Algebraic Theory of Probabilistic Processes ⋆

Manuel Núñez

Dept. Sistemas Informáticos y Programación.
Universidad Complutense de Madrid, E-28040 Madrid. Spain.

Abstract

In this paper we extend de Nicola & Hennessy's testing theory to deal with probabilities. We say that two processes are testing equivalent if the probabilities with which they pass any test are equal. We present three alternative semantic *views* of our testing equivalence. First, we introduce adequate extensions of acceptance sets (inducing an operational characterization) and acceptance trees (inducing a denotational semantics). We also present a sound and complete axiomatization of our testing equivalence. So, this paper represents a complete study of the adaptation of the classical testing theory for probabilistic processes.

Key words: Probabilistic process algebras, probabilistic testing semantics.

1 Introduction

Process algebras have been show to be an adequate mechanism to formally describe and analyze concurrent systems (an extensive presentation of the research in the field can be found in [4]). The process algebra literature includes numerous semantic models. These semantics are used to describe the behavior of processes as well as to define relations on them. Testing semantics [29,19] represents one of these semantic frameworks. Two processes are *testing equivalent* if they have the same *responses* for any *test*. Depending on how these responses are analyzed, several testing semantics can be defined: may, must, fair, etc. In [29,19], three alternative *views* (operational, denotational, and axiomatic) of the may, must, and may-must testing semantics are given.

During the last decade researchers in process algebras have tried to close the gap between formal models and real systems. In particular, features which

⋆ Research supported in part by the CICYT project TIC2000-0701-C02-01.
Email address: mn@sip.ucm.es (Manuel Núñez).

were abstracted before have been now introduced. This is the case of probabilistic information (see [22] for an overview on the different semantics for probabilistic processes). In particular, several probabilistic testing semantics have been defined (e.g. [8,37,21,32,31,34,28,24,9,33]). See [9,22] for a comparison between some of these proposals.

However, previous work on probabilistic testing semantics considered only *partial* views of the testing framework. This paper [1] constitutes a complete study of the classical testing theory for probabilistic processes (actually, we have borrowed the title of the paper from [19]). First we will define a testing equivalence: Two processes are testing equivalent if they pass any test with the same probability. Then, we present three different views of this equivalence. These alternative characterizations follow the classical pattern. In order to define an operational characterization we define a suitable probabilistic extension of acceptance sets. We have a fully abstract denotational semantics based on (probabilistic) acceptance trees. Finally, we present a sound and complete axiomatization.

As we will see along the paper, the adaptation of the non-probabilistic framework to deal with probabilities is far from trivial. This is related to the fact that it will be more expressive. For instance, it is able to capture some kind of *fairness*. Consider the process $P = (a; Nil) \oplus_p P$. In the previous process, the operator $\oplus_p$ denotes a pure probabilistic choice, that is, $P_1 \oplus_p P_2$ behaves with probability p as P_1 and with probability $1-p$ as P_2. If we *forget* probabilities, P is must equivalent to *divergence* because the lack of quantification could produce that the left hand side is never taken. Nevertheless, in a probabilistic setting we expect that if the environment offers a then P performs it with probability 1, and so, P should be (probabilistic) testing equivalent to $a; Nil$. In particular, this example illustrates why our axiomatization cannot be a simple adaptation of that for non-probabilistic processes. We will need a rule expressing that this kind of recursively defined processes are equivalent to a finite one (such as $a; Nil$ in the previous case).

In order to keep compatibility with the language [2] used in [19], we consider a probabilistic process algebra, that we call PPA, featuring two (probabilistic) choice operators: external and internal. Sometimes it has been argued that the external choice operator should not be extended with a probability. In fact, there are some proposals including a probabilistic internal choice while the external choice operator remains non-probabilistic. On the contrary, we find it very useful to have two probabilistic choice operators. In particular, in order to have the same expressive power as a natural probabilistic extension

[1] Some parts of this paper have previously appeared in [32,30].

[2] This language appears in [14]. Its principal feature is that the CCS choice operator is replaced by two choice operators similar to those in CSP [20].

of CCS,[3] we need to include probabilistic information in both operators. For example, with a non-probabilistic external choice we cannot simulate such a simple process as $P = a +_p b$, which relates a and b probabilistically but does not represent a pure probabilistic choice. Intuitively, if the environment offers a and b then a will be performed with probability p while b will be performed with probability $1 - p$; if the environment offers only a (resp. b) then a is performed with probability 1. This last point shows the difference between a probabilistic external choice and a probabilistic internal choice.

One could argue why we do not just work with a probabilistic version of CCS. This is because by working with two choice operators we get that the testing equivalence is in fact a congruence. As usual, working with a CCS-like operator we need to consider the largest congruence contained in the testing equivalence. Finally, there are some behaviors which can be specified more precisely by using a probabilistic external choice operator. We will illustrate this by means of a simple example. Suppose we specify the behavior of a library where two users can request books. If only one user requests a book then the book is given to him; if both users request the same book then the library must give *priority* to one of the users. Besides, the system must be somehow *fair*, avoiding the possibility that if the two users request the same book, this book is always given to the same person. A simplified version of the system can be specified as $P = a; P +_{\frac{1}{4}} b; P$, indicating that if both users request the same book, it will be given with probability $\frac{1}{4}$ to the user a and with probability $\frac{3}{4}$ to the user b. If only one user request the book then it is given to him/her with probability 1. Note that if we use a probabilistic internal choice then there is no guarantee that if only one user requests the book, it is given to him, while if we use a non-probabilistic external choice then we cannot specify the notion of *priority*. An interesting alternative to the inclusion of a probabilistic external choice appears in [27] where this operator is considered as derived from the probabilistic internal choice and the *priority* operators. Nevertheless it is also necessary to include some kind of *probability* (the *extremal* value 1) in the external choice operator.

We assume that the reader has some familiarity with testing theory. Although the paper is self-contained with respect to probabilistic processes, this knowledge will be useful when our model is compared with the classical testing theory for non-probabilistic processes. The rest of the paper is organized as follows. In Section 2 we introduce our language and our testing semantics. In Section 3 we present an alternative characterization of our testing equivalence. This characterization is based on a probabilistic extension of acceptance sets. In Section 4 we present a fully abstract denotational semantics. This semantics is based on a probabilistic extension of acceptance trees. In Section 5 we

[3] Most probabilistic models are based either on CCS or in labeled transition systems (which can be easily interpreted as CCS processes).

present a sound and complete axiomatization of our testing semantics. In Section 6 we show how our language can be extended with a notion of parallel composition. Besides, we discuss the difficulties involving the introduction of a hiding operator. Finally, in Section 7 we present our conclusions and some lines for future work. The Appendix of the paper contains the proofs of the main results.

2 The language PPA: Operational and Testing Semantics

In this section we present the probabilistic process algebra PPA. As we said in the introduction of this paper, PPA is based on the language used in [19]. The main difference with respect to that language is that we have labeled both choice operators (external and internal) with a probability. In this paper, *extremal* values of probabilities (i.e. 0 and 1) will not be considered because, in order to deal with priorities, a very complex extension of the model is needed (see [10] for a presentation of different priority models).

Definition 2.1 Given a denumerable set of actions Act and a set of process identifiers Id, the set of PPA *processes* is defined by the following BNF-expression: $P ::= \mathit{Nil} \mid \Omega \mid X \mid a;P \mid P \oplus_p P \mid P +_p P \mid \mathit{rec}X.P$, where $p \in (0,1)$, $a \in Act$, and $X \in Id$. □

From now on, except if noted, we only consider closed processes (i.e. processes without free occurrences of variables) and we usually omit trailing occurrences of Nil. In this language Nil is a deadlocked process, Ω is a divergent process, [4] $a;P$ denotes the action a prefixing the process P, $P \oplus_p Q$ denotes an internal choice between P and Q with associated probability p, $P +_p Q$ is an external choice between P and Q with associated probability p, and finally $\mathit{rec}X.P$ is used to define recursive processes. Even though both choice operators are probabilistic, their intuitive meaning is completely different. This will be more clear once we define our testing semantics.

Next we give a syntactic definition for the *stability* of a process. It expresses that a process has not unguarded internal choices, or equivalently that a process will not be able to (immediately) perform an internal transition. We also define a function *live* computing whether a stable process is operationally equivalent to Nil.

Definition 2.2 We define the predicate $\mathtt{stable}(P)$ over PPA processes as:

[4] Actually, such a process can be derived from the rest of the language. For example, the process $\mathit{rec}X.X$ generates the same transitions as Ω. We keep this syntactic expression of divergence for keeping compatibility with the classical framework.

$$(PRE)\frac{}{a;P \xrightarrow{a}_1 P} \qquad (INT1)\frac{}{P \oplus_p Q >\!\!\longrightarrow_p P} \qquad (INT2)\frac{}{P \oplus_p Q >\!\!\longrightarrow_{1-p} Q}$$

$$(EXT1)\frac{P >\!\!\longrightarrow_q P' \ \wedge\ \texttt{stable}(Q)}{P +_p Q >\!\!\longrightarrow_q P' +_p Q} \qquad (EXT2)\frac{Q >\!\!\longrightarrow_q Q' \ \wedge\ \texttt{stable}(P)}{P +_p Q >\!\!\longrightarrow_q P +_p Q'}$$

$$(EXT3)\frac{P >\!\!\longrightarrow_{q_1} P' \ \wedge\ Q >\!\!\longrightarrow_{q_2} Q'}{P +_p Q >\!\!\longrightarrow_{q_1 \cdot q_2} P' +_p Q'}$$

$$(EXT4)\frac{P \xrightarrow{a}_q P' \ \wedge\ \texttt{stable}(Q)}{P +_p Q \xrightarrow{a}_{p \cdot \hat{q}} P'} \qquad (EXT5)\frac{Q \xrightarrow{a}_q Q' \ \wedge\ \texttt{stable}(P)}{P +_p Q \xrightarrow{a}_{(1-p) \cdot \hat{q}} Q'}$$

$$(REC)\frac{}{rec\ X.P >\!\!\longrightarrow_1 P\{rec\ X.P/X\}} \qquad (DIV)\frac{}{\Omega >\!\!\longrightarrow_1 \Omega}$$

where $\hat{q} = \frac{q}{p \cdot \texttt{live}(P) + (1-p) \cdot \texttt{live}(Q)}$.

Fig. 1. Operational Semantics of PPA.

