)$ such that $x \leq 1-p$. By the second clause of Lemma 5.1, there exists a $q \in \mathcal{A}pply(T, P)$ such that $1-q \leq x$. It follows that $p \leq q$. Therefore $Q \sqsubseteq_{\mathrm{pmay}} P$. □

Example 5.3 To show that must testing is strictly more discriminating than may testing consider the processes $a \,\Box\, b$ and $a \sqcap b$, and expose them to test $a.\omega$. It is not hard to see that $\mathcal{A}pply(a.\omega, a \,\Box\, b) = \{1\}$, whereas $\mathcal{A}pply(a.\omega, a \sqcap b) = \{0, 1\}$. Since $min(\mathcal{A}pply(a.\omega, a \,\Box\, b)) = 1$ and $min(\mathcal{A}pply(a.\omega, a \sqcap b)) = 0$, using Proposition 2.1 we obtain that $(a \,\Box\, b) \not\sqsubseteq_{\mathrm{pmust}} (a \sqcap b)$.

Since $max(\mathcal{A}pply(a.\omega, a \,\Box\, b)) = max(\mathcal{A}pply(a.\omega, a \sqcap b)) = 1$, as a *may* test, the test $a.\omega$ does not differentiate between $a \,\Box\, b$ and $a \sqcap b$. In fact, we have $(a \sqcap b) \sqsubseteq_{\mathrm{pmay}} (a \,\Box\, b)$, and even $(a \,\Box\, b) \simeq_{\mathrm{pmay}} (a \sqcap b)$, but this cannot be shown so easily, as we would have to consider all possible tests. In Section 6 we will develop a tool to prove statements $P \sqsubseteq_{\mathrm{pmay}} Q$, and apply it to derive the equality above (axiom (**EI**) in Figure 8).

6 Simulations

The examples of Section 4 have been all negative, because one can easily demonstrate an inequivalence between two processes by exhibiting a test which distinguishes them in the appropriate manner. A direct application of the definition of the testing preorders is usually unsuitable for establishing positive results, as this involves a universal quantification over all possible tests that can be applied. To give positive results of the form $P \sqsubseteq_{\mathrm{pmay}} Q$ (and similarly for $P \sqsubseteq_{\mathrm{pmust}} Q$) we need to come up with a preorder $\sqsubseteq_{\mathrm{finer}}$ such that $(P \sqsubseteq_{\mathrm{finer}} Q) \Rightarrow (P \sqsubseteq_{\mathrm{pmay}} Q)$ and statements $P \sqsubseteq_{\mathrm{finer}} Q$ can be obtained by exhibiting a single witness.

In this section we report on investigations in this direction, using *simulations* as our witnesses. We confine ourselves to *may* testing, although similar results hold for *must* testing. The definitions are somewhat complicated by the fact that in a pLTS transitions go from states to distributions; consequently if we are to use sequences of transitions, or *weak transitions* $\stackrel{a}{\Longrightarrow}$ which abstract from sequences of internal actions that might precede or follow the a-transition, then we need to generalise transitions so that they go from distributions to distributions. We first develop the mathematical machinery for doing this.

6.1 Lifting relations

Let $\mathcal{R} \subseteq S \times \mathcal{D}(S)$ be a relation from states to distributions. We lift it to a relation $\overline{\mathcal{R}} \subseteq \mathcal{D}(S) \times \mathcal{D}(S)$ by letting $\Delta_1 \; \overline{\mathcal{R}} \; \Delta_2$ whenever

(i) $\Delta_1 = \sum_{i\in I} p_i \cdot \overline{s_i}$, where I is a finite index set and $\sum_{i\in I} p_i = 1$

(ii) For each $i \in I$ there is a distribution Φ_i such that $s_i \; \mathcal{R} \; \Phi_i$

(iii) $\Delta_2 = \sum_{i\in I} p_i \cdot \Phi_i$.

An important point here is that in the decomposition (i) of Δ_1 into $\sum_{i\in I} p_i \cdot \overline{s_i}$, the states s_i are *not necessarily distinct*: that is, the decomposition is not in general unique. Thus when establishing the relationship between Δ_1 and Δ_2, a given state s in Δ_1 may play a number of different roles, and this is seen clearly if we apply this definition to the action relations $\xrightarrow{\alpha} \subseteq S_p \times \mathcal{D}(S_p)$ in the operational semantics of pCSP. We obtain lifted relations between $\mathcal{D}(S_p)$ and $\mathcal{D}(S_p)$, which to ease the notation we write as $\Delta_1 \xrightarrow{\alpha} \Delta_2$; then, using pCSP terms to represent distributions, a simple instance of a transition between distributions is given by

$$(a.b \;\Box\; a.c) \;{}_{\frac{1}{2}}\!\oplus\; a.d \quad \xrightarrow{a} \quad b \;{}_{\frac{1}{2}}\!\oplus\; d$$

But we also have

$$(a.b \;\Box\; a.c) \;{}_{\frac{1}{2}}\!\oplus\; a.d \quad \xrightarrow{a} \quad (b \;{}_{\frac{1}{2}}\!\oplus\; c) \;{}_{\frac{1}{2}}\!\oplus\; d \tag{1}$$

because, viewed as a distribution, the term $(a.b \;\Box\; a.c) \;{}_{\frac{1}{2}}\!\oplus\; a.d$ may be re-written as $((a.b \;\Box\; a.c) \;{}_{\frac{1}{2}}\!\oplus\; (a.b \;\Box\; a.c)) \;{}_{\frac{1}{2}}\!\oplus\; a.d$ representing the sum of point distributions

$$\tfrac{1}{4} \cdot \overline{(a.b \;\Box\; a.c)} + \tfrac{1}{4} \cdot \overline{(a.b \;\Box\; a.c)} + \tfrac{1}{2} \cdot \overline{a.d}$$

from which the move (1) can easily be derived using the three moves from states

$$a.b \;\Box\; a.c \xrightarrow{a} \overline{b} \qquad\qquad a.b \;\Box\; a.c \xrightarrow{a} \overline{c} \qquad\qquad a.d \xrightarrow{a} \overline{d}$$

The lifting construction satisfies the following two useful properties, whose proofs we leave to the reader.

Proposition 6.1 Suppose $\mathcal{R} \subseteq S \times \mathcal{D}(S)$ and $\sum_{i\in I} p_i = 1$. Then we have

(i) $\Theta_i \; \overline{\mathcal{R}} \; \Delta_i$ implies $(\sum_{i\in I} p_i \cdot \Theta_i) \; \overline{\mathcal{R}} \; (\sum_{i\in I} p_i \cdot \Delta_i)$.

(ii) If $(\sum_{i\in I} p_i \cdot \Theta_i) \; \overline{\mathcal{R}} \; \Delta$ then $\Delta = \sum_{i\in I} p_i \cdot \Delta_i$ for some set of distributions Δ_i such that $\Theta_i \; \overline{\mathcal{R}} \; \Delta_i$. $\Box$

The lifting construction can also be used to define the concept of a *partial* internal move between distributions, one where part of the distribution does an internal move and the remainder remains unchanged. Write $s \xrightarrow{\hat{\tau}} \Delta$ if either $s \xrightarrow{\tau} \Delta$ or $\Delta = \overline{s}$. This relation between states and distributions can be lifted to one between distributions and distributions, and again for notational convenience we use $\Delta_1 \xrightarrow{\hat{\tau}} \Delta_2$ to denote the lifted relation. As an example, again using process terms to denote distributions, we have

$$(a \sqcap b) \,{}_{\frac{1}{2}}\!\oplus\, (a \sqcap c) \quad \xrightarrow{\hat{\tau}} \quad a \,{}_{\frac{1}{2}}\!\oplus\, (a \sqcap b \quad {}_{\frac{1}{2}}\!\oplus \quad c)$$

This follows because as a distribution $(a \sqcap b) \,{}_{\frac{1}{2}}\!\oplus\, (a \sqcap c)$ may be written as

$$\tfrac{1}{4} \cdot \overline{(a \sqcap b)} + \tfrac{1}{4} \cdot \overline{(a \sqcap b)} + \tfrac{1}{4} \cdot \overline{(a \sqcap c)} + \tfrac{1}{4} \cdot \overline{(a \sqcap c)}$$

and we have the four moves from states to distributions:

$$\begin{array}{ll} (a \sqcap b) \xrightarrow{\hat{\tau}} \overline{a} & \qquad (a \sqcap b) \xrightarrow{\hat{\tau}} \overline{(a \sqcap b)} \\ (a \sqcap c) \xrightarrow{\hat{\tau}} \overline{a} & \qquad (a \sqcap c) \xrightarrow{\hat{\tau}} \overline{c} \end{array}$$

6.2 The simulation preorder

Following tradition it would be natural to define simulations as relations between states in a pLTS [JWL01,SL94]. However, technically it is more convenient to use relations in $S_p \leftrightarrow \mathcal{D}(S_p)$. One reason may be understood through the example in Figure 5. Although in Example 4.1 we found that $R_2 \not\sqsubseteq_{\text{pmay}} R_1$, we do have $R_1 \sqsubseteq_{\text{pmay}} R_2$. If we are to relate these processes via a simulation-like relation, then the initial state of R_1 needs to be related to the initial *distribution* of R_2, containing the two states $a.b$ and $a.c$.

Our definition of simulation uses *weak transitions* [Mil89], which have the standard definitions except that they now apply to distributions, and $\xrightarrow{\hat{\tau}}$ is used instead of $\xrightarrow{\tau}$. This reflects the understanding that if a distribution may perform a sequence of internal moves before or after executing a visible action, different parts of the distribution may perform different numbers of internal actions:

- Let $\Delta_1 \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_2$ whenever $\Delta_1 \xrightarrow{\hat{\tau}}{}^{*} \Delta_2$.
- Similarly $\Delta_1 \stackrel{\hat{a}}{\Longrightarrow} \Delta_2$ denotes $\Delta_1 \xrightarrow{\hat{\tau}}{}^{*} \xrightarrow{a} \xrightarrow{\hat{\tau}}{}^{*} \Delta_2$ whenever $a \in \mathsf{Act}$.

Definition 6.2 [Seg95] A relation $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ is said to be a *simulation* if for all $s, \Delta, \alpha, \Theta$ we have that $s \mathrel{\mathcal{R}} \Delta$ and $s \xrightarrow{\alpha} \Theta$ implies there exists some Δ' with $\Delta \stackrel{\hat{\alpha}}{\Longrightarrow} \Delta'$ and $\Theta \mathrel{\overline{\mathcal{R}}} \Delta'$.

We write $s \triangleright_S \Delta$ to mean that there is some simulation $\mathcal{R}$ such that $s \mathrel{\mathcal{R}} \Delta$. Note that the lifting operation on relations is *monotone*, in the sense that $\mathcal{R} \subseteq \mathcal{S}$ implies $\overline{\mathcal{R}} \subseteq \overline{\mathcal{S}}$. Hence $\triangleright_S$, which is the union of all simulations, is a simulation itself. Therefore, $\triangleright_S$ could just as well have been defined co-inductively as the largest relation $\triangleright_S \subseteq S_p \times \mathcal{D}(S_p)$ satisfying, for all $s \in S_p$, $\Delta \in \mathcal{D}(S_p)$ and $\alpha \in \mathsf{Act}_\tau$,

$$s \mathrel{\mathcal{R}} \Delta \wedge s \xrightarrow{\alpha} \Theta \;\Rightarrow\; \exists \Delta'.\, \Delta \stackrel{\hat{\alpha}}{\Longrightarrow} \Delta' \wedge \Theta \mathrel{\overline{\mathcal{R}}} \Delta'$$

Definition 6.3 The *simulation preorder* on process terms is defined by letting $P \sqsubseteq_S Q$ whenever there is a distribution Δ with $[\![Q]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta$ and $[\![P]\!] \mathrel{\overline{\triangleright_S}} \Delta$. The symmetric closure of $\sqsubseteq_S$ is denoted $\simeq_S$.