- $\texttt{stable}(Nil) = \texttt{stable}(a;P) = \texttt{true}$
- $\texttt{stable}(\Omega) = \texttt{stable}(X) = \texttt{stable}(P_1 \oplus_p P_2) = \texttt{stable}(recX.P) = \texttt{false}$
- $\texttt{stable}(P_1 +_p P_2) = \texttt{stable}(P_1) \ \wedge\ \texttt{stable}(P_2)$

We define the function $\texttt{live}(P)$ over PPA processes as:

- $\texttt{live}(Nil) = 0$
- $\texttt{live}(a;P) = 1$
- $\texttt{live}(P_1 +_p P_2) = \max(\texttt{live}(P_1), \texttt{live}(P_2))$

□

Even though the function $\texttt{live}(\_)$ is not defined for unstable processes, this fact does not represent a problem as we will apply it only to stable processes. The set of rules defining the operational semantics is given in Figure 1. There are two types of transitions. The intuitive meaning of an *external* transition $P \xrightarrow{a}_p Q$ is that if the environment offers all the actions in Act, then the probability with which P performs a and then behaves as Q is equal to p. The meaning of an internal transition $P >\!\!\longrightarrow_p Q$ is that the process P evolves into Q with probability p, without interaction with the environment. In order to avoid the problem of deriving the same transition in different ways, we use multisets of transitions. For example, consider $P = a +_{\frac{1}{2}} a$. If we were not careful, we would have the transition $P \xrightarrow{a}_{\frac{1}{2}} Nil$ only once, while we should have this transition twice. This problem is similar for the $\oplus_p$ operator. So, in our model, if a transition can be derived in several ways, we consider that each derivation generates a different instance. In particular, when we define the testing semantics we will consider multisets of computations as well. Other approaches to solve this problem are to index transitions (e.g. [15]), to increase the number of rules (e.g. [25]), to define a transition probability function (e.g. [35]), or to add the probabilities associated with the same transition (e.g. [37]).

The rule for prefix (PRE) simply indicates that the action is performed with probability 1. The rules for internal choice $(INT1-2)$ indicate that with

probability p the left hand side process will be chosen while the right hand side process will be chosen with probability $1-p$. The rule (DIV) says that divergence can only evolve by performing an internal transition (with probability 1). The rule for recursion (REC) indicates that a recursive process must be *unfolded* before it can perform any transitions. This is the way recursion is defined in [19], but note that this is not the usual definition for CCS. Rules $(EXT1-3)$ indicate that whenever any of the arguments of an external choice can evolve via an internal transition, these transitions are performed until both arguments become *stable*. Note that this definition does not mean that the external choice operator is static. Rules $(EXT4-5)$ are applied when both processes are stable and (at least) one of them may perform some observable action. The value $\hat{q}$ is obtained by normalizing the probability q of performing this external transition where we take into account whether one or both processes can perform external transitions. For example, consider $P = (a; Nil) +_p Nil$. We have $P \stackrel{a}{\longrightarrow}_1 Nil$, while if we would not use this *normalization* we would obtain $P \stackrel{a}{\longrightarrow}_p Nil$.

As a consequence of this definition of operational semantics we have that internal and external transitions are not mixed, and then we have the following result.

Lemma 2.3 Let P be a process. If there exist p, P' such that $P \rightarrowtail_p P'$ then there do not exist q, a, P'' such that $P \stackrel{a}{\longrightarrow}_q P''$. Equivalently, if there exist p, a, P' such that $P \stackrel{a}{\longrightarrow}_p P'$ then there do not exist q, P'' such that $P \rightarrowtail_q P''$. □

We can extend the relation induced by internal transitions $\rightarrowtail_p$, to a relation $P \rightarrowtail_p^* P'$ indicating that the process P can evolve to the process P' after executing a sequence of internal transitions such that the product of the probabilities associated with them is equal to p. In our definition, P' must be unable to perform more internal transitions, that is, P' must be stable.

Definition 2.4 Let P, P' be processes. We say that P evolves to P' by means of a *generalized internal transition* with probability p, denoted by $P \rightarrowtail_p^* P'$, if either $\texttt{stable}(P) \wedge P = P' \wedge p = 1$ or there exist $q_1, \ldots, q_n, Q_1, \ldots Q_{n-1}$ such that $P \rightarrowtail_{q_1} Q_1 \rightarrowtail_{q_2} \cdots Q_{n-1} \rightarrowtail_{q_n} P' \wedge \texttt{stable}(P') \wedge p = \Pi q_i$. □

As for the previously introduced transitions, the new relation induces a multiset of transitions instead of just a set.

We finish the presentation of the language by generalizing the choice operators to deal with an arbitrary (finite) number of arguments. For the generalized external choice we will use a restricted form, in which all the arguments are prefixed by different actions. These operators will be used, in particular, when we define the notion of normal form.

Definition 2.5 Let $P_1, P_2, \ldots, P_n$ be processes, and $a_1, a_2, \ldots, a_n \in Act$ pairwise different actions. We inductively define the *generalized external choice* as

1. $\sum_{i=1}^{0}[p_i]\, a_i; P_i = Nil$ 2. $\sum_{i=1}^{1}[1]\, a_1; P_1 = a_1; P_1$

3. $\sum_{i=1}^{n}[p_i]\, a_i; P_i = (a_1; P_1) +_{p_1} (\sum_{i=1}^{n-1}[\frac{p_{i+1}}{1-p_1}]\, a_{i+1}; P_{i+1})$

where $p_1, p_2, \ldots, p_n > 0$ are such that $\sum p_i = 1$.

We inductively define the *generalized internal choice* by

1. $\bigoplus_{i=1}^{0}[p_i]\, P_i = \Omega$ 2. $\bigoplus_{i=1}^{1}[1]\, P_1 = P_1$

3. $\bigoplus_{i=1}^{n}[p_i]\, P_i = \bigoplus_{i=1}^{n}[\frac{p_i}{p}]\, P_i \oplus_p \Omega$ [if $p = \sum p_i < 1 \ \wedge \ n > 0$]

4. $\bigoplus_{i=1}^{n}[p_i]\, P_i = P_1 \oplus_{p_1} (\bigoplus_{i=1}^{n-1}[\frac{p_{i+1}}{1-p_1}]\, P_{i+1})$ [if $\sum p_i = 1 \ \wedge \ n > 1$]

where $p_1, p_2, \ldots, p_n > 0$ are such that $\sum p_i \leq 1$. □

Regarding the definition of the generalized external choice operator, setting the empty summation to *Nil* is consistent with the properties fulfilled by the external choice. In particular, *Nil* is the identity element of $+_p$. Let us remark that the sum of the probabilities associated with a generalized internal choice may be less than 1. The difference between 1 and this value indicates the probability of divergence. The idea is that the remaining probability can be considered as the degree of *undefinability* of the process. As we will show, the less *defined* process of the semantic domain will be associated with Ω. In this case, the third clause is firstly applied so that the sum of the probabilities associated with the remaining generalized internal choice is equal to 1 (afterwards the second or the fourth clauses will be used).

Having defined an operational semantics for our language, we now endow it with a testing semantics. We first give the notion of *test*. Following the classical testing semantics, tests are just processes but extending the set of actions with a new action $\omega \notin Act$, indicating the *successful* termination of the test. We denote by $\mathcal{T}est$ the set of tests. So, the operational semantics of tests is the same as the one for processes, just considering ω as a usual action.

As in the non-probabilistic case, the interaction between a process and a test is modelled by a parallel composition, considering as synchronization set the full set of actions Act (note that ω does not belong to the synchronization set). We denote by $P \mid T$ the parallel composition of the process P and the test T. The operational rules of this composition are given in Figure 2. These rules remind the pattern of the operational semantics for the external choice

$$\frac{P \rightarrowtail_p P' \wedge \texttt{stable}(T)}{P \mid T \rightarrowtail_p P' \mid T} \qquad \frac{T \rightarrowtail_p T' \wedge \texttt{stable}(P)}{P \mid T \rightarrowtail_p P \mid T'} \qquad \frac{P \rightarrowtail_p P' \wedge T \rightarrowtail_q T'}{P \mid T \rightarrowtail_{p \cdot q} P' \mid T'}$$

$$\frac{P \xrightarrow{a}_p P' \ \wedge \ T \xrightarrow{a}_q T'}{P \mid T \xrightarrow{a}_{r_1} P' \mid T'} \qquad \frac{T \xrightarrow{\omega}_p T' \ \wedge \ \texttt{stable}(P)}{P \mid T \xrightarrow{\omega}_{r_2} Nil}$$

where $r_1 = \frac{p \cdot q}{\mu(P,T)}$ and $r_2 = \frac{p}{\mu(P,T)}$

Fig. 2. Rules for the composition of processes and tests.

operator. We have two groups of rules: One of them dealing with internal transitions, and another one dealing with observable ones. The first three rules define how internal transitions are performed. The fourth rule indicates synchronization in actions belonging to Act. The fifth rule indicates execution of the successful action ω by the test. In this last case we have that the testing procedure (successfully) finishes. By using this mechanism, we avoid the derivation of useless transitions once an ω action is performed. Regarding probabilities, we use a *normalization factor*. The function $\mu(P, T)$ computes the probability associated with the transitions that $P \mid T$ may perform. In the case of synchronization, we multiply the corresponding probabilities; in the case of the ω action we simply take the probability of executing that action. This function is formally defined as:

$$\mu(P,T) = \sum \{\!|\, p \cdot q \mid \exists\, P', T', a : P \xrightarrow{a}_p P' \wedge T \xrightarrow{a}_q T' \,|\!\} + \sum \{\!|\, p \mid \exists\, T' : T \xrightarrow{\omega}_p T' \,|\!\}$$

The next step in defining our testing semantics is to introduce the notion of *computation*. In particular, we distinguish *successful computations*, which are those computations such that the test performs ω. Because computations have a probability associated with them, by adding the probabilities corresponding to the successful ones we obtain the probability with which a process passes a test.

Definition 2.6 Let P be a process and T be a test. A *computation* C from $P \mid T$ is a maximal [5] sequence of transitions of the form

$$C = P \mid T \longmapsto_{p_1} P_1 \mid T_1 \longmapsto_{p_2} \cdots \longmapsto_{p_{n-1}} P_{n-1} \mid T_{n-1} \longmapsto_{p_n} R$$

where $\longmapsto_p$ denotes either an internal transition $\rightarrowtail_p$ or a transition $\xrightarrow{\alpha}_q$, with $\alpha \in Act \cup \{\omega\}$. If the last transition of a computation is of the form $P_{n-1} \mid T_{n-1} \xrightarrow{\omega}_{p_n} Nil$ then we say that the computation is *successful* and that the length of C is equal to n, denoted by $\texttt{length}(S) = n$. The *set* of *successful computations* from $P \mid T$ is denoted by $Su(P, T)$.

[5] By maximal we mean that the sequence is either infinite or finite but deadlocked.

The *probability of a computation* C, denoted by $Pr(C)$, is inductively defined as:

$$Pr(Nil) = Pr(\epsilon) = 1$$
$$Pr(P \mid T \stackrel{*}{\longmapsto}_p C) = p \cdot Pr(C)$$

where ϵ denotes an empty sequence (i.e. the composition is deadlocked and the testing procedure has not successfully finished). It can be shown that Pr is a probability distribution on computations.

We define the function $pass(P, T)$, which computes the *probability with which* the process P *passes* the test T, as $pass(P\ ,\ T) = \sum_{C \in Su(P,T)} Pr(C)$. □

Let us remark that the role played by tests of the form $a +_p (\tau; \omega)$ in other models (e.g. [9]) is played in our model by tests of the form $a +_p \omega$, which are not trivially passed within our framework. For example, the process $P = a$ *passes* the test above with a probability $1 - p$. The following result gives an alternative definition of the probability with which a process passes a test. By considering *limits*, we can restrict ourselves to finite computations in the following way (the proof is trivial).

Lemma 2.7 Let P be a process and T be a test. We have $pass(P\ ,\ T) = \lim_{n\to\infty} \sum \{\!|\ Pr(S) \mid S \in Su(P,T)\ \wedge\ \mathtt{length}(S) < n\ |\!\}$. □

Definition 2.8 Let $P, Q \in$ PPA. We say that P and Q are *testing equivalent*, denoted by $P \approx Q$, if for any $T \in \mathcal{T}est$ we have $pass(P, T) = pass(Q, T)$. □

Example 2.9 Let us show some simple examples to illustrate our testing equivalence. Consider the processes $P_1 = a +_p b$ and $P_2 = a \oplus_p b$, and the test $T_1 = a; \omega$. We have $pass(P_1\ ,\ T_1) = 1 \neq p = pass(P_2\ ,\ T_1)$. This example shows that external choice is not a simple probabilistic choice: If one of the actions is not *offered* by the environment then the whole probability goes to the other one. The processes $P_3 = (a; Q) +_p (a; Q')$, $P_4 = a; (Q \oplus_p Q')$, and $P_5 = (a; Q) \oplus_p (a; Q')$ are testing equivalent. This is similar to the situation in the classical framework where non-determinism within an external choice can be transformed into an internal choice. Our testing semantics distinguishes between Nil (deadlock) and Ω (divergence). For instance, we have $pass(Nil\ ,\ \omega) = 1 \neq 0 = pass(\Omega\ ,\ \omega)$. Consider the processes $P_6 = a$ and $P_7 = recX.a \oplus_p X$. As we discussed in the introduction of this paper, we have $P_6 \approx P_7$. Let us remind that if we *delete* the probability appearing in P_7 and we consider the non-probabilistic must testing semantics we have that P_7 is equivalent to Ω. □

The next result shows that recursive tests do no add discriminatory power. Even though the proof is not difficult, we include it in the Appendix of this paper because we think that it can be useful for other probabilistic models.

Actually, a similar result/proof has been already used in frameworks quite different to the one presented in this paper (e.g. [16,26,7,6]). Let us remark that by finite test we mean a test with no occurrences of recursion.

Lemma 2.10 $P \approx Q$ iff $pass(P\ ,\ T) = pass(Q\ ,\ T)$ for any finite test T. □

3 Alternative Characterization: Probabilistic Acceptance Sets

In this section we present an alternative characterization of our testing semantics. To this end, we will use a probabilistic extension of *acceptance sets* [29,19]. These sets are used in the non-probabilistic setting for giving an alternative characterization of the *must* semantics. In short, acceptance sets are defined as those sets of actions (called *states*) that are available after a sequence of (visible) actions is performed. In the following examples we will sometimes use the non-probabilistic counterparts of our choice operators: $\oplus$ and $+$.

Example 3.1 Consider the non-probabilistic process $P = a \oplus ((b;d) + c)$. After performing the empty sequence of (visible) actions (that is, $P \succ\!\!\rightarrow^*$) we have that the only stable processes that P may reach are $P \succ\!\!\rightarrow^* a; Nil$ and $P \succ\!\!\rightarrow^* (b;d) + c$.[6] So, the reachable states after the empty sequence (we usually call them *immediately* reachable states) are the sets of actions $\{a\}$ and $\{b, c\}$. Similarly, after performing the sequence $\langle b \rangle$ we have that the only reachable state is $\{d\}$. □

The main changes with respect to the non-probabilistic framework are the following:

- States are not sets of actions but sets of pairs (action, probability) (see the forthcoming Example 3.2).
- In the non-probabilistic framework, acceptance sets are defined as the reachable states after a sequence of actions is performed (denoted by $\mathcal{A}(P, s)$, where s is a sequence of actions). In the probabilistic case, sequences of actions are not enough for determining uniquely the subsequent continuations. We have to consider sequences of (state, action) pairs (see Example 3.3).
- The notion of equivalence must be modified by taking into account the probabilistic information that appears in acceptance sets. Moreover, in contrast with the non-probabilistic case no notion of *closure* is necessary (see forthcoming Example 3.4).[7]

[6] The definition of $\succ\!\!\rightarrow^*$ is similar to that of $\succ\!\!\rightarrow_p^*$ but omitting probabilities.

[7] Even though closures (under union or convex) are not explicitly used when defining the equivalence between acceptance sets in [19], they are *hidden* under the definition of $\subset\!\subset$.

The next example illustrates the first of the changes.

Example 3.2 Consider the processes $P = a +_{\frac{1}{2}} b$ and $P' = a +_{\frac{1}{3}} b$. Both processes have as immediately reachable state one containing the actions a and b. If we do not introduce probabilistic information in the states, then both processes would be equivalent in the semantics based on acceptance sets. However, they are not equivalent with respect to the testing semantics because, for example, the test $T = (a;\omega) +_{\frac{1}{3}} (b; Nil)$ distinguishes them. Therefore, the (unique) immediately reachable state of P will be the (probabilistic) state $\{(a, \frac{1}{2}), (b, \frac{1}{2})\}$, while the one for P' will be $\{(a, \frac{1}{3}), (b, \frac{2}{3})\}$. □

In the non-probabilistic setting, states can be characterized just by using sequences of actions because the *continuations* of a process after an action is performed cannot be distinguished, and thus they must be joined into a common continuation. In other words, once the process performs an action, there is no possibility for knowing from which state this action was performed. This is not the case, however, in the probabilistic setting. In this case, we can distinguish via tests from which state the action was performed, so that continuations cannot be joined.

Example 3.3 Let P be the non-probabilistic process $a; d \oplus ((a; b) + c)$. We have that P is equivalent to $P' = a; (d \oplus b) \oplus ((a; (d \oplus b)) + c)$ with respect to the equivalence induced by acceptance sets. That is, continuations after a has been performed are joined (and so, they are repeated). But this does not happen in the probabilistic case. Let us consider the process $P = a; d \oplus_{\frac{1}{2}} ((a; b) +_{\frac{1}{2}} c)$, and suppose that there exist R_1, R_2 such that P is testing equivalent to $P' = a; R_1 \oplus_{\frac{1}{2}} ((a; R_1) +_{\frac{1}{2}} c; R_2)$. If we consider $T = (a; b; \omega) +_{\frac{1}{3}} c$ then on the one hand we obtain $pass(P\,,\,T) = \frac{1}{6}$, while on the other hand $pass(P'\,,\,T) = \frac{1}{2} \cdot pass(R_1\,,\,(b;\omega)) + \frac{1}{2} \cdot \frac{1}{3} \cdot pass(R_1\,,\,(b;\omega))$. By assumption, $P \approx P'$, we should have $pass(P'\,,\,T) = \frac{1}{6}$, which implies $pass(R_1\,,\,(b;\omega)) = \frac{1}{4}$.

Consider now the test $T' = a; b; \omega$. We obtain $pass(P\,,\,T') = \frac{1}{2}$, while we have $pass(P'\,,\,T') = \frac{1}{2} \cdot pass(R_1\,,\,(b;\omega)) + \frac{1}{2} \cdot pass(R_1\,,\,(b;\omega))$. Assuming $P \approx P'$, we have $pass(P'\,,\,T') = \frac{1}{2}$, and so $pass(R_1\,,\,(b;\omega)) = \frac{1}{2}$, which is a contradiction with the previous result. So, it is shown that there cannot exist a unique continuation R_1 fulfilling the required characteristics. □

In general, different continuations of the same action corresponding to different states can be distinguished. So we must include states in the sequences defining probabilistic acceptance sets. The next example illustrates the third difference.

Example 3.4 Let P be the non-probabilistic process $a \oplus b$. If we compute its acceptance sets we obtain $\mathcal{A}(P, \epsilon) = \{\{a\}, \{b\}\}$, $\mathcal{A}(P, \langle a \rangle) = \mathcal{A}(P, \langle b \rangle) = \{\emptyset\}$, while for any other trace s we have $\mathcal{A}(P, s) = \emptyset$. It is easy to check that P is equivalent to $Q = (a \oplus b) \oplus (a + b)$ with respect to the equivalence

induced by acceptance sets. That is, the state $\{a, b\}$ generated by the right hand side of Q can be *simulated* by the union of the states $\{a\}$ and $\{b\}$ belonging to P. Consider the probabilization of P, $P' = a \oplus_p b$. Let us suppose that there exist $p_1, p_2, r > 0$ such that P' is testing equivalent to the process $Q' = (a \oplus_{p_1} b) \oplus_{p_2} (a +_r b)$. If we consider the tests $T_1 = (a; \omega) +_{\frac{1}{2}} (b; Nil)$, and $T_2 = (a; \omega) +_{\frac{1}{3}} (b; Nil)$, we get, respectively, $p = p_2 \cdot p_1 + (1 - p_2) \cdot r$ and $p = p_2 \cdot p_1 + (1 - p_2) \cdot \frac{r}{2-r}$ which implies that r should be equal either to 0 or to 1. But both values are not valid in our framework.[8] So, it has been shown that such a process Q', being testing equivalent to P', cannot exist. In fact, P' will have as immediately reachable states $\{(a, 1)\}$, with probability p, and $\{(b, 1)\}$, with probability $1 - p$, while the state $\{(a, s), (b, 1 - s)\}$ is not reachable at all. On the contrary, Q' has as immediately reachable state $\{(a, r), (b, 1 - r)\}$ with a probability equal to $1 - p_2$. □

Once we have explained the changes with respect to the non-probabilistic framework, we will give some previous definitions which are necessary for defining (probabilistic) acceptance sets. First we introduce the notions of *probabilistic state* and (immediately) *reachable state*. Afterwards we will define a relation $P \stackrel{s}{\Longrightarrow}_p P'$ indicating that the process P may evolve into P', with a probability equal to p, after performing the actions appearing in s through a sequence of states which are also indicated in the sequence s.

Definition 3.5 Let $A \subseteq Act \times (0, 1]$. We define the *multiset of actions* of A as $Act(A) = \{\!\{\, a \mid \exists p : \ (a, p) \in A \,\}\!\}$. We say that A is a (*probabilistic*) *state* if every action $a \in Act$ appears at most one time in $Act(A)$ (i.e. $Act(A)$ is a set), and either $\sum \{\!\{\, p \mid (a, p) \in A \,\}\!\}$ is equal to 1, or it is equal to 0 (when $A = \emptyset$). For any probabilistic state A we define the *probability* of the action a in A, denoted by $pro(a, A)$, as p if $(a, p) \in A$, and as 0 otherwise. Given a stable process P we define its (immediately) *reachable probabilistic state*, denoted by $S(P)$, as the set $S(P) = \{(a, p) \mid p = \sum_R \{\!\{\, p_i \mid P \stackrel{a}{\longrightarrow}_{p_i} R \,\}\!\} \wedge p > 0\}$.