$P_1 = a.(b \;{}_{\frac{1}{2}}\!\oplus (c \;{}_{\frac{1}{2}}\!\oplus c))$ $\qquad\qquad$ $P_2 = a.((b \;{}_{\frac{1}{2}}\!\oplus b) \;{}_{\frac{1}{2}}\!\oplus c)$

Fig. 7. Two simulation equivalent processes

If $P \in S_p$, that is if P is a state in the pLTS of pCSP and so $[\![P]\!] = \overline{P}$, then to establish $P \sqsubseteq_S Q$ it is sufficient to exhibit a simulation between the state P and the distribution $[\![Q]\!]$, because trivially $s \vartriangleright_S \Delta$ implies $\overline{s}\ \overline{\vartriangleright_S}\ \Delta$.

Example 6.4 Consider the two processes P_i in Figure 7. To show $P_1 \sqsubseteq_S P_2$ it is sufficient to exhibit a simulation $\mathcal{R}$ such that $s_0\ \mathcal{R}\ \overline{t_0}$. Let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be defined by

$$s_0\ \mathcal{R}\ \overline{t_0} \qquad s_1\ \mathcal{R}\ \Delta_t \qquad s_2\ \mathcal{R}\ \overline{t_3} \qquad s_3\ \mathcal{R}\ \overline{t_3} \qquad \mathbf{0}\ \mathcal{R}\ \overline{\mathbf{0}}$$

where Δ_t is the two-point distribution mapping both t_1 and t_2 to the probability $\frac{1}{2}$. Then it is straightforward to check that it satisfies the requirements of a simulation: the only non-trivial requirement is that $\Delta_1\ \overline{\mathcal{R}}\ \Delta_2$. But this follows from the fact that

$$\begin{array}{lcl} \Delta_1 & = & \frac{1}{2}\cdot\overline{s_1} + \frac{1}{4}\cdot\overline{s_2} + \frac{1}{4}\cdot\overline{s_3} \\ \Delta_2 & = & \frac{1}{2}\cdot\Delta_t + \frac{1}{4}\cdot\overline{t_3} + \frac{1}{4}\cdot\overline{t_3} \end{array}$$

As another example reconsider $R_1 = a.(b \;{}_{\frac{1}{2}}\!\oplus c)$ and $R_2 = a.b \;{}_{\frac{1}{2}}\!\oplus a.c$ from Figure 5, where for convenience we use process terms to denote their semantic interpretations. It is easy to see that $R_1 \sqsubseteq_S R_2$ because of the simulation

$$R_1\ \mathcal{R}\ [\![R_2]\!] \qquad\qquad b\ \mathcal{R}\ \overline{b} \qquad\qquad c\ \mathcal{R}\ \overline{c} \qquad\qquad \mathbf{0}\ \mathcal{R}\ \overline{\mathbf{0}}$$

Namely $R_2 \xrightarrow{a} (b \;{}_{\frac{1}{2}}\!\oplus c)$ and $(b \;{}_{\frac{1}{2}}\!\oplus c)\ \overline{\mathcal{R}}\ (b \;{}_{\frac{1}{2}}\!\oplus c)$.

Similarly $(a \;{}_{\frac{1}{2}}\!\oplus c) \sqcap (b \;{}_{\frac{1}{2}}\!\oplus c) \sqsubseteq_S (a \sqcap b) \;{}_{\frac{1}{2}}\!\oplus c$ because it is possible to find a simulation between the state $(a \;{}_{\frac{1}{2}}\!\oplus c) \sqcap (b \;{}_{\frac{1}{2}}\!\oplus c)$ and the distribution $(a \sqcap b) \;{}_{\frac{1}{2}}\!\oplus c$.

In case $P \notin S_p$, a statement $P \sqsubseteq_S Q$ cannot always be established by a simulation $\mathcal{R}$ such that $[\![P]\!]\ \overline{\mathcal{R}}\ [\![Q]\!]$.

Example 6.5 Compare the processes $P = a \;{}_{\frac{1}{2}}\!\oplus b$ and $P \sqcap P$. Note that $[\![P]\!]$ is the distribution $\frac{1}{2}\,\overline{a} + \frac{1}{2}\,\overline{b}$ whereas $[\![P \sqcap P]\!]$ is the point distribution $\overline{P \sqcap P}$. The relation $\mathcal{R}$ given by

$$(P \sqcap P)\ \mathcal{R}\ (\tfrac{1}{2}\,\overline{a} + \tfrac{1}{2}\,\overline{b}) \qquad a\ \mathcal{R}\ \overline{a} \qquad b\ \mathcal{R}\ \overline{b} \qquad \mathbf{0}\ \mathcal{R}\ \overline{\mathbf{0}}$$

is a simulation, because the τ-step $P \sqcap P \xrightarrow{\tau} (\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b})$ can be matched by the idle transition $(\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b}) \stackrel{\hat{\tau}}{\Longrightarrow} (\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b})$, and we have $(\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b})\ \overline{\mathcal{R}}\ (\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b})$. Thus $(P \sqcap P) \triangleright_S (\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b}) = [\![P]\!]$, hence $[\![P \sqcap P]\!]\ \overline{\triangleright_S}\ [\![P]\!]$, and therefore $P \sqcap P \sqsubseteq_S P$.

This type of reasoning does not apply to the other direction. Any simulation $\mathcal{R}$ with $(\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b})\ \overline{\mathcal{R}}\ \overline{P \sqcap P}$ would have to satisfy $a\ \mathcal{R}\ \overline{P \sqcap P}$ and $b\ \mathcal{R}\ \overline{P \sqcap P}$. However, the move $a \xrightarrow{a} \mathbf{0}$ cannot be matched by the process $\overline{P \sqcap P}$, as the only transition the latter process can do is $\overline{P \sqcap P} \xrightarrow{\tau} (\frac{1}{2}\,\overline{a} +\frac{1}{2}\,\overline{b})$, and only half of that distribution can match the a-move. Thus, no such simulation exists, and we find $[\![P]\!]\ \overline{\not\triangleright_S}\ [\![P \sqcap P]\!]$. Nevertheless, we still have $P \sqsubseteq_S P \sqcap P$. Here, the transition $\stackrel{\hat{\tau}}{\Longrightarrow}$ from Definition 6.3 comes to the rescue. As $[\![P \sqcap P]\!] \stackrel{\hat{\tau}}{\Longrightarrow} [\![P]\!]$ and $[\![P]\!]\ \overline{\triangleright_S}\ [\![P]\!]$, we obtain $P \sqsubseteq_S P \sqcap P$.

Because of the asymmetric use of distributions in the definition of simulation it is not immediately obvious that $\sqsubseteq_S$ is actually a preorder (a reflexive and transitive relation) and hence $\simeq_S$ an equivalence relation. In order to show this, we first need to establish some properties of $\overline{\triangleright_S}$.

Lemma 6.6 Suppose $\sum_{i\in I} p_i = 1$ and $\Delta_i \stackrel{\hat{\alpha}}{\Longrightarrow} \Phi_i$ for each $i\in I$, with I a finite index set. (Recall that $\sum_{i\in I} p_i \cdot \Delta_i$ is only defined for finite I.) Then

$$\sum_{i\in I} p_i \cdot \Delta_i \stackrel{\hat{\alpha}}{\Longrightarrow} \sum_{i\in I} p_i \cdot \Phi_i$$

Proof. We first prove the case $\alpha = \tau$. For each $i\in I$ there is a number k_i such that $\Delta_i = \Delta_{i0} \xrightarrow{\hat{\tau}} \Delta_{i1} \xrightarrow{\hat{\tau}} \Delta_{i2} \xrightarrow{\hat{\tau}} \cdots \xrightarrow{\hat{\tau}} \Delta_{ik_i} = \Delta'_i$. Let $k = \mathit{max}\{k_i \mid i\in I\}$, using that I is finite. Since we have $\Phi \xrightarrow{\hat{\tau}} \Phi$ for any $\Phi\in\mathcal{D}(S)$, we can add spurious transitions to these sequences, until all k_i equal k. After this preparation the lemma follows by k applications of Proposition 6.1(i), taking $\xrightarrow{\hat{\tau}}$ for $\mathcal{R}$.

The case $\alpha\in\mathsf{Act}$ now follows by one more application of Proposition 6.1(i), this time with $\mathcal{R} = \xrightarrow{a}$, preceded and followed by an application of the case $\alpha = \tau$. $\square$

Lemma 6.7 Suppose $\Delta\ \overline{\triangleright_S}\ \Phi$ and $\Delta \xrightarrow{\alpha} \Delta'$. Then $\Phi \stackrel{\hat{\alpha}}{\Longrightarrow} \Phi'$ for some Φ' such that $\Delta'\ \overline{\triangleright_S}\ \Phi'$.

Proof. First $\Delta\ \overline{\triangleright_S}\ \Phi$ means that

$$\Delta = \sum_{i\in I} p_i \cdot \overline{s_i}, \qquad s_i \triangleright_S \Phi_i, \qquad \Phi = \sum_{i\in I} p_i \cdot \Phi_i\ ; \tag{2}$$

also $\Delta \xrightarrow{\alpha} \Delta'$ means

$$\Delta = \sum_{j\in J} q_j \cdot \overline{t_j}, \qquad t_j \xrightarrow{\alpha} \Theta_j, \qquad \Delta' = \sum_{j\in J} q_j \cdot \Theta_j\ , \tag{3}$$

and we can assume *w.l.o.g.* that all the coefficients p_i, q_j are non-zero. Now define $I_j = \{\, i\in I \mid s_i = t_j \,\}$ and $J_i = \{\, j\in J \mid t_j = s_i \,\}$, so that trivially

$$\{(i,j) \mid i \in I,\ j\in J_i\} \qquad = \qquad \{(i,j) \mid j \in J,\ i\in I_j\} \tag{4}$$

and note that

$$\Delta(s_i) = \sum_{j \in J_i} q_j \qquad \text{and} \qquad \Delta(t_j) = \sum_{i \in I_j} p_i \tag{5}$$

Because of (5) we have

$$\begin{aligned}\Phi = \sum_{i \in I} p_i \cdot \Phi_i &= \sum_{i \in I} p_i \cdot \sum_{j \in J_i} \frac{q_j}{\Delta(s_i)} \cdot \Phi_i \\ &= \sum_{i \in I} \sum_{j \in J_i} \frac{p_i \cdot q_j}{\Delta(s_i)} \cdot \Phi_i\end{aligned}$$

Now for each j in J_i we know that in fact $t_j = s_i$, and so from the middle parts of (2) and (3) we obtain $\Phi_i \stackrel{\hat{\alpha}}{\Longrightarrow} \Phi_{ij}$ such that $\Theta_j \mathrel{\overline{\rhd_S}} \Phi_{ij}$. Lemma 6.6 yields

$$\Phi \stackrel{\hat{\alpha}}{\Longrightarrow} \Phi' = \sum_{i \in I} \sum_{j \in J_i} \frac{p_i \cdot q_j}{\Delta(s_i)} \cdot \Phi_{ij}$$

where within the summations $s_i = t_j$, so that, using (4), Φ' can also be written as

$$\sum_{j \in J} \sum_{i \in I_j} \frac{p_i \cdot q_j}{\Delta(t_j)} \cdot \Phi_{ij} \tag{6}$$

All that remains is to show that $\Delta' \mathrel{\overline{\rhd_S}} \Phi'$, which we do by manipulating Δ' so that it takes on a form similar to that in (6):

$$\begin{aligned}\Delta' &= \sum_{j \in J} q_j \cdot \Theta_j \\ &= \sum_{j \in J} q_j \cdot \sum_{i \in I_j} \frac{p_i}{\Delta(t_j)} \cdot \Theta_j \qquad \text{using (5) again} \\ &= \sum_{j \in J} \sum_{i \in I_j} \frac{p_i \cdot q_j}{\Delta(t_j)} \cdot \Theta_j\end{aligned}$$

Comparing this with (6) above we see that the required result, $\Delta' \mathrel{\overline{\rhd_S}} \Phi'$, follows from an application of Proposition 6.1(i). □

Lemma 6.8 Suppose $\Delta \mathrel{\overline{\rhd_S}} \Phi$ and $\Delta \stackrel{\hat{\alpha}}{\Longrightarrow} \Delta'$. Then $\Phi \stackrel{\hat{\alpha}}{\Longrightarrow} \Phi'$ for some Φ' such that $\Delta' \mathrel{\overline{\rhd_S}} \Phi'$.