Let A be a probabilistic state and $a \in Act$. We say that the pair (A, a) is a *station* if $pro(a, A) > 0$. In order to lighten the notation we will usually denote stations by $A\,a$ instead of (A, a). Sequences of stations will be written as $\langle A_1\,a_1, A_2\,a_2, \ldots, A_n\,a_n \rangle$, and ϵ denotes an empty sequence. Finally, $s_1 \circ s_2$ denotes the concatenation of the sequences s_1 and s_2. □

Usually, when no confusion can arise, we will omit the word *probabilistic* when referring to probabilistic states. Note that immediately reachable states are defined only for stable processes. This is enough since the reachable states for non-stable processes will be defined from the (immediately) reachable states of the stable processes to which it can evolve after executing a generalized

[8] As we indicated in the introduction of Section 2 we do not consider priorities, that is, r is constrained by $0 < r < 1$.

internal transition. If a process cannot be *stabilized* then we have a divergent process (that is, it behaves as Ω). Also note that stable processes have one (and only one) immediately reachable state. Finally, let us remark that if (A, a) is a station then we have both $A \neq \emptyset$ and $a \in Act(A)$.

Definition 3.6 Given the sequence of stations $s = \langle A_1\, a_1, A_2\, a_2, \ldots, A_n\, a_n \rangle$ and $0 < p \leq 1$, we inductively define the relation $P \stackrel{s}{\Longrightarrow}_p P'$ as:

$$P \stackrel{\epsilon}{\Longrightarrow}_p P' \text{ iff } P \rightarrowtail\!\!\!\!\rightarrow^*_p P'$$

$$P \stackrel{s}{\Longrightarrow}_p P' \text{ iff } \exists\, Q_1, P_1, p_1, q_1 : P \rightarrowtail\!\!\!\!\rightarrow^*_{p_1} Q_1 \stackrel{a_1}{\longrightarrow}_{q_1} P_1 \stackrel{s'}{\Longrightarrow}_{p'} P' \wedge S(Q_1) = A_1 \wedge p = \frac{p_1}{r_1} \cdot q_1 \cdot p'$$

where $s' = \langle A_2\, a_2, \ldots, A_n\, a_n \rangle$ and $r_1 = pro(a_1, A_1)$. We will write $P \stackrel{s}{\not\Longrightarrow}$ if there do not exist P' and $p > 0$ such that $P \stackrel{s}{\Longrightarrow}_p P'$. □

As in the previously defined operational relations ($\rightarrowtail\!\!\!\!\rightarrow_p$, $\stackrel{a}{\longrightarrow}_p$ and $\rightarrowtail\!\!\!\!\rightarrow^*_p$), we must take care of the repetitions when generating $\stackrel{s}{\Longrightarrow}_p$ transitions. Intuitively, $P \stackrel{s}{\Longrightarrow}_p P'$ iff P can successively perform the actions a_i passing through a series of stable processes Q_i such that $S(Q_i) = A_i$, and finally evolving into P' by a generalized internal transition (that is, $\rightarrowtail\!\!\!\!\rightarrow^*_q$ for some q). Note that P' must be stable. The value p is computed from the probabilities with which the stable processes Q_i are reached (by a generalized internal transition), from the *relative* probabilities of executing the actions a_i (i.e. the values $\frac{q_i}{pro(a_i, A_i)}$, which denote the quotient of the probability of performing the corresponding transition among the addition of the probabilities associated with the transitions labeled by a_i), and from the probability associated with the last generalized internal transition reaching P'. Now we can define the probabilistic acceptance sets of a process after performing a sequence of stations.

Definition 3.7 Let P be a process, s be a sequence of stations, and A be a state. We say that P *reaches* A with probability p *after performing* the sequence s, denoted by $P \stackrel{s}{\Longrightarrow}_p A$, if $p = \sum_{P'} \{\!\!\{\, p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \wedge S(P') = A \,\}\!\!\}$.

Given a non-empty sequence of stations $s = \langle A_1\, a_1, A_2\, a_2, \ldots, A_n\, a_n \rangle$, we say that A_n is the *last state* of the sequence, denoted by $last(s)$. Besides, we define the *sequence containing all the stations but the last one*, denoted by $abl(s)$, as $\langle A_1\, a_1, A_2\, a_2, \ldots, A_{n-1}\, a_{n-1} \rangle$.

Let P be a process and s be a sequence of stations. We define the (*probabilistic*) *acceptance sets* of P *after* s, denoted by $\mathcal{A}(P, s)$, as

$$\mathcal{A}(P, s) = \begin{cases} \{(A, p_A) \mid P \stackrel{s}{\Longrightarrow}_{p_A} A \ \wedge\ p_A > 0\} & \text{if } s = \epsilon \\ \{(A, p_A/q_s) \mid P \stackrel{s}{\Longrightarrow}_{p_A} A \ \wedge\ p_A > 0 \ \wedge\ P \stackrel{abl(s)}{\Longrightarrow}_{q_s} last(s)\} & \text{otherwise} \end{cases}$$

□

Note that we have overloaded the relation $\stackrel{s}{\Longrightarrow}_p$. Nevertheless, it will be clear from the context whether we are referring to a transition (i.e. $P \stackrel{s}{\Longrightarrow}_p P'$) or to the probability of reaching a state (i.e. $P \stackrel{s}{\Longrightarrow}_p A$). When no confusion can arise, we will omit the term probabilistic. Intuitively, in order to compute the acceptance sets of a process P after performing a sequence s, we firstly compute the states that P may reach after performing that sequence. The probability associated with these states is computed by dividing the total probability of reaching each state (note that every state can possibly be reached several times, by using different derivations) by the probability of reaching the last state of the sequence s (after performing the stations of this sequence). This division is just a technicality allowing that after any sequence s the addition of the probabilities associated with the states belonging to the corresponding acceptance set is equal to 1 minus the probability of diverging after the sequence. Let us remark that a (totally) divergent process cannot reach any state, that is, $\mathcal{A}(\Omega, s) = \emptyset$ for any sequence s, because there do not exist P and $0 < p \leq 1$ such that $\Omega \stackrel{s}{\Longrightarrow}_p P$.

Definition 3.8 (*Alternative Characterization of* $\approx$) Let P, P' be processes. We write $P \cong P'$ if $\mathcal{A}(P, s) = \mathcal{A}(P', s)$ for any sequence of stations s. □

Example 3.9 Consider the process $P = ((a +_{\frac{1}{3}} b) \oplus_{\frac{1}{2}} (b; c)) \oplus_{\frac{1}{2}} (b; d)$. The operational behavior of P is given by:

$$P \rightarrowtail_{\frac{1}{2}} P_1 = (a +_{\frac{1}{3}} b) \oplus_{\frac{1}{2}} (b; c) \rightarrowtail_{\frac{1}{2}} P_3 = b; c \stackrel{b}{\longrightarrow}_1 P_4 = c \stackrel{c}{\longrightarrow}_1 \mathit{Nil}$$
$$\rightarrowtail_{\frac{1}{2}} P_2 = a +_{\frac{1}{3}} b \stackrel{a}{\longrightarrow}_{\frac{1}{3}} \mathit{Nil}$$
$$\stackrel{b}{\longrightarrow}_{\frac{2}{3}} \mathit{Nil}$$
$$P \rightarrowtail_{\frac{1}{2}} P_5 = b; d \stackrel{b}{\longrightarrow}_1 P_6 = d \stackrel{d}{\longrightarrow}_1 \mathit{Nil}$$

In order to compute the acceptance sets of P after a sequence of stations s we need to compute the processes to which P can evolve after executing the sequence s. We compute $\mathcal{A}(P, \epsilon)$. We obtain $P \stackrel{\epsilon}{\Longrightarrow}_{\frac{1}{4}} P_2$, $P \stackrel{\epsilon}{\Longrightarrow}_{\frac{1}{4}} P_3$, $P \stackrel{\epsilon}{\Longrightarrow}_{\frac{1}{2}} P_5$. Also, $S(P_2) = A = \{(a, \frac{1}{3}), (b, \frac{2}{3})\}$, while $S(P_3) = S(P_5) = B = \{(b, 1)\}$. So, we finally obtain $\mathcal{A}(P, \epsilon) = \{\ (A, 1/4), (B, 3/4)\ \}$.

Now, let us compute the values for $s = \langle A\, a \rangle$ (note that $\mathit{last}(s) = A$ and $\mathit{abl}(s) = \epsilon$). We have $P \stackrel{\langle A\, a \rangle}{\Longrightarrow}_{\frac{1}{4}} \mathit{Nil}$ and so $P \stackrel{\langle A\, a \rangle}{\Longrightarrow}_{\frac{1}{4}} \emptyset$. Besides, $P \stackrel{\mathit{abl}(s)}{\Longrightarrow}_{\frac{1}{4}} A$. So, we have $\mathcal{A}(P, \langle A\, a \rangle) = \{\ (\emptyset, 1)\ \}$. Similarly, we obtain $\mathcal{A}(P, \langle A\, b \rangle) = \{\ (\emptyset, 1)\ \}$.

In order to compute $\mathcal{A}(P, s)$, for $s = \langle B\, b \rangle$, we have $P \stackrel{\langle B\, b \rangle}{\Longrightarrow}_{\frac{1}{4}} P_4$ and $P \stackrel{\langle B\, b \rangle}{\Longrightarrow}_{\frac{1}{2}} P_6$. Given the fact that $S(P_4) = \{(c, 1)\}$, $S(P_6) = \{(d, 1)\}$, and taking into account that $P \stackrel{\mathit{abl}(s)}{\Longrightarrow}_{\frac{3}{4}} B$, we obtain $\mathcal{A}(P, \langle B\, b \rangle) = \{(\{(c, 1)\}, 1/3), (\{(d, 1)\}, 2/3)\}$ (note that $(\frac{1/4}{3/4} = \frac{1}{3}$ and $\frac{1/2}{3/4} = \frac{2}{3})$. In a similar way to that used for $\mathcal{A}(P, \langle A\, a \rangle)$, we

obtain $\mathcal{A}(P, \langle B\, b, \{(c,1)\}\, c\rangle) = \{\ (\emptyset, 1)\ \}$ and $\mathcal{A}(P, \langle B\, b, \{(d,1)\}\, d\rangle) = \{\ (\emptyset, 1)\ \}$. Finally, for any other sequence s we have $\mathcal{A}(P, s) = \emptyset$.

It can be checked that P is equivalent to $P' = (a +_{\frac{1}{3}} b) \oplus_{\frac{1}{4}} (b; (c \oplus_{\frac{1}{3}} d))$, that is $P \cong P'$. In fact, we will show later that P' is the *normal form* of P.

Consider now the process $P = (a; b) \oplus_{\frac{1}{3}} \Omega$, and the states $A = \{(a,1)\}$ and $B = \{(b,1)\}$. We obtain $\mathcal{A}(P, \epsilon) = \{\ (A, 1/3)\ \}$, $\mathcal{A}(P, \langle A\, a\rangle) = \{\ (B, 1)\ \}$, $\mathcal{A}(P, \langle A\, a, B\, b\rangle) = \{\ (\emptyset, 1)\ \}$, and $\mathcal{A}(P, s) = \emptyset$ for any other sequence s.

The following example shows how acceptance sets can be *computed* for recursive processes. Consider the process $P = recX.(a \oplus_p X)$ and the state $A = \{(a,1)\}$. The operational transitions of P are:

$$P >\!\!\longrightarrow_1 a \oplus_p P >\!\!\longrightarrow_p a \xrightarrow{a}_1 Nil$$
$$P >\!\!\longrightarrow_1 a \oplus_p P >\!\!\longrightarrow_{1-p} P >\!\!\longrightarrow_1 a \oplus_p P >\!\!\rightarrow_p a \xrightarrow{a}_1 Nil$$
$$P >\!\!\longrightarrow_1 a \oplus_p P >\!\!\longrightarrow_{1-p} P >\!\!\longrightarrow_1 a \oplus_p P >\!\!\rightarrow_{1-p} P >\!\!\rightarrow_1 a \oplus_p P >\!\!\longrightarrow_p a \xrightarrow{a}_1 Nil$$
$$\ldots \qquad \ldots$$

Taking into account that $p \cdot \sum_{i=0}^{\infty}(1-p)^i = p \cdot \frac{1}{1-(1-p)} = \frac{p}{p} = 1$, it easily follows that $\mathcal{A}(P, \epsilon) = \{(A,1)\}$ and $\mathcal{A}(P, \langle A\, a\rangle) = \{(\emptyset, 1)\}$, while for any other sequence $s \notin \{\epsilon,\ \langle A\, a\rangle\}$ we have $\mathcal{A}(P, s) = \emptyset$. Note that $P \cong a; Nil$. □

In the rest of this section we will show that the relations given by Definitions 2.8 and 3.8 are *equal*, that is, they relate the same processes. To do so, we first associate with any syntactic process its so called *computation tree*. In a computation tree there is a strict alternation between *internal* and *external* nodes.[9] If we associate generalized internal choices with internal nodes, and generalized external choices with external nodes, we can see these trees as *generalized* syntactic processes. The difference with respect to usual processes comes from the fact that generalized processes may be infinite (if they are generated from a recursive process). We will associate with any syntactic process an *equivalent* generalized process. The definition of these processes will be completely operational, being based on acceptance sets.

Definition 3.10 Let P be a process. We define the *normalized process associated* with P, denoted by $\widehat{\mathcal{A}}(P)$, as $\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P, \epsilon)$, where

$$\widehat{\mathcal{A}}(P, s) = \bigoplus_{i=1}^{n} [p_i] \sum_{j=1}^{r_i} [p_{i,j}]\, a_{i,j}\, ; \widehat{\mathcal{A}}(P, s \circ \langle A_i\, a_{i,j}\rangle)$$

[9] A similar situation appears in the *alternating model* [17]. However, they *alternate* between probabilistic nodes (equivalent to our internal nodes) and non-deterministic nodes (where there is no probabilistic information).

where $\mathcal{A}(P,s) = \{(A_1,p_1),\ldots,(A_n,p_n)\}$, $A_i = \{(a_{i,1},p_{i,1}),\ldots,(a_{i,r_i},p_{i,r_i})\}$, and assuming that $\sum_{j=1}^{0} P_i$ stands for the process Nil. □

Note that $1-\sum p_i$ indicates the probability with which the process can diverge in each of its internal states. Because of the close relationship between $\widehat{\mathcal{A}}(P)$ and acceptance sets of the form $\mathcal{A}(P,s)$, it is reasonable to call $\widehat{\mathcal{A}}(P)$ the *normal form* of process P. Let us comment on the inductive character of the previous definition. We say that the normal form of a process P is defined as $\widehat{\mathcal{A}}(P,\epsilon)$. So, we compute the first *floor* of the normal form and then we recursively call the function by computing $\widehat{\mathcal{A}}(P,\langle A_i\, a_{i,j}\rangle)$, for any immediately reachable state A_i and any $a_{i,j} \in A_i$. This recursive call will generate the second *floor* of the normal form and so on.

Example 3.11 Consider the process $P = ((a +_{\frac{1}{3}} b) \oplus_{\frac{1}{2}} (b;c)) \oplus_{\frac{1}{2}} (b;d)$ given in Example 3.9. In order to compute $\widehat{\mathcal{A}}(P)$ we must compute $\widehat{\mathcal{A}}(P,\epsilon)$. The immediately reachable states of P are $A = \{(a,\frac{1}{3}),(b,\frac{2}{3})\}$, with probability $\frac{1}{4}$, and $B = \{(b,1)\}$, with probability $\frac{3}{4}$. So we have

$$\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P,\epsilon) = \left(\left(a;\widehat{\mathcal{A}}(P,\langle A\, a\rangle\right) +_{\frac{1}{3}} \left(b;\widehat{\mathcal{A}}(P,\langle A\, b\rangle\right)\right) \oplus_{\frac{1}{4}} \left(b;\widehat{\mathcal{A}}(P,\langle B\, b\rangle\right)$$

Now, we need to compute $\widehat{\mathcal{A}}(P,\langle A\, a\rangle)$, $\widehat{\mathcal{A}}(P,\langle A\, b\rangle)$, and $\widehat{\mathcal{A}}(P,\langle B\, b\rangle)$. From Example 3.9 we obtain $\mathcal{A}(P,\langle A\, a\rangle) = \mathcal{A}(P,\langle A\, b\rangle) = \{\ (\emptyset,1)\ \}$. So we have $\widehat{\mathcal{A}}(P,\langle A\, a\rangle) = \widehat{\mathcal{A}}(P,\langle A\, b\rangle) = Nil$. From the same example we also obtain $\mathcal{A}(P,\langle B\, b\rangle) = C = \{\ (\{(c,1)\},1/3),(\{(d,1)\},2/3)\ \}$. So,

$$\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P,\epsilon) = \left((a;Nil) +_{\frac{1}{3}} (b;Nil)\right) \oplus_{\frac{1}{4}} \left(b;\left(\begin{array}{c}\left(c;\widehat{\mathcal{A}}(P,\langle B\, b, C\, c\rangle\right)\\ \oplus_{\frac{1}{3}}\ \left(d;\widehat{\mathcal{A}}(P,\langle B\, b, C\, d\rangle\right)\end{array}\right)\right)$$

Taking into account that $\mathcal{A}(P,\langle B\, b, C\, c\rangle) = \mathcal{A}(P,\langle B\, b, C\, d\rangle) = \{\ (\emptyset,1)\ \}$, we finish the definition of $\widehat{\mathcal{A}}(P)$:

$$\widehat{\mathcal{A}}(P) = \left((a;Nil) +_{\frac{1}{3}} (b;Nil)\right) \oplus_{\frac{1}{4}} \left(b;\left((c;Nil) \oplus_{\frac{1}{3}} (d;Nil)\right)\right)$$

□

Let us remark that the operational semantics can be extended to deal with normalized processes. Thus, tests can be also applied to normalized processes. In order to generate the transitions of $\widehat{\mathcal{A}}(P)$ we apply the operational rules given in Figure 3 to $\widehat{Q} = \widehat{\mathcal{A}}(P,\epsilon)$. The first rule indicates that if the probabilities associated with the first generalized internal choice add to less than one then we generate an internal transition to divergence. The second rule generates internal transitions for each of the components of the generalized internal choice. These transitions reach (normalized) processes with a unique component. In the third rule we consider the case of a generalized internal choice with a unique component having as associated probability a value less

$$\frac{\widehat{Q} = \bigoplus_{i=1}^{n} [p_i] \sum_{j=1}^{r_i} [p_{i,j}]\, a_{i,j}\,;\widehat{\mathcal{A}}(P, s \circ \langle A_i\, a_{i,j}\rangle) \;\wedge\; \sum p_i < 1}{\widehat{Q} >\!\!\longrightarrow_{1-\Sigma p_i} \Omega}$$

$$\frac{\widehat{Q} = \bigoplus_{i=1}^{n} [p_i] \sum_{j=1}^{r_i} [p_{i,j}]\, a_{i,j}\,;\widehat{\mathcal{A}}(P, s \circ \langle A_i\, a_{i,j}\rangle) \;\wedge\; n > 1}{\widehat{Q} >\!\!\longrightarrow_{p_k} \bigoplus_{i=1}^{1} [1] \sum_{j=1}^{r_k} [p_{k,j}]\, a_{k,j}\,;\widehat{\mathcal{A}}(P, s \circ \langle A_k\, a_{k,j}\rangle)}$$

$$\frac{\widehat{Q} = \bigoplus_{i=1}^{n} [p_i] \sum_{j=1}^{r_i} [p_{i,j}]\, a_{i,j}\,;\widehat{\mathcal{A}}(P, s \circ \langle A_i\, a_{i,j}\rangle) \;\wedge\; n = 1 \;\wedge\; p_1 < 1}{\widehat{Q} >\!\!\longrightarrow_{p_1} \bigoplus_{i=1}^{1} [1] \sum_{j=1}^{r_1} [p_{1,j}]\, a_{1,j}\,;\widehat{\mathcal{A}}(P, s \circ \langle A_1\, a_{1,j}\rangle)}$$

$$\frac{\widehat{Q} = \bigoplus_{i=1}^{n} [p_i] \sum_{j=1}^{r_i} [p_{i,j}]\, a_{i,j}\,;\widehat{\mathcal{A}}(P, s \circ \langle A_i\, a_{i,j}\rangle) \;\wedge\; n = 1 \;\wedge\; p_1 = 1}{\widehat{Q} \xrightarrow{a_{1,j}}_{p_{1,j}} \widehat{\mathcal{A}}(P, s \circ \langle A_1\, a_{1,j}\rangle)}$$

Fig. 3. Operational Semantics for Normalized Processes.

than 1. The corresponding internal transition is generated. Finally, the fourth rule considers the case of *deterministic* normalized processes. Even though the generalized internal choice operator remains, given the fact that it is associated to a unique process we can omit the trailing $>\!\!\longrightarrow_1$ transition, and so we consider the external transitions associated with the (unique) generalized external choice. These transitions reach the next floor of the normalized process.

The next result, whose proof is trivial from the definition of normalized process, states that a process and its normal form are equivalent in terms of acceptance sets.

Lemma 3.12 For any process P we have $P \cong \widehat{\mathcal{A}}(P)$. □

Moreover, we have defined normal forms in such a way that it trivially follows their uniqueness in each equivalence class (up to commutativity).

Lemma 3.13 Let P, P' be processes. We have $P \cong P'$ iff $\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P')$. □

Example 3.14 The process P' appearing in Example 3.9 is a normal form, that is $P' = \widehat{\mathcal{A}}(P')$. The process P appearing in the same example is not a normal form, but it can be checked that $\widehat{\mathcal{A}}(P) = P'$ (see Example 3.11). □

The first step for proving that the alternative characterization based on acceptance sets is equivalent to the testing semantics consists in showing that a process is testing equivalent to its normal form. First, we need the following results (the proof of the first one is trivial while the second one can be easily proved by induction on the length of the sequence s).