Proof. First we consider two claims

(i) If $\Delta \mathrel{\overline{\rhd_S}} \Phi$ and $\Delta \stackrel{\hat{\tau}}{\longrightarrow} \Delta'$, then $\Phi \stackrel{\hat{\tau}}{\Longrightarrow} \Phi'$ for some Φ' such that $\Delta' \mathrel{\overline{\rhd_S}} \Phi'$.

(ii) If $\Delta \mathrel{\overline{\rhd_S}} \Phi$ and $\Delta \stackrel{\hat{\tau}}{\Longrightarrow} \Delta'$, then $\Phi \stackrel{\hat{\tau}}{\Longrightarrow} \Phi'$ for some Φ' such that $\Delta' \mathrel{\overline{\rhd_S}} \Phi'$.

The proof of claim (i) is similar to the proof of Lemma 6.7. Claim (ii) follows from claim (i) by induction on the length of the derivation of $\stackrel{\hat{\tau}}{\Longrightarrow}$. By combining claim (ii) with Lemma 6.7, we obtain the required result. □

Proposition 6.9 The relation $\overline{\triangleright_S}$ is both reflexive and transitive on distributions.

Proof. We leave reflexivity to the reader; it relies on the fact that $s \triangleright_S \overline{s}$ for every state s.

For transitivity, let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be given by $s \mathrel{\mathcal{R}} \Phi$ iff $s \triangleright_S \Delta \mathrel{\overline{\triangleright_S}} \Phi$ for some intermediate distribution Δ. Transitivity follows from the two claims

(i) $\Theta \mathrel{\overline{\triangleright_S}} \Delta \mathrel{\overline{\triangleright_S}} \Phi$ implies $\Theta \mathrel{\overline{\mathcal{R}}} \Phi$

(ii) $\mathcal{R}$ is a simulation, hence $\overline{\mathcal{R}} \subseteq \overline{\triangleright_S}$.

Claim (ii) is a straightforward application of Lemma 6.8, so let us look at (i). From $\Theta \mathrel{\overline{\triangleright_S}} \Delta$ we have

$$\Theta = \sum_{i \in I} p_i \cdot \overline{s_i}, \qquad s_i \triangleright_S \Delta_i, \qquad \Delta = \sum_{i \in I} p_i \cdot \Delta_i$$

Since $\Delta \mathrel{\overline{\triangleright_S}} \Phi$, from part (ii) of Proposition 6.1 we know $\Phi = \sum_{i \in I} p_i \cdot \Phi_i$ such that $\Delta_i \mathrel{\overline{\triangleright_S}} \Phi_i$. So for each i we have $s_i \mathrel{\mathcal{R}} \Phi_i$, from which it follows that $\Theta \mathrel{\overline{\mathcal{R}}} \Phi$. $\Box$

Corollary 6.10 $\sqsubseteq_S$ is a preorder, i.e. it is reflexive and transitive.

Proof. By combination of Lemma 6.8 and Proposition 6.9. $\Box$

6.3 The simulation preorder is a precongruence

In Theorem 6.13 of this section we establish that the pCSP operators are monotone w.r.t. the simulation preorder $\sqsubseteq_S$, i.e. that $\sqsubseteq_S$ is a precongruence for pCSP. This implies that the pCSP operators are compositional for $\simeq_S$ or, equivalently, that $\simeq_S$ is a congruence for pCSP. The following two lemmas gather some facts we need in the proof of this theorem. Their proofs are straightforward, although somewhat tedious.

Lemma 6.11 (i) If $\Phi \stackrel{\hat{\tau}}{\Longrightarrow} \Phi'$ then $\Phi \mathbin{\Box} \Delta \stackrel{\hat{\tau}}{\Longrightarrow} \Phi' \mathbin{\Box} \Delta$ and $\Delta \mathbin{\Box} \Phi \stackrel{\hat{\tau}}{\Longrightarrow} \Delta \mathbin{\Box} \Phi'$.

(ii) If $\Phi \xrightarrow{a} \Phi'$ then $\Phi \mathbin{\Box} \Delta \xrightarrow{a} \Phi'$ and $\Delta \mathbin{\Box} \Phi \xrightarrow{a} \Phi'$.

(iii) $(\sum_{j \in J} p_j \cdot \Phi_j) \mathbin{\Box} (\sum_{k \in K} q_k \cdot \Delta_k) \;=\; \sum_{j \in J} \sum_{k \in K} (p_j \cdot q_k) \cdot (\Phi_j \mathbin{\Box} \Delta_k)$.

(iv) Given relations $\mathcal{R}, \mathcal{R}' \subseteq S_p \times \mathcal{D}(S_p)$ satisfying $s \mathcal{R}' \Delta$ whenever $s = s_1 \mathbin{\Box} s_2$ and $\Delta = \Delta_1 \mathbin{\Box} \Delta_2$ with $s_1 \mathrel{\mathcal{R}} \Delta_1$ and $s_2 \mathrel{\mathcal{R}} \Delta_2$. Then $\Phi_i \mathrel{\overline{\mathcal{R}}} \Delta_i$ for $i = 1, 2$ implies $(\Phi_1 \mathbin{\Box} \Phi_2) \mathrel{\overline{\mathcal{R}'}} (\Delta_1 \mathbin{\Box} \Delta_2)$. $\Box$

Lemma 6.12 (i) If $\Phi \stackrel{\hat{\tau}}{\Longrightarrow} \Phi'$ then $\Phi \mid_A \Delta \stackrel{\hat{\tau}}{\Longrightarrow} \Phi' \mid_A \Delta$ and $\Delta \mid_A \Phi \stackrel{\hat{\tau}}{\Longrightarrow} \Delta \mid_A \Phi'$.

(ii) If $\Phi \xrightarrow{a} \Phi'$ and $a \notin A$ then $\Phi \mid_A \Delta \xrightarrow{a} \Phi' \mid_A \Delta$ and $\Delta \mid_A \Phi \xrightarrow{a} \Delta \mid_A \Phi'$.

(iii) If $\Phi \xrightarrow{a} \Phi'$, $\Delta \xrightarrow{a} \Delta'$ and $a \in A$ then $\Delta \mid_A \Phi \xrightarrow{\tau} \Delta' \mid_A \Phi'$.

(iv) $(\sum_{j \in J} p_j \cdot \Phi_j) \mid_A (\sum_{k \in K} q_k \cdot \Delta_k) = \sum_{j \in J} \sum_{k \in K} (p_j \cdot q_k) \cdot (\Phi_j \mid_A \Delta_k)$.

(v) Given relations $\mathcal{R}, \mathcal{R}' \subseteq S_p \times \mathcal{D}(S_p)$ satisfying $s \mathcal{R}' \Delta$ whenever $s = s_1 \mid_A s_2$ and $\Delta = \Delta_1 \mid_A \Delta_2$ with $s_1 \mathrel{\mathcal{R}} \Delta_1$ and $s_2 \mathrel{\mathcal{R}} \Delta_2$. Then $\Phi_i \mathrel{\overline{\mathcal{R}}} \Delta_i$ for $i = 1, 2$ implies $(\Phi_1 \mid_A \Phi_2) \mathrel{\overline{\mathcal{R}'}} (\Delta_1 \mid_A \Delta_2)$. $\Box$

Theorem 6.13 Suppose $P_i \sqsubseteq_S Q_i$ for $i = 1, 2$. Then

(i) $a.P_1 \sqsubseteq_S a.Q_1$

(ii) $P_1 \sqcap P_2 \sqsubseteq_S Q_1 \sqcap Q_2$

(iii) $P_1 \Box P_2 \sqsubseteq_S Q_1 \Box Q_2$

(iv) $P_1 \,{}_p\!\oplus P_2 \sqsubseteq_S Q_1 \,{}_p\!\oplus Q_2$

(v) $P_1 |_A P_2 \sqsubseteq_S Q_1 |_A Q_2$

Proof.

(i) Since $P_1 \sqsubseteq_S Q_1$, there must be a Δ_1 such that $[\![Q_1]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_1$ and $[\![P_1]\!] \mathrel{\overline{\triangleright_S}} \Delta_1$. It is easy to see that $a.P_1 \triangleright_S \overline{a.Q_1}$ because the transition $a.P_1 \xrightarrow{a} [\![P_1]\!]$ can be matched by $\overline{a.Q_1} \xrightarrow{a} [\![Q_1]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_1$. Thus $[\![a.P_1]\!] = \overline{a.P_1} \mathrel{\overline{\triangleright_S}} \overline{a.Q_1} = [\![a.Q_1]\!]$.

(ii) Since $P_i \sqsubseteq_S Q_i$, there must be a Δ_i such that $[\![Q_i]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_i$ and $[\![P_i]\!] \mathrel{\overline{\triangleright_S}} \Delta_i$. It is easy to see that $P_1 \sqcap P_2 \triangleright_S \overline{Q_1 \sqcap Q_2}$ because the transition $P_1 \sqcap P_2 \xrightarrow{\tau} [\![P_i]\!]$, for $i = 1$ or $i = 2$, can be matched by $\overline{Q_1 \sqcap Q_2} \xrightarrow{\tau} [\![Q_i]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_i$. Thus $[\![P_1 \sqcap P_2]\!] = \overline{P_1 \sqcap P_2} \mathrel{\overline{\triangleright_S}} \overline{Q_1 \sqcap Q_2} = [\![Q_1 \sqcap Q_2]\!]$.

(iii) Let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be defined by $s \mathrel{\mathcal{R}} \Delta$ iff either $s \triangleright_S \Delta$ or $s = s_1 \Box s_2$ and $\Delta = \Delta_1 \Box \Delta_2$ with $s_1 \triangleright_S \Delta_1$ and $s_2 \triangleright_S \Delta_2$. We show that $\mathcal{R}$ is a simulation.

Suppose $s_1 \triangleright_S \Delta_1$, $s_2 \triangleright_S \Delta_2$ and $s_1 \Box s_2 \xrightarrow{a} \Theta$ with $a \in \mathsf{Act}$. Then $s_i \xrightarrow{a} \Theta$ for $i = 1$ or $i = 2$. Thus $\Delta_i \stackrel{\hat{a}}{\Longrightarrow} \Delta$ for some Δ with $\Theta \mathrel{\overline{\triangleright_S}} \Delta$, and hence $\Theta \mathrel{\overline{\mathcal{R}}} \Delta$. By Lemma 6.11 we have $\Delta_1 \Box \Delta_2 \stackrel{\hat{a}}{\Longrightarrow} \Delta$.

Now suppose $s_1 \triangleright_S \Delta_1$, $s_2 \triangleright_S \Delta_2$ and $s_1 \Box s_2 \xrightarrow{\tau} \Theta$. Then $s_1 \xrightarrow{\tau} \Phi$ and $\Theta = \Phi \Box \overline{s_2}$ or $s_2 \xrightarrow{\tau} \Phi$ and $\Theta = \overline{s_1} \Box \Phi$. By symmetry we may restrict attention to the first case. Thus $\Delta_1 \stackrel{\hat{\tau}}{\Longrightarrow} \Delta$ for some Δ with $\Phi \mathrel{\overline{\triangleright_S}} \Delta$. By Lemma 6.11 we have $(\Phi \Box \overline{s_2}) \mathrel{\overline{\mathcal{R}}} (\Delta \Box \Delta_2)$ and $\Delta_1 \Box \Delta_2 \stackrel{\hat{\tau}}{\Longrightarrow} \Delta \Box \Delta_2$.