Lemma 3.15 For any process P and any sequence of stations s we have

$$\sum_{P'} \{\!|\, p \mid P \stackrel{s}{\Longrightarrow}_p P' \,|\!\} = \begin{cases} \sum_A \{\!|\, q \mid (A,q) \in \mathcal{A}(P,\epsilon) \,|\!\} & \text{if } s = \epsilon \\ \sum_A \{\!|\, q \cdot q_s \mid (A,q) \in \mathcal{A}(P,s) \ \wedge\ P \stackrel{abl(s)}{\Longrightarrow}_{q_s} last(s) \,|\!\} & \text{if } s \neq \epsilon \end{cases}$$

□

Lemma 3.16 For any process P and any sequence of stations s we have that $\sum_{P'} \{\!|\, p \mid P \stackrel{s}{\Longrightarrow}_p P' \,|\!\} = \sum_{P'} \{\!|\, p \mid \widehat{\mathcal{A}}(P) \stackrel{s}{\Longrightarrow}_p P' \,|\!\}$ □

The following result states that a process is testing equivalent to its associated normal form. It follows from the previous two results and by taking into account that the operational behavior of a process can be encoded into the transitions of its associated generalized process (the proof is easy but tedious).

Proposition 3.17 For any process P we have $P \approx \widehat{\mathcal{A}}(P)$. □

Now, we are going to prove that the testing equivalence and the equivalence based on acceptance sets are the same. For this purpose, because of Lemma 3.13, it will be enough to restrict ourselves to normal forms. First, we need a *uniqueness* technical result (a slightly different result appears in [36]). The proof is highly technical. We include it in the Appendix because we think that such a result (or a similar one) will be needed for any alternative characterization based on acceptance sets of a testing semantics where probabilities are considered (for instance, a similar result is used in the alternative characterization of the stochastic testing semantics presented in [26]).

Lemma 3.18 Let f and f' be two rational functions of $n \geq 0$ variables $x_1, \ldots, x_n$, defined as follows:

$$f = \sum_{i \in I} \frac{c_i}{1 + \sum_{j=1}^{n} d_{j,i} \cdot x_j} \qquad f' = \sum_{i' \in I'} \frac{c'_{i'}}{1 + \sum_{j=1}^{n} d'_{j,i'} \cdot x_j}$$

where I, I' are finite sets of indices; $c_i, c'_{i'} > 0$; for each distinct $r, s \in I$, the tuples $(d_{1,r}, d_{2,r}, \ldots, d_{n,r})$ and $(d_{1,s}, d_{2,s}, \ldots, d_{n,s})$ are distinct; and for each distinct $r, s \in I'$, the tuples $(d'_{1,r}, d'_{2,r}, \ldots, d'_{n,r})$ and $(d'_{1,s}, d'_{2,s}, \ldots, d'_{n,s})$ are distinct. If $f = f'$, then there exists a bijection $h : I \longrightarrow I'$ such that $d_{j,i} = d'_{j,h(i)}$ and $c_i = c'_{h(i)}$ for all $i \in I$ and $1 \leq j \leq n$. So, the expressions defining f and f' are identical up to commutativity. □

The next technical result will be also used in the proof of the characterization theorem (the proof is immediate).

Lemma 3.19 Let $p_1 \ldots p_n, p'_1 \ldots p'_n \geq 0$ such that $\sum p_i = \sum p'_i$, and $r, r' > 0$. Then, $\forall\ 1 \leq i \leq n :\ \frac{p_i}{r} = \frac{p'_i}{r'}$ implies $\forall\ 1 \leq i \leq n :\ p_i = p'_i\ \wedge\ r = r'$. □

Finally, we present the main result of this section: Two normal forms are testing equivalent iff they are (syntactically) equal. Because of the uniqueness of normal forms (see Lemma 3.13) the right to left implication is trivial. The proof of the other implication is presented in the appendix.

Theorem 3.20 Let P, P' be processes. $\widehat{\mathcal{A}}(P) \approx \widehat{\mathcal{A}}(P')$ iff $\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P')$. □

The result that we are looking for follows from this one: Two processes are testing equivalent iff they are equivalent with respect to acceptance sets.

Corollary 3.21 For any processes P and P' we have $P \approx P'$ iff $P \cong P'$.

Proof: From Proposition 3.17 we obtain $P \approx P'$ iff $\widehat{\mathcal{A}}(P) \approx \widehat{\mathcal{A}}(P')$. By applying Theorem 3.20 we obtain $\widehat{\mathcal{A}}(P) \approx \widehat{\mathcal{A}}(P')$ iff $\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P')$. Finally, by Lemma 3.13, we have $\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P')$ iff $P \cong P'$. □

The proof of Theorem 3.20, in addition to prove that the relations $\approx$ and $\cong$ coincide, gives us a set of *essential* tests. This set has enough discriminatory power to distinguish any pair of non-equivalent processes. We will precisely define this set of tests, which we call *probabilistic barbs*. Let us remark that there exists a great similitude between our probabilistic barbs and *probabilistic traces* in [9], if we consider the latter as probabilistic tests.

Definition 3.22 The set of *probabilistic barbs*, denoted by $\mathcal{PB}$, is defined by means of the following BNF expression:

$$T ::= \sum_{i=1}^{s} [p_i]\ (b_i; \mathit{Nil}) +_p \omega \mid \sum_{i=1}^{s} [p_i]\ b_i; T_i \qquad \text{where } T_i = \begin{cases} T & \text{if } i = s \\ \mathit{Nil} & \text{otherwise} \end{cases}$$

where $p \in (0,1)$, $\sum p_i = 1$, and $b_i \in Act$. We will write $P \approx_{\mathcal{PB}} Q$ if for any $T \in \mathcal{PB}$ we have $pass(P\ ,\ T)\ =\ pass(Q\ ,\ T)$. □

Intuitively, a probabilistic barb is either a generalized external choice whose continuations are the test *Nil* composed in external choice with the action ω, or a generalized external choice whose continuations are the test *Nil*, but one, whose continuation is another probabilistic barb.

Theorem 3.23 For any processes P and P', we have $P \approx P'$ iff $P \approx_{\mathcal{PB}} P'$.

Proof: Immediate from Theorem 3.17, which indicates that two processes are testing equivalent iff their normal forms are, and from the proof of The-

orem 3.20 (see the Appendix of this paper), in which we see that in order to distinguish two different normal forms is enough to consider probabilistic barbs. □

4 Fully Abstract Denotational Semantics

In this section we present a fully abstract denotational semantics with respect to the testing semantics. It is based on *acceptance trees* [18]. First, we will define the semantic domain and then we will define semantic functions corresponding to the syntactic operators of the language. Finally, we will show that the denotational semantics relates the same process that our testing equivalence. The semantic domain will be the set of *probabilistic acceptance trees* (in short *pat*) over Act. We will denote this domain by $\mathbf{PAT}_{Act}$. The elements will be trees with two kind of nodes: internal (labeled by $\oplus$) and external (labeled by $+$). These trees fulfill the following conditions:

(1) The root of the tree is an internal node.
(2) The arcs outgoing from an internal node verify the following conditions:
   - They are labeled by a state (see Definition 3.5) and a probability, being all the states different.
   - The sum of the probabilities labeling these arcs must be less than or equal to 1.
   - These arcs reach external nodes.
(3) The arcs outgoing from an external node verify the following conditions:
   - They are labeled by the actions belonging to the state labeling the incoming arc.
   - For any action in that state there exist a unique arc labeled by this action.
   - These arcs reach internal nodes.

We usually will denote by $R, R_1, \ldots$ to the elements of $\mathbf{PAT}_{Act}$. Let us remark that it is possible that several outgoing arcs from an internal node are labeled with states which have the same set of actions but with different probability distributions. For example, there can exist arcs labeled by the states $\{(a, \frac{1}{2}), (b, \frac{1}{2})\}$ and $\{(a, \frac{1}{3}), (b, \frac{2}{3})\}$. In internal nodes, the sum of probabilities associated with outgoing arcs can be less than one. The difference between this sum and 1 denotes the probability of divergence at that point.

Some examples of probabilistic acceptance trees are given in Figure 4. As we did for acceptance sets, we will characterize the nodes of these trees as the reachable nodes after a sequence of pairs (state, action). In the following definition we present the concepts needed to handle the reachable states of a *pat* after a sequence of pairs (state, action).

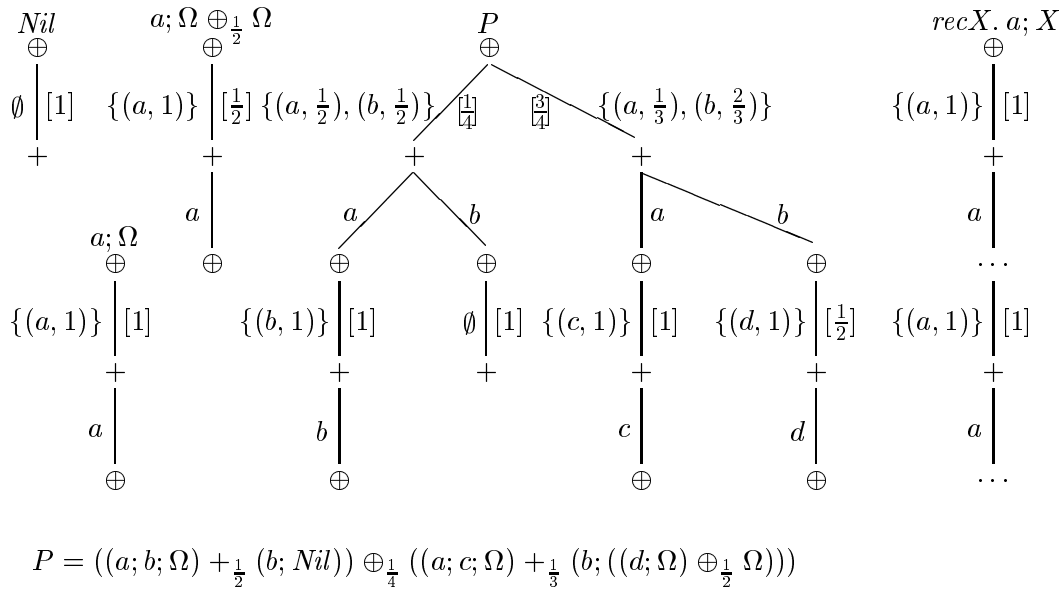

Fig. 4. Examples of probabilistic acceptance trees.

Definition 4.1 Let R be a *pat* and A be a state. We define the *probability* with which R (immediately) *reaches* the state A, denoted by $p(R, A)$, as p_A if there exists an outgoing arc labeled $[p_A]\ A$ from the root node of R; if there does not exist such an arc then $p(R, A) = 0$.

Let A be a state such that $p_A = p(R, A) > 0$ and let $a \in act(A)$. We define the *continuation after the performance of a in A*, denoted by $R/(A, a)$, as the tree whose root is the internal node reached by the arc labeled by a outgoing from the external node reached by the arc labelled by $[p_A]\ A$.

Let $s = \langle A_1 a_1, A_2 a_2, \ldots, A_n a_n \rangle$ be a sequence of stations. We define the *probability* with which R *reaches the external node associated with A after performing the sequence s*, denoted by $p(R, s, A)$, as

$$p(R, \epsilon, A) = p(R, A)$$
$$p(R, \langle A_1 a_1 \rangle \circ s, A) = p(R, A_1) \cdot p(R/(A_1, a_1), s, A)$$

□

In Figure 5 we present a graphical representation of the previously defined concepts. Let us note that a tree R can be easily rebuilt from the values $p(R, s, A)$. First, in order to determinate the (local) probabilities associated with the states labeling outgoing arcs from an internal node, which is reached after a sequence of stations, it is enough to divide by the probability of reaching the last state of the sequence. Formally, if we denote by $\text{prob}(R, s, X)$ to the probability labeling the outgoing arc labeled by X from the internal node of

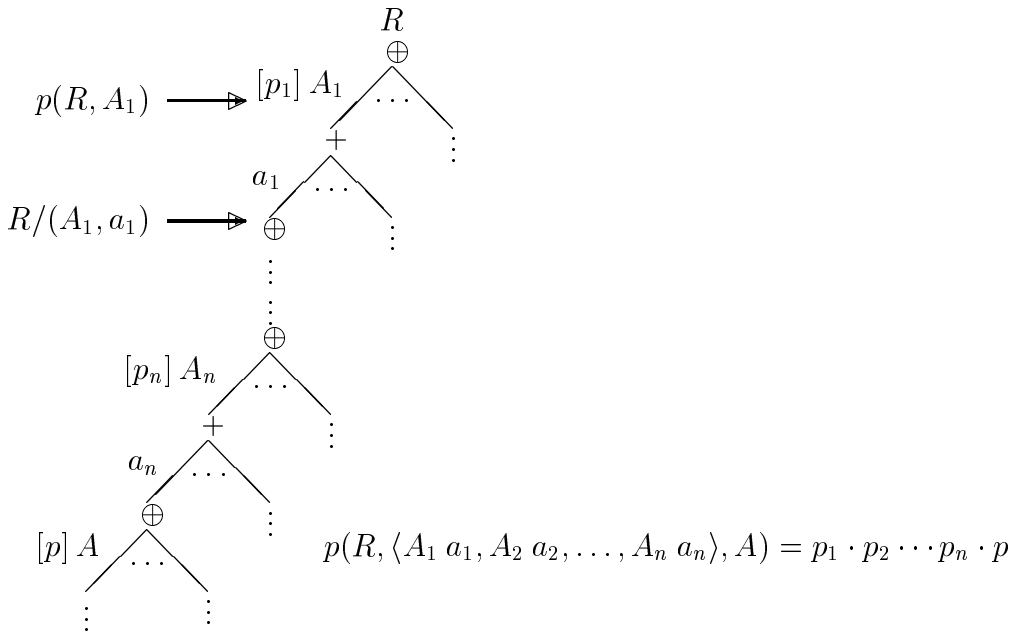

Fig. 5. Definition of $p(R, A)$, $R/(A, a)$, and $p(R, s, A)$.

R which is reached after the sequence s [10], we have:

$$\text{prob}(R, s, X) = \begin{cases} p(R, X) & \text{if } s = \epsilon \\ \dfrac{p(R, s, X)}{p(R, abl(s), A)} & \text{otherwise} \end{cases}$$

In fact, in the rest of this section we will implicitly use this property for defining some probabilistic acceptance trees. Next we define the order between probabilistic acceptance trees.

Definition 4.2 Let R_1, R_2 be *pat*'s. We write $R_1 \sqsubseteq_{\mathbf{PAT}} R_2$ if for any sequence s and state A we have $p(R_1, s, A) \leq p(R_2, s, A)$. We write $R_1 =_{\mathbf{PAT}} R_2$ if $R_1 \sqsubseteq_{\mathbf{PAT}} R_2$ and $R_2 \sqsubseteq_{\mathbf{PAT}} R_1$. □

The previously defined relation $\sqsubseteq_{\mathbf{PAT}}$ is trivially an order. Moreover, it induces a complete partial order. In the Appendix of this paper we show the proof of this result.

Theorem 4.3 $(\mathbf{PAT}_{Act}, \sqsubseteq_{\mathbf{PAT}})$ is a *complete partial order* (cpo). □

Next we define semantic functions for the syntactic operators of the language. We will show that these functions are continuous. This fact allows us to apply fixed point techniques for defining the semantics of recursive processes. As usually, we will denote by $[\![P]\!]$ to the semantics of the syntactic process P.

[10] That is, if we have $s = \langle A_1\, a_1, A_2\, a_2, \ldots, A_n\, a_n \rangle$ then we consider the tree $R/(A_1, a_1)/(A_2, a_2) \cdots /(A_n, a_n)$.

The process Nil has a unique (immediately) reachable state: The state $\emptyset$, which is reached with a probability equal to 1. So, $[\![Nil]\!]$ is a tree having an internal node which has a unique arc labeled by the empty state and the probability 1. This arc reaches an external node having no outgoing arcs. That is, $p([\![Nil]\!], s, A) = 1$ if $s = \epsilon \ \wedge \ A = \emptyset$; otherwise, we have $p([\![Nil]\!], s, A) = 0$.

The process Ω has no reachable states. So, $[\![\Omega]\!]$ is a tree having a unique internal node without outgoing arcs. That is, for any sequence s and state A we have $p([\![\Omega]\!], s, A) = 0$.

Regarding the prefix operator, for each $a \in Act$, we define the semantic function $a;\_ :: \ \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$. The acceptance tree $a; R$ is equal to the tree R but prefixed with an internal node and an external node, corresponding to the action a. That is, its root will be an internal node having a unique outgoing arc labeled by the state $\{(a, 1)\}$ and the probability 1. This arc reaches an external node having a unique outgoing arc labeled by the action a. This arc reaches the root of R.

$$p(a; R, s, A) = \begin{cases} 1 & \text{if } \ s = \epsilon \ \wedge \ A = \{(a, 1)\} \\ p(R, s', A) & \text{if } \ s = \langle \{(a, 1)\}\, a \rangle \circ s' \\ 0 & \text{otherwise} \end{cases}$$

Regarding the internal choice operator, given a value $p \in (0, 1)$, the function $\_ \oplus_p \_ :: \ \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ returns a tree which is the union of the parameters, considering the probability p. A state will be reachable in the new tree if it can be reached in any of the arguments. In this case, the probability with which the state is reached in the first (resp. second) argument is multiplied by p (resp. $1-p$). The sum of these values gives us the probability with which this state is reached in the tree $R_1 \oplus_p R_2$.

$$p(R_1 \oplus_p R_2, s, A) = p \cdot p(R_1, s, A) + (1 - p) \cdot p(R_2, s, A)$$

Finally, for the external choice operator, given a value $p \in (0, 1)$ the function $\_ +_p \_ :: \ \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ returns an acceptance tree representing the *external* union of the parameters with respect to the probability p. Before we define the semantic function $+_p$, we will introduce an auxiliar function which joins two states according to a probability. If one of the states is empty then the result is the other one; otherwise, the result will be a new state whose set of actions is the union of the set of actions of the arguments. The probability associated with these actions is computed from the probabilities that they have in the former states and the parameter of the function.

Definition 4.4 Let X, Y be states, and $p \in (0, 1)$. We define the *union* of

the states X and Y with *associated probability equal to* p as

$$X \cup_p Y = \begin{cases} X & \text{if } Y = \emptyset \\ Y & \text{if } X = \emptyset \\ \{(a, p \cdot pro(a, X) + (1-p) \cdot pro(a, Y))\} & \text{otherwise} \end{cases}$$

□

In order to compute the result of the application of the semantic function $+_p$, we must consider two cases. As in the non-probabilistic case, we must distinguish between the root and the continuations under the root.

Root of the tree $R_1 +_p R_2$

The arcs outgoing from the root of the new tree are defined by considering the union of the initial states of the trees that we are composing.

$$p(R_1 +_p R_2, \epsilon, A) = \sum_{A = B \cup_p C} p(R_1, \epsilon, B) \cdot p(R_2, \epsilon, C)$$

That is, there exists an outgoing arc labeled by the state A if there exist two arcs, one outgoing from the root of R_1 labeled by B, and another one from the root of R_2 labeled by C, such that $A = B \cup_p C$. The probability labeling this new arc is equal to the addition of the products of the probabilities labeling the arcs corresponding to each pair of those states. As an immediate consequence, we have that the function $+_p$ is strict in both arguments. So, if one of the arguments is the tree associated with Ω then the result will be $[\![\Omega]\!]$. That is, $[\![P +_p \Omega]\!] = [\![\Omega +_p P]\!] = [\![\Omega]\!]$, for any process P and $0 < p < 1$. This is so because there are no outgoing arcs from the root of $[\![\Omega]\!]$. Let us also remark that Nil is an identity element of these functions: $[\![P +_p Nil]\!] = [\![Nil +_p P]\!] = [\![P]\!]$, for any process P and $0 < p < 1$.

Example 4.5 Let us consider the following processes: $P_1 = (a +_{\frac{1}{2}} b) \oplus_{\frac{1}{3}} a$, $P_2 = b \oplus_{\frac{1}{4}} Nil$, and $P_3 = b \oplus_{\frac{1}{4}} \Omega$. We have:

$$\begin{cases} p([\![P_1]\!], \epsilon, C) = \frac{1}{3} \\ p([\![P_1]\!], \epsilon, A_1) = \frac{2}{3} \end{cases} \qquad \begin{cases} p([\![P_2]\!], \epsilon, B_1) = \frac{1}{4} \\ p([\![P_2]\!], \epsilon, \emptyset) = \frac{3}{4} \end{cases} \qquad \begin{cases} p([\![P_3]\!], \epsilon, B_1) = \frac{1}{4} \end{cases}$$

where $A_1 = \{(a, 1)\}, B_1 = \{(b, 1)\}$, and $C = \{(a, \frac{1}{2}), (b, \frac{1}{2})\}$. Additionally, we have $p([\![P_i]\!], \epsilon, X) = 0$ for any other state X. Let us show how the roots of some compositions, using $+_p$, are defined:

- The root of the tree $R_1 = [\![P_1]\!] +_{\frac{1}{2}} [\![P_2]\!]$ is given by:

$$p(R_1, \epsilon, A_1) = \tfrac{2}{3} \cdot \tfrac{3}{4} = \tfrac{1}{2} \qquad \text{(join } A_1 \text{ of } P_1 \text{ with } \emptyset \text{ of } P_2\text{)}$$

$$p(R_1, \epsilon, \{(a, \tfrac{1}{4}), (b, \tfrac{3}{4})\}) = \tfrac{1}{3} \cdot \tfrac{1}{4} = \tfrac{1}{12} \qquad \text{(join } C \text{ of } P_1 \text{ with } B_1 \text{ of } P_2\text{)}$$

$$p(R_1, \epsilon, C) = \tfrac{1}{3} \cdot \tfrac{3}{4} + \tfrac{2}{3} \cdot \tfrac{1}{4} = \tfrac{5}{12} \qquad \text{(join } C \text{ of } P_1 \text{ with } \emptyset \text{ of } P_2 \text{ and } A_1 \text{ of } P_1 \text{ with } B_1 \text{ of } P_2\text{)}$$