The case that $s \triangleright_S \Delta$ is trivial, so we have checked that $\mathcal{R}$ is a simulation indeed. Using this, we proceed to show that $P_1 \Box P_2 \sqsubseteq_S Q_1 \Box Q_2$.

Since $P_i \sqsubseteq_S Q_i$, there must be a Δ_i such that $[\![Q_i]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_i$ and $[\![P_i]\!] \mathrel{\overline{\triangleright_S}} \Delta_i$. By Lemma 6.11, we have $[\![P_1 \Box P_2]\!] = ([\![P_1]\!] \Box [\![P_2]\!]) \mathrel{\overline{\mathcal{R}}} (\Delta_1 \Box \Delta_2)$. Therefore $[\![P_1 \Box P_2]\!] \mathrel{\overline{\triangleright_S}} (\Delta_1 \Box \Delta_2)$. By Lemma 6.11 we also obtain $[\![Q_1 \Box Q_2]\!] = [\![Q_1]\!] \Box [\![Q_2]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_1 \Box [\![Q_2]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_1 \Box \Delta_2$, so the required result is established.

(iv) Since $P_i \sqsubseteq_S Q_i$, there must be a Δ_i such that $[\![Q_i]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_i$ and $[\![P_i]\!] \mathrel{\overline{\triangleright_S}} \Delta_i$. Thus $[\![Q_1 \,{}_p\!\oplus Q_2]\!] = p \cdot [\![Q_1]\!] + (1-p) \cdot [\![Q_2]\!] \stackrel{\hat{\tau}}{\Longrightarrow} p \cdot \Delta_1 + (1-p) \cdot \Delta_2$ by Lemma 6.6 and $[\![P_1 \,{}_p\!\oplus P_2]\!] = p \cdot [\![P_1]\!] + (1-p) \cdot [\![P_2]\!] \mathrel{\overline{\triangleright_S}} p \cdot \Delta_1 + (1-p) \cdot \Delta_2$ by Proposition 6.1(i). Hence $P_1 \,{}_p\!\oplus P_2 \sqsubseteq_S Q_1 \,{}_p\!\oplus Q_2$.

(v) Let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be defined by $s \mathrel{\mathcal{R}} \Delta$ iff $s = s_1 |_A s_2$ and $\Delta = \Delta_1 |_A \Delta_2$ with $s_1 \triangleright_S \Delta_1$ and $s_2 \triangleright_S \Delta_2$. We show that $\mathcal{R}$ is a simulation. There are three cases to consider.

(a) Suppose $s_1 \triangleright_S \Delta_1$, $s_2 \triangleright_S \Delta_2$ and $s_1 |_A s_2 \xrightarrow{\alpha} \Theta_1 |_A \overline{s_2}$ because of the transition $s_1 \xrightarrow{\alpha} \Theta_1$ with $\alpha \notin A$. Then $\Delta_1 \stackrel{\hat{\alpha}}{\Longrightarrow} \Delta_1'$ for some Δ_1' with $\Theta_1 \mathrel{\overline{\triangleright_S}} \Delta_1'$. By Lemma 6.12 we have $\Delta_1 |_A \Delta_2 \stackrel{\hat{\alpha}}{\Longrightarrow} \Delta_1' |_A \Delta_2$ and also $(\Theta_1 |_A \overline{s_2}) \mathrel{\overline{\mathcal{R}}} (\Delta_1' |_A \Delta_2)$.

(b) The symmetric case can be similarly analysed.

(c) Suppose $s_1 \rhd_S \Delta_1$, $s_2 \rhd_S \Delta_2$ and $s_1 \mid_A s_2 \xrightarrow{\tau} \Theta_1 \mid_A \Theta_2$ because of the transitions $s_1 \xrightarrow{a} \Theta_1$ and $s_2 \xrightarrow{a} \Theta_2$ with $a \in A$. Then for $i = 1$ and $i = 2$ we have $\Delta_i \stackrel{\hat{\tau}}{\Longrightarrow} \Delta'_i \xrightarrow{a} \Delta''_i \stackrel{\hat{\tau}}{\Longrightarrow} \Delta'''_i$ for some $\Delta'_i, \Delta''_i, \Delta'''_i$ with $\Theta_1 \overline{\rhd_S} \Delta'''_i$. By Lemma 6.12 we have $\Delta_1 \mid_A \Delta_2 \stackrel{\hat{\tau}}{\Longrightarrow} \Delta'_1 \mid_A \Delta'_2 \xrightarrow{\tau} \Delta''_1 \mid_A \Delta''_2 \stackrel{\hat{\tau}}{\Longrightarrow} \Delta'''_1 \mid_A \Delta'''_2$ and $(\Theta_1 \mid_A \Theta_2) \overline{\mathcal{R}} (\Delta'''_1 \mid_A \Delta'''_2)$.

So we have checked that $\mathcal{R}$ is a simulation.

Since $P_i \sqsubseteq_S Q_i$, there must be a Δ_i such that $[\![Q_i]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_i$ and $[\![P_i]\!] \overline{\rhd_S} \Delta_i$. By Lemma 6.12 we have $[\![P_1 \mid_A P_2]\!] = ([\![P_1]\!] \mid_A [\![P_2]\!]) \overline{\mathcal{R}} (\Delta_1 \mid_A \Delta_2)$. Therefore $[\![P_1 \mid_A P_2]\!] \overline{\rhd_S} (\Delta_1 \mid_A \Delta_2)$. By Lemma 6.12 we also obtain $[\![Q_1 \mid_A Q_2]\!] = [\![Q_1]\!] \mid_A [\![Q_2]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_1 \mid_A [\![Q_2]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta_1 \mid_A \Delta_2$, which had to be established. □

6.4 Simulation is sound for may testing

This section is devoted to the proof that $P \sqsubseteq_S Q$ implies $P \sqsubseteq_{\mathrm{pmay}} Q$. It involves a slightly different method of analysing the transition systems which result from applying a test to a process. Instead of collecting the set of probabilities of success it calculates the maximum probability that *some* action can be performed. Let $maxlive : S_p \to [0,1]$ be defined by

$$maxlive(s) = \begin{cases} 1 & \text{if } s \xrightarrow{a} \text{ for some } a \in \mathsf{Act}^\omega \\ max(\{\, maxlive(\Delta) \mid s \xrightarrow{\tau} \Delta \,\}) & \text{if } s \not\xrightarrow{a} \text{ for all } a \in \mathsf{Act}^\omega \text{ and } s \xrightarrow{\tau} \\ 0 & \text{otherwise} \end{cases}$$

where

$$maxlive(\Delta) = \sum_{s \in \lceil \Delta \rceil} \Delta(s) \cdot maxlive(s)$$

If $s \in S_p$ only ever gives rise to the special action ω then it is easy to see that $maxlive(s) = max(\mathcal{R}esults(s))$. Because of this fact the following result is straightforward:

Proposition 6.14 $P \sqsubseteq_{\mathrm{pmay}} Q$ if and only if for every test T we have

$$maxlive([\![T \mid_{\mathsf{Act}} P]\!]) \leq maxlive([\![T \mid_{\mathsf{Act}} Q]\!])$$ □

The main technical property we require of $maxlive(\_)$ is that it does not increase as τ-transitions are performed:

Lemma 6.15 $\Delta_1 \xrightarrow{\hat{\tau}}{}^* \Delta_2$ implies $maxlive(\Delta_1) \geq maxlive(\Delta_2)$.

Proof. First we prove one special case:

$$\Delta_1 \xrightarrow{\hat{\tau}} \Delta_2 \text{ implies } maxlive(\Delta_1) \geq maxlive(\Delta_2)\,. \tag{7}$$

We know from $\Delta_1 \xrightarrow{\hat{\tau}} \Delta_2$ that

$$\Delta_1 = \sum_{i \in I} p_i \cdot \overline{s_i}, \qquad s_i \xrightarrow{\hat{\tau}} \Phi_i, \qquad \Delta_2 = \sum_{i \in I} p_i \cdot \Phi_i \tag{8}$$

The second part of (8) means that (i) either $\Phi_i = \overline{s_i}$ or (ii) $s_i \xrightarrow{\tau} \Phi_i$. In case (i) we have $maxlive(s_i) = maxlive(\Phi_i)$; in case (ii) we know from the definition of $maxlive(\_)$ that $maxlive(s_i) \geq maxlive(\Phi_i)$. Therefore,

$$\begin{aligned} maxlive(\Delta_1) &= \textstyle\sum_{i\in I} p_i \cdot maxlive(s_i) \\ &\geq \textstyle\sum_{i\in I} p_i \cdot maxlive(\Phi_i) \\ &= maxlive(\Delta_2) \end{aligned}$$

This completes the proof of (7), from which the general case follows by transition induction. □

This lemma is the main ingredient to the following result:

Proposition 6.16 Suppose $\mathcal{R}$ is a simulation. Then

(i) $s \mathrel{\mathcal{R}} \Delta$ implies $maxlive(s) \leq maxlive(\Delta)$

(ii) $\Theta \mathrel{\overline{\mathcal{R}}} \Delta$ implies $maxlive(\Theta) \leq maxlive(\Delta)$.

Proof. Given that the states of our pLTS are pCSP expressions, there exists a well-founded order on the combination of states in S_p and distributions in $\mathcal{D}(S_p)$, such that $s \xrightarrow{\alpha} \Delta$ implies that s is larger than Δ, and any distribution is larger than the states in its support. We prove (i) and (ii) by simultaneous induction on this order, applied to s and Θ.

(i) We distinguish two cases.

- If $s \xrightarrow{a} \Theta$ for some action $a \in \mathsf{Act}$ and distribution Θ, then $maxlive(s) = 1$. Since $s \mathrel{\mathcal{R}} \Delta$, there exists Δ', Δ'' such that $\Delta \xrightarrow{\hat{\tau}}{}^* \Delta' \xrightarrow{a} \xrightarrow{\hat{\tau}}{}^* \Delta''$ and $\Theta \mathrel{\overline{\mathcal{R}}} \Delta''$. By definition $maxlive(\Delta') = 1$, and $maxlive(\Delta) \geq maxlive(\Delta')$ by Lemma 6.15. Therefore, $maxlive(\Delta) = 1 = maxlive(s)$.
- If $s \xrightarrow{\tau} \Theta$, then $s \mathrel{\mathcal{R}} \Delta$ implies the existence of some Δ_Θ such that $\Delta \xrightarrow{\hat{\tau}}{}^* \Delta_\Theta$ and $\Theta \mathrel{\overline{\mathcal{R}}} \Delta_\Theta$. By induction, using (ii), $maxlive(\Theta) \leq maxlive(\Delta_\Theta)$. Consequently, we have that

$$\begin{aligned} maxlive(s) &= max(\{maxlive(\Theta) \mid s \xrightarrow{\tau} \Theta\}) \\ &\leq max(\{maxlive(\Delta_\Theta) \mid s \xrightarrow{\tau} \Theta\}) \\ &\leq max(\{maxlive(\Delta) \mid s \xrightarrow{\tau} \Theta\}) && \text{(by Lemma 6.15)} \\ &= maxlive(\Delta) \end{aligned}$$

(ii) $\Theta \mathrel{\overline{\mathcal{R}}} \Delta$ means

$$\Theta = \sum_{i\in I} p_i \cdot \overline{s_i}, \qquad s_i \mathrel{\mathcal{R}} \Delta_i, \qquad \Delta = \sum_{i\in I} p_i \cdot \Delta_i$$

So we can derive that

$$\begin{aligned} \mathit{maxlive}(\Theta) &= \textstyle\sum_{i\in I} p_i \cdot \mathit{maxlive}(s_i) \\ &\leq \textstyle\sum_{i\in I} p_i \cdot \mathit{maxlive}(\Delta_i) && \text{(by induction)} \\ &= \mathit{maxlive}(\Delta) \end{aligned}$$ □

We now have all of the ingredients to prove that showing $P \sqsubseteq_S Q$ is a sound method of establishing $P \sqsubseteq_{\mathrm{pmay}} Q$.

Theorem 6.17 $P \sqsubseteq_S Q$ implies $P \sqsubseteq_{\mathrm{pmay}} Q$.

Proof. Suppose $P \sqsubseteq_S Q$. Because of Proposition 6.14 it is sufficient to prove $\mathit{maxlive}([\![T \mid_{\mathsf{Act}} P]\!]) \leq \mathit{maxlive}([\![T \mid_{\mathsf{Act}} Q]\!])$ for an arbitrary test T. Since $\sqsubseteq_S$ is preserved by the parallel operator we have that $T \mid_{\mathsf{Act}} P \sqsubseteq_S T \mid_{\mathsf{Act}} Q$. (Strictly speaking, since ω is not actually allowed to appear in processes, here we can replace it in T with a fresh action $\nu \in \mathsf{Act}$, and accordingly use $\mid_{\mathsf{Act}\backslash\{\nu\}}$ instead of $\mid_{\mathsf{Act}}$.) By definition $T \mid_{\mathsf{Act}} P \sqsubseteq_S T \mid_{\mathsf{Act}} Q$ means that there is a distribution Δ and a simulation $\mathcal{R}$ such that $[\![T \mid_{\mathsf{Act}} Q]\!] \stackrel{\hat{\tau}}{\Longrightarrow} \Delta$ and $[\![T \mid_{\mathsf{Act}} P]\!] \; \overline{\mathcal{R}} \; \Delta$. The result now follows from the second part of Proposition 6.16 and Lemma 6.15. □

6.5 Some properties of simulations

Because of the co-inductive nature of the definition of simulations we can begin to develop properties of the preorder $\sqsubseteq_S$ on pCSP terms. By Theorem 6.17, any equation $P = Q$ or $P \sqsubseteq Q$ that we show to be sound for $\simeq_S$, respectively $\sqsubseteq_S$, is also sound for $\simeq_{\mathrm{pmay}}$, respectively $\sqsubseteq_{\mathrm{pmay}}$. In section 4 we have seen that many of the equations true for standard testing no longer apply to probabilistic processes. But some interesting identities can be salvaged.

Proposition 6.18 All the equations in Figure 8 are valid for $\simeq_S$ over pCSP.

Proof.

- Case (**I1**): It is clear that $P \sqsubseteq_S P \sqcap P$ since $[\![P \sqcap P]\!] = \overline{P \sqcap P} \xrightarrow{\tau} [\![P]\!]$ and $[\![P]\!] \;\overline{\triangleright_S}\; [\![P]\!]$. For the inverse direction, observe that $P \sqcap P \triangleright_S [\![P]\!]$ because the transition $P \sqcap P \xrightarrow{\tau} [\![P]\!]$ is matched by $[\![P]\!] \stackrel{\hat{\tau}}{\Longrightarrow} [\![P]\!]$. Therefore we have $[\![P \sqcap P]\!] = \overline{P \sqcap P} \;\overline{\triangleright_S}\; [\![P]\!]$, thus $P \sqcap P \sqsubseteq_S P$.
- Case (**I3**): Think for the moment of $P \sqcap Q \sqcap R$ as the ternary instance of a new auxiliary operator $\bigsqcap_{i\in I} P_i$ with I a finite index set, whose operational semantics consists of the transitions $\bigsqcap_{i\in I} P_i \xrightarrow{\tau} P_i$ for $i \in I$. Now we show $P \sqcap Q \sqcap R \simeq_{\mathrm{pmay}} (P \sqcap Q) \sqcap R$, and the proof of $P \sqcap Q \sqcap R \simeq_{\mathrm{pmay}} P \sqcap (Q \sqcap R)$ goes likewise.

  That $P \sqcap Q \sqcap R \triangleright_S (P \sqcap Q) \sqcap R$ follows because a move $\overline{P \sqcap Q \sqcap R} \xrightarrow{\tau} P$ can be simulated by a sequence of two τ-steps from $\overline{(P \sqcap Q) \sqcap R}$. Conversely, that $(P \sqcap Q) \sqcap R \triangleright_S P \sqcap Q \sqcap R$ follows because the move $\overline{(P \sqcap Q) \sqcap R} \xrightarrow{\tau} P \sqcap Q$ can be simulated by the idle $\hat{\tau}$-move $P \sqcap Q \sqcap R \xrightarrow{\hat{\tau}} P \sqcap Q \sqcap R$ and we have $P \sqcap Q \triangleright_S P \sqcap Q \sqcap R$. The latter follows because any outgoing transition of $P \sqcap Q$ is also an outgoing transition of $P \sqcap Q \sqcap R$.

  Given the above, and its obvious extension to the case $|I| > 3$, it doesn't matter how to add brackets to the right-hand sides of (**D3**), (**L6**) and (**L7**); one can just

$$
\begin{array}{lrl}
(\mathbf{I1}) & P \sqcap P & = P \\
(\mathbf{I2}) & P \sqcap Q & = Q \sqcap P \\
(\mathbf{I3}) & (P \sqcap Q) \sqcap R & = P \sqcap (Q \sqcap R) \\
(\mathbf{P1}) & P\,{}_p\!\oplus P & = P \\
(\mathbf{P2}) & P\,{}_p\!\oplus Q & = Q\,{}_{1-p}\!\oplus P \\
(\mathbf{P3}) & (P\,{}_p\!\oplus Q)\,{}_q\!\oplus R & = P\,{}_{p\cdot q}\!\oplus (Q\,{}_{\frac{(1-p)\cdot q}{1-p\cdot q}}\!\oplus R) \\
(\mathbf{E1}) & P \mathrel{\Box} \mathbf{0} & = P \\
(\mathbf{E2}) & P \mathrel{\Box} Q & = Q \mathrel{\Box} P \\
(\mathbf{E3}) & (P \mathrel{\Box} Q) \mathrel{\Box} R & = P \mathrel{\Box} (Q \mathrel{\Box} R) \\
(\mathbf{EI}) & a.P \mathrel{\Box} b.Q & = a.P \sqcap b.Q \\
(\mathbf{D1}) & P \mathrel{\Box} (Q\,{}_p\!\oplus R) & = (P \mathrel{\Box} Q)\,{}_p\!\oplus (P \mathrel{\Box} R) \\
(\mathbf{D2}) & a.P \mathrel{\Box} (Q \sqcap R) & = (a.P \mathrel{\Box} Q) \sqcap (a.P \mathrel{\Box} R) \\
(\mathbf{D3}) & P \mathrel{\Box} Q & = (P_1 \mathrel{\Box} Q) \sqcap (P_2 \mathrel{\Box} Q) \sqcap (P \mathrel{\Box} Q_1) \sqcap (P \mathrel{\Box} Q_2), \\
& & \qquad\qquad \text{provided } P = P_1 \sqcap P_2,\ Q = Q_1 \sqcap Q_2 \\
(\mathbf{L1}) & P \mid_A Q & = Q \mid_A P \\
(\mathbf{L2}) & \mathbf{0} \mid_A \mathbf{0} & = \mathbf{0} \\
(\mathbf{L3}) & \mathbf{0} \mid_A a.P & = \begin{cases} a.(\mathbf{0} \mid_A P) & \text{if } a \notin A \\ \mathbf{0} & \text{if } a \in A \end{cases} \\
(\mathbf{L4}) & \mathbf{0} \mid_A (P \sqcap Q) & = (\mathbf{0} \mid_A P) \sqcap (\mathbf{0} \mid_A Q) \\
(\mathbf{L5}) & a.P \mid_A b.Q & = \begin{cases} \mathbf{0} & \text{if } a, b \in A \text{ and } a \neq b \\ P \mid_A Q & \text{if } a, b \in A \text{ and } a = b \\ a.(P \mid_A b.Q) & \text{if } a \notin A \text{ and } b \in A \\ a.(P \mid_A b.Q) \mathrel{\Box} b.(a.P \mid_A Q) & \text{if } a, b \notin A \end{cases} \\
(\mathbf{L6}) & (P \sqcap Q) \mid_A a.R & = \begin{cases} (P \mid_A a.R) \sqcap (Q \mid_A a.R) \sqcap a.((P \sqcap Q) \mid_A R) & \text{if } a \notin A \\ (P \mid_A a.R) \sqcap (Q \mid_A a.R) & \text{if } a \in A \end{cases} \\
(\mathbf{L7}) & P \mid_A Q & = (P_1 \mid_A Q) \sqcap (P_2 \mid_A Q) \sqcap (P \mid_A Q_1) \sqcap (P \mid_A Q_2) \\
& & \qquad\qquad \text{provided } P = P_1 \sqcap P_2,\ Q = Q_1 \sqcap Q_2 \\
(\mathbf{L8}) & P \mid_A (Q\,{}_p\!\oplus R) & = (P \mid_A Q)\,{}_p\!\oplus (P \mid_A R)
\end{array}
$$

Fig. 8. Some equations

as well think of these right-hand sides as instances of $\bigsqcap_{i\in I} P_i$.

- Case (**I2**): It can easily be verified that $[\![P \sqcap Q]\!] \ \overline{\triangleright_S}\ [\![Q \sqcap P]\!]$ because both sides of the equation have the same outgoing transitions. Similarly for (**D3**), (**L2**–**4**) and (**L7**).
- Case (**P1**): By definition we have $[\![P\,{}_p\!\oplus P]\!] = [\![P]\!]$. Similarly for (**P2**) and (**P3**).
- Case (**E1**): Recall that $\mathbf{0}$ is a deadlock state in S_p, i.e. $\mathbf{0} \not\xrightarrow{\alpha}$ for all $\alpha \in \mathsf{Act}_\tau$. Let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be defined by $s\,\mathcal{R}\,\Delta$ iff either $\Delta = \overline{s}$ or $s = t \,\Box\, \mathbf{0}$ and $\Delta = \overline{t}$ for some $t \in S_p$. It can be checked that $\mathcal{R}$ is a simulation. Let $[\![P]\!] = \sum_{j\in J} p_j \cdot \overline{s_j}$. Then $[\![P \,\Box\, \mathbf{0}]\!] = \sum_{j\in J} p_j \cdot \overline{s_j \,\Box\, \mathbf{0}}$, thus $[\![P \,\Box\, \mathbf{0}]\!]\ \overline{\mathcal{R}}\ [\![P]\!]$ and $[\![P \,\Box\, \mathbf{0}]\!]\ \overline{\triangleright_S}\ [\![P]\!]$. Therefore we have $P \,\Box\, \mathbf{0} \sqsubseteq_S P$. Similarly we can prove $P \sqsubseteq_S P \,\Box\, \mathbf{0}$.
- Case (**E2**): Let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be defined by $s\ \mathcal{R}\ \Delta$ iff either $\Delta = \overline{s}$ or $s = t_1 \,\Box\, t_2$ and $\Delta = \overline{t_2 \,\Box\, t_1}$ for some $t_1, t_2 \in S_p$. It can be checked that $\mathcal{R}$ is a simulation. Now it is easy to show that $[\![P \,\Box\, Q]\!]\ \overline{\triangleright_S}\ [\![Q \,\Box\, P]\!]$, thus $P \,\Box\, Q \sqsubseteq_S Q \,\Box\, P$. The proof of (**L1**) goes likewise.
- Case (**E3**): Let $\mathcal{R} \subseteq S_p \times \mathcal{D}(S_p)$ be defined by $s\,\mathcal{R}\,\Delta$ iff either $s = (t_1 \,\Box\, t_2) \,\Box\, t_3$ and $\Delta = \overline{t_1 \,\Box\, (t_2 \,\Box\, t_3)}$ for some $t_1, t_2, t_3 \in S_p$ or $\Delta = \overline{s}$. It can be checked that $\mathcal{R}$ is a simulation. Again it is easy to show that $[\![(P \,\Box\, Q) \,\Box\, R]\!]\ \overline{\triangleright_S}\ [\![P \,\Box\, (Q \,\Box\, R)]\!]$, hence $(P \,\Box\, Q) \,\Box\, R \sqsubseteq_S P \,\Box\, (Q \,\Box\, R)$. The other direction goes likewise.
- Case (**EI**): Let R_1 and R_2 be the processes on the left and right hand side of the equation. It can be checked that both $[\![R_1]\!]\ \overline{\triangleright_S}\ [\![R_2]\!]$ and $[\![R_2]\!]\ \overline{\triangleright_S}\ [\![R_1]\!]$ hold. In particular, the τ-move from R_2 to $a.P$ can be simulated by the idle move $R_1 \stackrel{\hat{\tau}}{\Longrightarrow} R_1$, since we also have $[\![a.P]\!]\ \overline{\triangleright_S}\ [\![R_1]\!]$. Similarly for (**L6**).
- Case (**D1**): $[\![P \,\Box\, (Q\,{}_p\!\oplus R)]\!] = [\![P]\!] \,\Box\, (p \cdot [\![Q]\!] + (1-p) \cdot [\![R]\!]) = p \cdot ([\![P]\!] \,\Box\, [\![Q]\!]) + (1-p) \cdot ([\![P]\!] \,\Box\, [\![R]\!]) = [\![(P \,\Box\, Q)\,{}_p\!\oplus (P \,\Box\, R)]\!]$. Similarly for (**L8**).
- Cases (**D2**) and (**L5**): It is trivial to construct simulations in both directions. □

It can be argued that the equations (**L1**–**8**) are not all that interesting, because they merely apply to our own parallel composition operator. The reason to mention them here, is that, as we will see below, they allow the parallel operator to be eliminated from pCSP expressions, and we believe the same can be achieved, with similar axioms, for the more traditional parallel composition and hiding operators of CSP.

We also have certain inequations which are valid for $\sqsubseteq_S$, the most obvious being

$$(\mathbf{May1}) \qquad\qquad P \ \sqsubseteq\ P \sqcap Q$$

More inequations are listed in Figure 9. The inequations (**Q1**–**8**) complement the counterexamples 4.1–4.8. In each case, an equation that holds in non-probabilistic may-testing semantics is salvaged in the shape of an inequation, with a counterexample against the opposite direction.

Proposition 6.19 All the inequations in Figure 9 are valid for $\sqsubseteq_S$ over pCSP.

Proof. The soundness of (**Q0**) and (**Q1**) is easy to check. All other inequations can be derived from (**Q0**), (**May1**) and the equations in Figure 8 by using the

$$\begin{array}{lrcl}
(\mathbf{Q0}) & \mathbf{0} & \sqsubseteq & P \\
(\mathbf{Q1}) & a.(P \,{}_p\!\oplus Q) & \sqsubseteq & a.P \,{}_p\!\oplus a.Q \\
(\mathbf{Q2}) & a.P \sqcap a.Q & \sqsubseteq & a.(P \sqcap Q) \\
(\mathbf{Q3}) & a.P \,\Box\, a.Q & \sqsubseteq & a.(P \,\Box\, Q) \\
(\mathbf{Q4}) & P & \sqsubseteq & P \,\Box\, Q \\
(\mathbf{Q5}) & (P \,{}_p\!\oplus Q) \sqcap (P \,{}_p\!\oplus R) & \sqsubseteq & P \,{}_p\!\oplus (Q \sqcap R) \\
(\mathbf{Q6}) & P \sqcap (Q \,{}_p\!\oplus R) & \sqsubseteq & (P \sqcap Q) \,{}_p\!\oplus (P \sqcap R) \\
(\mathbf{Q7}) & (P \,\Box\, Q) \sqcap (P \,\Box\, R) & \sqsubseteq & P \,\Box\, (Q \sqcap R) \\
(\mathbf{Q8}) & P \sqcap (Q \,\Box\, R) & \sqsubseteq & (P \sqcap Q) \,\Box\, (P \sqcap R)
\end{array}$$

Fig. 9. Some inequations

precongruence property of $\sqsubseteq_S$. For example, we can reason

$$\begin{array}{rll}
P & = P \,\Box\, \mathbf{0} & \text{by } (\mathbf{E1}) \\
 & \sqsubseteq P \,\Box\, Q & \text{by } (\mathbf{Q0})
\end{array}$$

so as to obtain (**Q4**).

We can also reason

$$\begin{array}{rll}
a.P \,\Box\, a.Q & \sqsubseteq a.(P \,\Box\, Q) \,\Box\, a.(P \,\Box\, Q) & \text{by } (\mathbf{Q4}) \\
 & = a.(P \,\Box\, Q) \sqcap a.(P \,\Box\, Q) & \text{by } (\mathbf{EI}) \\
 & = a.(P \,\Box\, Q) & \text{by } (\mathbf{I1})
\end{array}$$

so as to obtain (**Q3**).

As another example, we can reason

$$\begin{array}{rll}
P \sqcap (Q \,{}_p\!\oplus R) & = (P \,{}_p\!\oplus P) \sqcap (Q \,{}_p\!\oplus R) & \text{by } (\mathbf{P1}) \\
 & \sqsubseteq ((P \sqcap Q) \,{}_p\!\oplus (P \sqcap R)) \sqcap ((P \sqcap Q) \,{}_p\!\oplus (P \sqcap R)) & \text{by } (\mathbf{May1})(4\times) \\
 & = (P \sqcap Q) \,{}_p\!\oplus (P \sqcap R) & \text{by } (\mathbf{I1})
\end{array}$$

so as to obtain (**Q6**). $\Box$

Another important inequation that follows from (**May1**) and (**P1**) is

$$P \,{}_p\!\oplus Q \;\sqsubseteq\; P \sqcap Q$$

saying that any probabilistic choice can be simulated by an internal choice.

Now with these equations and inequations, together with Theorem 6.13 we have the beginnings of an algebraic theory for pCSP processes. An application of this theory is to show that the use of the parallel composition and external choice operators can be eliminated. Let us use $P =_E Q$ to indicate that $P = Q$ can be derived using applications of the equations in Figure 8, without the inequations. Then define *standard processes* to be the least subset of pCSP satisfying:

(i) $\mathbf{0} \in \text{standardP}$

(ii) $a.P \in \text{standardP}$, whenever $P \in \text{standardP}$

(iii) $P_1 \sqcap P_2 \in \text{standardP}$, whenever $P_i \in \text{standardP}$

(iv) $P_1\,{}_p{\oplus}\, P_2 \in \text{standardP}$, whenever $P_i \in \text{standardP}$.

Proposition 6.20 For every P in pCSP there exists a *standard process* $\text{sf}(P)$ such that $P =_E \text{sf}(P)$.

Proof. By induction on the structure of P. □

7 Another look at CSP

We have already seen that probabilistic tests have greater distinguishing power than purely nondeterministic tests, when applied to processes in CSP, that is those which contain no probabilistic choice. In Example 4.2 we have seen that $a.(P \sqcap Q)$ and $a.P \sqcap a.Q$ can be distinguished using a probabilistic test, while it is well known that they can not be distinguished using non-probabilistic tests [Hen88]. In this section we look briefly at the theory of CSP relative to probabilistic testing; we concentrate on *may* testing, although again similar results can be obtained for *must* testing.

First we have the converse of Theorem 6.17, showing that simulations are both sound and complete with respect to probabilistic *may* testing for this sub-language.

Theorem 7.1 For $P, Q \in \text{CSP}$, $P \sqsubseteq_{\text{pmay}} Q$ if and only if $P \sqsubseteq_S Q$.

Proof. ($\Leftarrow$) This implication is proved in Theorem 6.17.

($\Rightarrow$) Instead of directly showing that $\sqsubseteq_{\text{pmay}}$ implies $\sqsubseteq_S$, we define a seemingly weaker preorder $\sqsubseteq_w$:

$$P \sqsubseteq_w Q \text{ iff for every test } T\text{: } (\mathit{maxlive}([\![T\,|_{\mathsf{Act}}\,P]\!]) = 1) \Rightarrow (\mathit{maxlive}([\![T\,|_{\mathsf{Act}}\,Q]\!]) = 1).$$

By Proposition 6.14, we can see that $\sqsubseteq_{\text{pmay}}$ implies $\sqsubseteq_w$. If we can show that $\sqsubseteq_w$ is a simulation, then we are done because $P \sqsubseteq_{\text{pmay}} Q$ will imply $P \sqsubseteq_S Q$.

Given $P \sqsubseteq_w Q$ and $P \xrightarrow{\alpha} P'$, we want to find a process Q' such that $Q \stackrel{\hat{\alpha}}{\Longrightarrow} Q'$ and $P' \sqsubseteq_w Q'$. We distinguish two cases.

(i) $\alpha = \tau$. Then $[\![T\,|_{\mathsf{Act}}\,P]\!] \xrightarrow{\tau} [\![T\,|_{\mathsf{Act}}\,P']\!]$ for every test T, so $P' \sqsubseteq_w P$ by Lemma 6.15 and hence $P' \sqsubseteq_w Q$. By taking $Q' = Q$, we complete the proof for this case.

(ii) $\alpha = a$ for some $a \in \mathsf{Act}$. We define the set $Z = \{R \mid Q \stackrel{\hat{a}}{\Longrightarrow} R \text{ and } P' \not\sqsubseteq_w R\}$. As Q is a CSP process, Z is always finite. For each $R \in Z$ there must be a test

T_R such that $maxlive([\![T_R \mid_{\mathsf{Act}} P']\!]) = 1$ but $maxlive([\![T_R \mid_{\mathsf{Act}} R]\!]) < 1$. Let T_Z be the test $\bigoplus_{R\in Z} p_R T_R$, where each p_R is a positive probability. Then

$$maxlive([\![T_Z \mid_{\mathsf{Act}} P']\!]) = \sum_{R\in Z} p_R \cdot maxlive([\![T_R \mid_{\mathsf{Act}} P']\!]) = 1$$

and, for all $R' \in Z$,

$$maxlive([\![T_Z \mid_{\mathsf{Act}} R']\!]) = \sum_{R\in Z} p_R \cdot maxlive([\![T_R \mid_{\mathsf{Act}} R']\!]) < 1$$

So we have $maxlive([\![a.T_Z \mid_{\mathsf{Act}} P]\!]) = 1$, which implies $maxlive([\![a.T_Z \mid_{\mathsf{Act}} Q]\!]) = 1$ since $P \sqsubseteq_w Q$. By the definition of $maxlive(\_)$ there exists a Q' such that $Q \xrightarrow{\tau}{}^* \xrightarrow{a} Q'$ and $maxlive([\![T_Z \mid_{\mathsf{Act}} Q']\!]) = 1$. Thus $Q' \notin Z$, and $P' \sqsubseteq_w Q'$. $\square$

We can also give a complete equational characterisation. The extra axiom which is valid for this non-probabilistic sub-language is

$$(\mathbf{May2}) \qquad\qquad P \,\Box\, Q = P \sqcap Q$$

In particular this means that, as with standard testing, there is no difference between internal and external choice. Now let us write $P \sqsubseteq_{E_{\mathrm{may}}} Q$ to denote the fact that $P \sqsubseteq Q$ can be derived using the equations of Figure 8 and the inequations (**May1**) and (**May2**).

Theorem 7.2 For P, $Q \in \mathsf{CSP}$, $P \sqsubseteq_{\mathrm{pmay}} Q$ if and only if $P \sqsubseteq_{E_{\mathrm{may}}} Q$.

Proof. One direction follows from the fact that all the equations and inequations mentioned are valid for $\sqsubseteq_S$. The converse depends on being able to rewrite all processes into a normal form: $\Box_{i\in I}\, a_i.P_i$, where I is a finite and possibly empty index set, is a *normal form* if each P_i is in turn a normal form. This conversion into normal form is enabled by Proposition 6.20 and (**May2**). Using (**May1**), (**E1–3**) and the law $P \,\Box\, P = P$, which follows from (**May2**) and (**I1**), it is straightforward to show that for normal forms

$$\Box_{i\in I} a_i.P_i \;\sqsubseteq_S\; \Box_{j\in J} b_j.Q_j \;\text{ implies }\; \Box_{i\in I} a_i.P_i \;\sqsubseteq_{E_{\mathrm{may}}}\; \Box_{j\in J} b_j.Q_j$$ $\square$

8 Related work

Models for probabilistic concurrent systems have been studied for a long time [Rab63,Der70,Var85,JP89]. One of the first models obtained as a simple adaptation of the traditional labelled transition systems from concurrency theory appears in [LS91]. Their *probabilistic transition systems* are classical labelled transition systems, where in addition every transition is labelled with a probability, a real number in the interval [0,1], such that for every state s and every action a, the probabilities of all a-labelled transitions leaving s sum up to either 0 or 1.

In [GJS90] a similar model was proposed, but where the probabilities of *all* transitions leaving s sum up to either 0 or 1. [GSST90] propose the terminology *reactive* for the type of model studied in [LS91], and *generative* for the type of model studied in [GJS90]. In a generative model, a process can be considered to spontaneously generate actions, unless restricted by the environment; in generating actions, a probabilistic choice is made between all transitions that can be taken from a given state, even if they have different labels. In a reactive model, on the other hand, processes are supposed to perform actions only in response to requests by the environment. The choice between two different actions is therefore not under the control of the process itself. When the environment requests a specific action, a probabilistic choice is made between all transitions (if any) that are labelled with the requested action.

In the above-mentioned models, the nondeterministic choice that can be modelled by non-probabilistic labelled transition systems is *replaced* by a probabilistic choice (and in the generative model also a deterministic choice, a choice between different actions, is made probabilistic). Hence reactive and generative probabilistic transition systems do not generalise non-probabilistic labelled transition systems. A model, or rather a calculus, that features both nondeterministic and reactive probabilistic choice was proposed in [HJ90]. It was slightly reformulated in [SL94] under the name *simple probabilistic automata*, and is essentially the same model we use in this paper.

Following the classification above, our model is reactive rather than generative. The reactive model of [LS91] can be reformulated by saying that a state s has at most one outgoing transition for any action a, and this transition ends in a probability distribution over its successor states. The generalisation of [SL94], that we use here as well, is that a state can have multiple outgoing transitions with the same label, each ending in a probability distribution. Simple probabilistic automata are a special case of the *probabilistic automata* of [SL94], that also generalise the generative models of probabilistic processes to a setting with nondeterministic choice.

8.1 Bisimulation, and the alternating approach

Whereas the testing semantics explored in the present paper is based on the idea that processes should be *distinguished* only when there is a compelling reason to do so, (strong) bisimulation semantics [Mil89] is based on the idea that processes should be *identified* only when there is a compelling reason to do so. It has been extended to reactive probabilistic processes in [LS91], to generative ones in [GSST90], and to processes combining nondeterminism and probability in [HJ90]. The latter paper also features a complete axiomatisation of a probabilistic extension of recursion-free CCS.

Weak and branching bisimulation [Mil89,GW96] are versions of strong bisimulation that respect the hidden nature of the internal action τ. Generalisations of these notions to nondeterministic probabilistic processes appear, amongst others, in [SL94,Seg95,PLS00,AB01,BS01,DP05,AW06], with complete axiomatisations reported in [BS01,DP05,DPP05,ABW06]. The authors of these paper tend to distinguish whether they work in an *alternating* [PLS00,AB01,AW06,ABW06] or a *non-alternating* model of probabilistic processes [SL94,Seg95,DP05,DPP05], the two approaches being compared in [BS01]. The non-alternating model stems from [SL94] and is similar to our model of Section 3.2. The alternating model is attributed to [HJ90], and resembles our graphical representation of processes in Section 3.4. It is easy to see that mathematically the alternating and non-alternating model can be translated into each other without loss of information [BS01]. The difference between the two is one of interpretation. In the alternating interpretation, the nodes of form $\circ$ in our graphical representations are interpreted as actual states a process can be in, whereas in the non-alternating representation they are not. Take for example the process $R_1 = a.(b\ {}_{\frac{1}{2}}\!\oplus\ c)$ depicted in Figure 5. In the alternating representation this process passes through a state in which a has already happened, but the probabilistic choice between b and c has not yet been made. In the non-alternating interpretation on the other hand the execution of a is what constitutes this probabilistic choice; after doing a there is a fifty-fifty change of ending up in either state. Although in strong bisimulation semantics the alternating and non-alternating interpretation lead to the same semantic equivalence, in weak and branching bisimulation semantics the resulting equivalences are different, as illustrated in [PLS00,BS01,AW06]. Our testing and simulation preorders as presented here can be classified as non-alternating; however, we believe that an alternating approach would lead to the very same preorders.

8.2 Testing

Generalisations of the testing theory of [DH84] to probabilistic systems first appear in [Chr90] and [CSZ92], for generative processes without nondeterministic choice. The application of testing to the probabilistic processes we consider here stems from [WL92]. In [Seg96] a richer testing framework is proposed, for essentially the same class of processes, namely one in which multiple success actions ω_i for $i = 1, 2, \ldots$ exists, and the application of a test to a process yields not a set of real numbers, indicating success probabilities, but a set of tuples of real numbers, the i^{th} component in the tuple indicating the success probability of ω_i.

In [JHW94], a testing theory is proposed that associates a *reward*, a non-negative real number, to every success-state in a test process; in calculating the set of results of applying a test to a process, the probabilities of reaching a success-state are multiplied by the reward associated to that state. They allow non-probabilistic tests only, but apply these to arbitrary nondeterministic probabilistic processes, and provide a trace-like denotational characterisation of the resulting may-testing preorder. Denotational characterisations of the variant of our testing preorders in which only τ-free processes are allowed as test-processes appear in [JW95,JW99]. These characterisations are improved in [JW02], discussed below.

In [CCR$^+$03] a testing theory for nondeterministic probabilistic processes is developed in which, as in [MMSS96], all probabilistic choices are resolved first. A consequence of this is that the idempotence of internal choice (our axiom (**I1**)) must be sacrificed. Some papers distill preorders for probabilistic processes by means of *testing scenarios* in which the premise that a test is itself a process is given up. These include [LS91,KN98] and [SV03].

8.3 Simulations

Four different notions of simulation for probabilistic processes occur in the literature, each a generalisation of the well know concept of simulation for nondeterministic processes [Par81]. The most straightforward generalisation [JL91] defines a simulation as a relation $\mathcal{R}$ between states, satisfying, for all s, t, α, Θ,

$$\text{if } s\,\mathcal{R}\,t \text{ and } s \xrightarrow{\alpha} \Theta \text{ then there is a } \Delta' \text{ with } t \xrightarrow{\alpha} \Delta' \text{ and } \Theta\,\overline{\mathcal{R}}\,\Delta'.$$

This simulation induces a preorder that does not satisfy the law

$$a.(P \,{}_p\!\oplus\, Q) \sqsubseteq a.P \,\Box\, a.Q$$

which holds in probabilistic *may* testing semantics. The reason is that the process $a.P \,\Box\, a.Q$ can answer the initial a-move of $a.(P \,{}_p\!\oplus\, Q)$ by taking either the a-move to P, or the a-move to Q, but not by a probabilistic combination of the two. Such probabilistic combinations are allowed in the probabilistic simulation of [SL94], which induces a coarser preorder on processes, satisfying the above law. In our terminology it can be defined by changing the requirement above into

$$\text{if } s\,\mathcal{R}\,t \text{ and } s \xrightarrow{\alpha} \Theta \text{ then there is a } \Delta' \text{ with } \overline{t} \xrightarrow{\alpha} \Delta' \text{ and } \Theta\,\overline{\mathcal{R}}\,\Delta'.$$

A *weak* version of this probabilistic simulation, abstracting from the internal action τ, weakens this requirement into

$$\text{if } s\,\mathcal{R}\,t \text{ and } s \xrightarrow{\alpha} \Theta \text{ then there is a } \Delta' \text{ with } \overline{t} \stackrel{\hat{\alpha}}{\Longrightarrow} \Delta' \text{ and } \Theta\,\overline{\mathcal{R}}\,\Delta'.$$

Nevertheless, also this probabilistic simulation does not satisfy all the laws we have shown to hold for probabilistic may testing. In particular, it does not satisfy the law

$$a.(P \,{}_p\!\oplus\, Q) \sqsubseteq a.P \,{}_p\!\oplus\, a.Q.$$

Consider for instance the processes $R_1 = a.b.c.(d \,{}_{\frac{1}{2}}\!\oplus\, e)$ and $R_2 = a.(b.c.d \,{}_{\frac{1}{2}}\!\oplus\, b.c.e)$. The law (**Q1**) above, which holds for probabilistic *may* testing, would yield $R_1 \sqsubseteq R_2$. If we are to relate these processes via a probabilistic simulation à la [SL94], the state $c.(d \,{}_{\frac{1}{2}}\!\oplus\, e)$ of R_1, reachable after an a and a b-step, needs to be related to the *distribution* $(c.d \,{}_{\frac{1}{2}}\!\oplus\, c.e)$ of R_2, containing the two states $a.b$ and $a.c$. This relation cannot be obtained through lifting, as this would entail relating the single state $c.(d \,{}_{\frac{1}{2}}\!\oplus\, e)$ to each of the states $c.d$ and $c.e$. Such a relation would not be sound, because $c.(d \,{}_{\frac{1}{2}}\!\oplus\, e)$ is able to perform the sequence of actions ce half of the time, whereas the process $c.d$ cannot mimic this.

In [JW02], another notion of simulation is proposed, whose definition is too complicated to explain in a few sentences. They show for a class of probabilistic processes that do not contain τ-actions, that probabilistic *may* testing is captured exactly by their notion of simulation. Nevertheless, their notion of simulation makes strictly more identifications than ours. As an example, consider the processes $R_1 = a \,{}_{\frac{1}{2}}\!\oplus\, (b \,\Box\, c)$ and $R_3 = (a \,\Box\, b) \,{}_{\frac{1}{2}}\!\oplus\, (a \,\Box\, c)$ of Example 4.9, which also appear in Section 5 of [JW02]. There it is shown that $R_1 \sqsubseteq R_3$ holds in their semantics. However, in our framework we have $R_1 \not\sqsubseteq_{\text{pmay}} R_3$, as demonstrated in Example 4.9. The difference can only be explained by the circumstance that in [JW02] processes, and hence also tests, may not have internal actions. So this example shows that tests with internal moves can distinguish more processes than tests without internal moves, even when applied to processes that have no internal moves themselves.

Our notion of simulation first appears in [Seg95], although the preorder $\sqsubseteq_S$ of Definition 6.3 is new. Segala has no expressions that denote distributions and consequently is only interested in the restriction of the simulation preorder to states (*automata* in his framework). It turns out that for states s and t (which in our framework are expressions in the set S_p) we have $s \sqsubseteq_S t$ iff $s \vartriangleright_S \overline{t}$, so on their common domain of definition, the simulation preorder of [Seg95] agrees with ours.

This notion of simulation is strictly more discriminating than the simulation of [JW02], and strictly less discriminating than the ones from [SL94] and [JL91]. We conjecture that, for the class of processes considered in this paper, it captures probabilistic *may* testing exactly, i.e. that $P \sqsubseteq_S Q$ iff $P \sqsubseteq_{\text{pmay}} Q$. In [LSV03] it has been shown that the simulation preorder of [Seg95] coincides with the *congruence closure of inclusion of sets of trace distributions*, a preorder also defined in [Seg95]. In [Seg96] it has been shown that the latter notion coincides with the preorder generated by Segala's different form of probabilistic *may* testing, mentioned earlier, in which there are countably many different success actions, and the set of outcomes $\mathcal{O}$ consists of the countably-dimensional vectors over the unit interval [0,1]. To the best of our knowledge, the question whether this form of testing yields the same preorders as the notion of testing from [WL92] and our paper is, as of yet, unanswered.

References

[AB01] S. Andova & J.C.M. Baeten (2001): *Abstraction in probabilistic process algebra.* In Proceedings of the 7th International Conference on *Tools and Algorithms for the Construction and Analysis of Systems*, LNCS 2031, Springer, pp. 204–219.

[ABW06] S. Andova, J.C.M. Baeten & T.A.C. Willemse (2006): *A complete axiomatisation of branching bisimulation for probabilistic systems with an application in protocol verification.* In Proceedings of the 17th International Conference on *Concurrency Theory*, LNCS 4137, Springer, pp. 327–342.

[AJ94] S. Abramsky & A. Jung (1994): *Domain theory.* In *Handbook of Logic and Computer Science*, volume 3, Clarendon Press, pp. 1–168.

[AW06] S. Andova & T.A.C. Willemse (2006): *Branching bisimulation for probabilistic systems: Characteristics and decidability. Theoretical Computer Science* 356(3), pp. 325–355.

[BHR84] S.D. Brookes, C.A.R. Hoare & A.W. Roscoe (1984): *A theory of communicating sequential processes. Journal of the ACM* 31(3), pp. 560–599.

[BJ78] D. Bjørner & C.B. Jones, editors (1978): *The Vienna Development Method: The Meta-Language*, LNCS 61. Springer.

[BS01] E. Bandini & R. Segala (2001): *Axiomatizations for probabilistic bisimulation.* In Proceedings of the 28th International Colloquium on *Automata, Languages and Programming*, LNCS 2076, Springer, pp. 370–381.

[CCR+03] D. Cazorla, F. Cuartero, V.V. Ruiz, F.L. Pelayo & J.J. Pardo (2003): *Algebraic theory of probabilistic and nondeterministic processes.* *Journal of Logic and Algebraic Programming* 55(1-2), pp. 57–103.

[Chr90] I. Christoff (1990): *Testing equivalences and fully abstract models for probabilistic processes.* In Proceedings the 3rd International Conference on *Concurrency Theory*, LNCS 458, Springer, pp. 126–140.

[CSZ92] R. Cleaveland, S.A. Smolka & A.E. Zwarico (1992): *Testing preorders for probabilistic processes.* In Proceedings of the 19th International Colloquium on *Automata, Languages and Programming*, LNCS 623, Springer, pp. 708–719.

[Der70] C. Derman (1970): *Finite State Markovian Decision Processes.* Academic Press.

[DH84] R. De Nicola & M. Hennessy (1984): *Testing equivalences for processes.* *Theoretical Computer Science* 34, pp. 83–133.

[DP05] Y. Deng & C. Palamidessi (2005): *Axiomatizations for probabilistic finite-state behaviors.* In Proceedings of the 8th International Conference on *Foundations of Software Science and Computation Structures*, LNCS 3441, Springer, pp. 110–124.

[DPP05] Y. Deng, C. Palamidessi & J. Pang (2005): *Compositional reasoning for probabilistic finite-state behaviors.* In *Processes, Terms and Cycles: Steps on the Road to Infinity, Essays Dedicated to Jan Willem Klop, on the Occasion of His 60th Birthday*, LNCS 3838, Springer, pp. 309–337.

[ER64] C.C. Elgot & A. Robinson (1964): *Random-access stored-program machines, an approach to programming languages.* *Journal of the ACM* 11(4), pp. 365–399.

[GJS90] A. Giacalone, C.-C. Jou & S.A. Smolka (1990): *Algebraic reasoning for probabilistic concurrent systems.* In Proceedings of IFIP TC 2 Working Conference on *Programming Concepts and Methods*, pp. 443–458.

[GSST90] R.J. van Glabbeek, S.A. Smolka, B. Steffen & C.M.N. Tofts (1990): *Reactive, generative, and stratified models of probabilistic processes.* In Proceedings of the 5th Annual IEEE Symposium on *Logic in Computer Science*, Computer Society Press, pp. 130–141.

[GW96] R.J. van Glabbeek & W.P. Weijland (1996): *Branching time and abstraction in bisimulation semantics.* *Journal of the ACM* 43(3), pp. 555–600.

[Hen82] M. Hennessy (1982): *Powerdomains and nondeterministic recursive definitions.* In Proceedings of the 5th International *Symposium on Programming*, LNCS 137, Springer, pp. 178–193.

[Hen88] M. Hennessy (1988): *An Algebraic Theory of Processes.* MIT Press.

[HJ90] H. Hansson & B. Jonsson (1990): *A calculus for communicating systems with time and probabilities.* In Proceedings of the *Real-Time Systems Symposium* (RTSS '90), Computer Society Press, pp. 278–287.

[Hoa85] C.A.R. Hoare (1985): *Communicating Sequential Processes.* Prentice-Hall.

[HSM97] Jifeng He, K. Seidel & A.K. McIver (1997): *Probabilistic models for the guarded command language.* *Science of Computer Programming* 28, pp. 171–192.

[JHW94] B. Jonsson, C. Ho-Stuart & Wang Yi (1994): *Testing and refinement for nondeterministic and probabilistic processes.* In Proceedings of the 3rd International Symposium on *Formal Techniques in Real-Time and Fault-Tolerant Systems*, LNCS 863, Springer, pp. 418–430.

[JL91] B. Jonsson & K.G. Larsen (1991): *Specification and refinement of probabilistic processes.* In Proceedings of the 6th Annual IEEE Symposium on *Logic in Computer Science*, Computer Society Press, pp. 266–277.

[JP89] C. Jones & G. Plotkin (1989): *A probabilistic powerdomain of evaluations.* In Proceedings of the 4th Annual IEEE Symposium on *Logic in Computer Science*, Computer Society Press, pp. 186–195.

[JW95] B. Jonsson & Wang Yi (1995): *Compositional testing preorders for probabilistic processes.* In Proceedings of the 10th Annual IEEE Symposium on *Logic in Computer Science*, Computer Society Press, pp. 431–441.

[JW99] B. Jonsson & Wang Yi (1999): *Fully abstract characterization of probabilistic may testing.* In Proceedings of the 5th International AMAST Workshop on *Formal Methods for Real-Time and Probabilistic Systems*, LNCS 1601, Springer, pp. 1–18.

[JW02] B. Jonsson & Wang Yi (2002): *Testing preorders for probabilistic processes can be characterized by simulations.* *Theoretical Computer Science* 282(1), pp. 33–51.

[JWL01] B. Jonsson, Wang Yi & K.G. Larsen (2001): *Probabilistic extensions of process algebras.* In *Handbook of Process Algebra*, chapter 11, Elsevier, pp. 685–710.

[KN98] M.Z. Kwiatkowska & G. Norman (1998): *A testing equivalence for reactive probabilistic processes.* *Electronic Notes in Theoretical Computer Science*, 16(2).

[Lan63] P.J. Landin (1963): *The mechanical evaluation of expressions.* *Computer Journal* 6(4), pp. 308–320.

[Low93] G. Lowe (1993): *Representing nondeterminism and probabilistic behaviour in reactive processes.* Technical Report TR-11-93, Computing laboratory, Oxford University.

[Low95] G. Lowe (1995): *Probabilistic and prioritized models of timed CSP.* *Theoretical Computer Science* 138, pp. 315–352.

[LS91] K.G. Larsen & A. Skou (1991): *Bisimulation through probabilistic testing.* *Information and Computation* 94(1), pp. 1–28.

[LS92] K.G. Larsen & A. Skou (1992): *Compositional verification of probabilistic processes.* In Proceedings of the 3rd International Conference on *Concurrency Theory*, LNCS 630, Springer, pp. 456–471.

[LSV03] N. Lynch, R. Segala & F.W. Vaandrager (2003): *Compositionality for probabilistic automata.* In Proceedings of the 14th International Conference on *Concurrency Theory*, LNCS 2761, Springer, pp. 204–222.

[Luc71] P. Lucas (1971): *Formal definition of programming languages and systems.* In *IFIP Congress*, volume 1, pp. 291–297.

[Mil89] R. Milner (1989): *Communication and Concurrency.* Prentice-Hall.

[Mis00] M.W. Mislove (2000): *Nondeterminism and probabilistic choice: Obeying the laws.* In Proc. of *CONCUR'00*, LNCS 1877, Springer, pp. 350–364.

[MM01] A.K. McIver & C.C. Morgan (2001): *Partial correctness for probabilistic programs.* Theoretical Computer Science 266(1–2), pp. 513–41.

[MMS96] C.C. Morgan, A.K. McIver & K. Seidel (1996): *Probabilistic predicate transformers.* ACM Transactions on Programming Languages and Systems 18(3), pp. 325–353.

[MMSS96] C.C. Morgan, A.K. McIver, K. Seidel & J.W. Sanders (1996): *Refinement-oriented probability for CSP.* Formal Aspects of Computing 8(6), pp. 617–47.

[MOW04] M.W. Mislove, J. Ouaknine & J. Worrell (2004): *Axioms for probability and nondeterminism.* Electronic Notes in Theoretical Computer Science 96, pp. 7–28.

[OH86] E.-R. Olderog & C.A.R. Hoare (1986): *Specification-oriented semantics for communicating processes.* Acta Informatica 23, pp. 9–66.

[Par81] D.M.R. Park (1981): *Concurrency and automata on infinite sequences.* In Proceedings of 5th GI Conference, LNCS 104, Springer, pp. 167–183.

[Plo81] G.D. Plotkin (1981): *A Structural Approach to Operational Semantics.* Technical Report DAIMI FN-19, University of Aarhus.

[Plo04] G.D. Plotkin (2004): *A Structural Approach to Operational Semantics.* Journal of Logic and Algebraic Programming 60-61, pp. 17–139.

[PLS00] A. Philippou, I. Lee & O. Sokolsky (2000): *Weak bisimulation for probabilistic systems.* In Proceedings of the 11th International Conference on Concurrency Theory, LNCS 1877, Springer, pp. 334–349.

[Rab63] M.O. Rabin (1963): *Probabilistic automata.* Information and Control 6, pp. 230–245.

[Seg95] R. Segala (1995): *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, MIT.

[Seg96] R. Segala (1996): *Testing probabilistic automata.* In Proceedings of the 7th International Conference on Concurrency Theory, LNCS 1119, Springer, pp. 299–314.

[SL94] R. Segala & N.A. Lynch (1994): *Probabilistic simulations for probabilistic processes.* In Proceedings of the 5th International Conference on Concurrency Theory, LNCS 836, Springer, pp. 481–496.

[SV03] M.I.A. Stoelinga & F.W. Vaandrager (2003): *A testing scenario for probabilistic automata.* In Proceedings of the 30th International Colloquium on Automata, Languages and Programming, LNCS 2719, Springer, pp. 407–18.

[TKP05] R. Tix, K. Keimel & G.D. Plotkin (2005): *Semantic domains for combining probability and non-determinism.* Electronic Notes in Theoretical Computer Science 129, pp. 1–104.

[Var85] M.Y. Vardi (1985): *Automatic verification of probabilistic concurrent finite-state programs.* In Proceedings 26th Annual Symposium on Foundations of Computer Science, pp. 327–338.

[VW06] D. Varacca & G. Winskel (2006): *Distributing probability over non-determinism.* Mathematical Structures in Computer Science 16(1), pp. 87–113.

[WL92] Wang Yi & K.G. Larsen (1992): *Testing probabilistic and nondeterministic processes.* In Proceedings of the IFIP TC6/WG6.1 Twelfth International Symposium on Protocol Specification, Testing and Verification, IFIP Transactions C-8, North-Holland, pp. 47–61.