- The root of the tree $R_2 = [\![P_1]\!] +_{\frac{1}{3}} [\![P_3]\!]$ is given by:

$$p(R_2, \epsilon, \{(a, \tfrac{1}{6}), (b, \tfrac{5}{6})\}) = \tfrac{1}{3} \cdot \tfrac{1}{4} = \tfrac{1}{12} \qquad \text{(join } C \text{ of } P_1 \text{ with } B_1 \text{ of } P_3\text{)}$$

$$p(R_2, \epsilon, \{(a, \tfrac{1}{3}), (b, \tfrac{2}{3})\}) = \tfrac{2}{3} \cdot \tfrac{1}{4} = \tfrac{1}{6} \qquad \text{(join } A_1 \text{ of } P_1 \text{ with } B_1 \text{ of } P_3\text{)}$$

- The root of the tree $R_3 = [\![P_2]\!] +_{\frac{1}{2}} [\![P_3]\!]$ is given by:

$$p(R_3, \epsilon, B_1) = \tfrac{1}{4} \cdot \tfrac{1}{4} + \tfrac{3}{4} \cdot \tfrac{1}{4} = \tfrac{1}{4} \qquad \text{(join } B_1 \text{ of } P_2 \text{ with } B_1 \text{ of } P_3 \text{ and } \emptyset \text{ of } P_2 \text{ with } B_1 \text{ of } P_3\text{)}$$

□

Continuations of the tree $R_1 +_p R_2$

Now we have to define the rest of the tree, that is, how we can obtain the values $p(R_1 +_p R_2, s, X)$, for each $s \neq \epsilon$. First, we must compute how the first state of the sequence s can be built from the initial states of R_1 and R_2. Depending on which of the states (the one of R_1, the one of R_2, or both) contains the first action of the sequence s, the continuation will be that of R_1, that of R_2 or a combination of both. In the latter case, continuations will be combined by using an *internal choice* in which the probability associated with the external choice and the probabilities associated with the action in the corresponding states of R_1 and R_2 take part. In conclusion, we have

$$\begin{aligned} p(R_1 +_p R_2, \langle A\, a\rangle \circ s', X) = & \sum_{A = B \cup_p C} \tfrac{p \cdot pro(a,B)}{p \cdot pro(a,B) + (1-p) \cdot pro(a,C)} \cdot p(R_1, s_B, X) \cdot p(R_2, C) \\ & + \sum_{A = B \cup_p C} \tfrac{(1-p) \cdot pro(a,C)}{p \cdot pro(a,B) + (1-p) \cdot pro(a,C)} \cdot p(R_2, s_C, X) \cdot p(R_1, B) \end{aligned}$$

where $s_B = \langle B\, a\rangle \circ s'$ and $s_C = \langle C\, a\rangle \circ s'$. The next example illustrates this definition.

Example 4.6 Consider the processes $P_1 = ((a; Q_1) +_{\frac{1}{2}} (b; Q_1')) \oplus_{\frac{1}{3}} (a; Q_2)$ and $P_2 = (b; Q_2') \oplus_{\frac{1}{4}} \mathit{Nil}$. In Example 4.5 we showed that the arcs outgoing from the root of $R_1 = [\![P_1]\!] +_{\frac{1}{2}} [\![P_2]\!]$ are labeled by the states $A_1 = \{(a, 1)\}$, $A_2 = \{(a, \frac{1}{4}), (b, \frac{3}{4})\}$, and $A_3 = \{(a, \frac{1}{2}), (b, \frac{1}{2})\}$, being the probabilities associated

with them $\frac{1}{2}$, $\frac{1}{12}$, and $\frac{5}{12}$, respectively. Let us show how the rest of the tree is defined.

$$p(R_1, \langle A_1\ a\rangle \circ s', X) = \frac{\frac{1}{2}\cdot 1}{\frac{1}{2}\cdot 1+\frac{1}{2}\cdot 0} \cdot \frac{2}{3} \cdot p(Q_2, s', X) \cdot \frac{3}{4} = \frac{1}{2} \cdot p(Q_2, s', X)$$

$$p(R_1, \langle A_2\ a\rangle \circ s', X) = \frac{\frac{1}{2}\cdot \frac{1}{2}}{\frac{1}{2}\cdot \frac{1}{2}+\frac{1}{2}\cdot 0} \cdot \frac{1}{3} \cdot p(Q_1, s', X) \cdot \frac{1}{4} = \frac{1}{12} \cdot p(Q_1, s', X)$$

$$\begin{aligned} p(R_1, \langle A_2\ b\rangle \circ s', X) &= \frac{\frac{1}{2}\cdot \frac{1}{2}}{\frac{1}{2}\cdot \frac{1}{2}+\frac{1}{2}\cdot 1} \cdot \frac{1}{3} \cdot p(Q'_1, s', X) \cdot \frac{1}{4} + \frac{\frac{1}{2}\cdot 1}{\frac{1}{2}\cdot \frac{1}{2}+\frac{1}{2}\cdot 1} \cdot \frac{1}{4} \cdot p(Q'_2, s', X) \cdot \frac{1}{3} \\ &= \frac{1}{36} \cdot p(Q'_1, s', X) + \frac{1}{18} \cdot p(Q'_2, s', X) \end{aligned}$$

$$\begin{aligned} p(R_1, \langle A_3\ a\rangle \circ s', X) &= \frac{\frac{1}{2}\cdot \frac{1}{2}}{\frac{1}{2}\cdot \frac{1}{2}+\frac{1}{2}\cdot 0} \cdot \frac{1}{3} \cdot p(Q_1, s', X) \cdot \frac{3}{4} + \frac{\frac{1}{2}\cdot 1}{\frac{1}{2}\cdot 1+\frac{1}{2}\cdot 0} \cdot \frac{2}{3} \cdot p(Q_2, s', X) \cdot \frac{1}{4} \\ &= \frac{1}{4} \cdot p(Q_1, s', X) + \frac{1}{6} \cdot p(Q_2, s', X) \end{aligned}$$

$$\begin{aligned} p(R_1, \langle A_3\ b\rangle \circ s', X) &= \frac{\frac{1}{2}\cdot \frac{1}{2}}{\frac{1}{2}\cdot \frac{1}{2}+\frac{1}{2}\cdot 0} \cdot \frac{1}{3} \cdot p(Q'_1, s', X) \cdot \frac{3}{4} + \frac{\frac{1}{2}\cdot 1}{\frac{1}{2}\cdot 0+\frac{1}{2}\cdot 1} \cdot \frac{1}{4} \cdot p(Q'_2, s', X) \cdot \frac{2}{3} \\ &= \frac{1}{4} \cdot p(Q'_1, s', X) + \frac{1}{6} \cdot p(Q'_2, s', X) \end{aligned}$$

Obviously, for any sequence beginning with another station we have that the probability of reaching any state is equal to zero. □

The next result states that the semantic operators are monotonous and continuous (the proof is given in the Appendix of the paper).

Proposition 4.7 For any $a \in Act$, the function $a;\_ :: \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ is monotonous and continuous. Moreover, for any $0 < p < 1$, the functions $\_\oplus_p\_, \_+_p\_ :: \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ are monotonous and continuous in both arguments. □

As usual when defining a denotational semantics, the meaning of recursive processes is obtained as the limit of its finite approximations. They are defined as $P_0 = \Omega, P_1 = P(\Omega), \ldots, P_n = P^n(\Omega)$. Because all of the semantic operators are continuous, this limit is the least fixed point of the equation $X = P(X)$. That is, we have $[\![recX.\ P(X)]\!] = \bigsqcup_{n=0}^{\infty} [\![P_n]\!]$.

Finally, we will show that the equivalence induced by the denotational semantics is equal to our testing equivalence. Instead of proving this result, we will show that the equivalence induced by the denotational semantics relate the same processes as the equivalence based on acceptance sets. Once we have proven this result, by using Corollary 3.21 we will obtain that both semantics, denotational and testing, are equivalent. We need the following result relating the operational behavior of a recursive process and the one of its finite approximations. The proof is easy by induction on the number of times that

recursion is *unfolded*.

Lemma 4.8 Let $P = recX.P(X)$. Then, for any sequence s and any $p \in (0,1]$ we have $P \stackrel{s}{\Longrightarrow}_p$ iff $\exists\ n \in \mathbb{N}^+ :\ P_n \stackrel{s}{\Longrightarrow}_p$. □

The next result shows the close relation between acceptance sets and acceptance trees (the proof is given in the appendix of the paper).

Theorem 4.9 For any process P and any sequence s we have

$$p([\![P]\!], s, A) = \sum_{P'} \{\!|\ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge\ S(P') = A\ |\!\}$$

□

As an immediate corollary of the previous theorem we obtain the following result.

Lemma 4.10 For any process P we have $\mathcal{A}(P,s) = \{(A_1,p_1),\ldots,(A_n,p_n)\}$ where $p_i = \frac{p([\![P]\!],s,A_i)}{p^P_s}$ for each i, and $p([\![P]\!],s,A) = 0$ for each A distinct of the A_i, where we consider $p^P_\epsilon = 1$ and $p^P_{s'\circ\langle B\, b\rangle} = p([\![P]\!], s', B)$.

Proof: Trivial from Theorem 4.9 and Definition 3.7. □

Corollary 4.11 Let P, Q be processes. We have $P \cong Q$ iff $[\![P]\!] =_{\mathbf{PAT}} [\![Q]\!]$. □

Corollary 4.12 (Full Abstraction for PAT$_{Act}$)
Let P, Q be processes. We have $P \approx Q$ iff $[\![P]\!] =_{\mathbf{PAT}} [\![Q]\!]$.

Proof: Immediate from Corollaries 3.21 and 4.11. □

5 Axiomatization of the Testing Semantics

Even though there have appeared several axiomatizations for probabilistic processes dealing with a notion of bisimulation (e.g. [23,15,3,1,35]), axiomatizations for probabilistic testing are more scarce. In addition to [30], that we follow in this part of the paper, we may mention [12,6]. In this section we will define an axiomatization inducing an equivalence relation, denoted by $\equiv$, among the terms of our language. We will also use an order relation $\sqsubseteq$ to define this equivalence relation. This system includes axioms expressing algebraic properties of the operators as well as relations among them, like distributivity. We will also present some axioms which are sound in the non-probabilistic framework but not in our setting. We will first study the language PPA$_{fin}$ which is the subset of PPA where recursive definitions are not allowed. During the rest of this section we will usually call *finite* processes

to the processes belonging to this subset of PPA. Besides, we call recursive (or infinite) processes to those processes containing occurrences of recursion. Once we have studied finite processes, we will consider the full language. So, from the logic system for finite processes, we will define a new set of axioms and rules dealing with recursive processes. In order to prove completeness of the new system (taking into account that we have included infinitary rules, and so we cannot talk about *real* completeness) we will have the usual infinitary rules for recursion given in [19]. Besides, we need to add a *technical* rule because the semantics of finite processes is given by non-compact elements of the semantic domain (we will comment more thoroughly on this rule when we present it). Soundness of rules (axioms) dealing with $\equiv$ will be shown with respect to the testing equivalence, while soundness of the ones corresponding to $\sqsubseteq$ will be shown with respect to our fully abstract denotational semantics. Although we will mix soundness (and completeness) proofs with respect to either the testing or the denotational semantics, this process is correct. First, we will prove $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ iff $\vdash P \sqsubseteq Q$. From this result, given that both $\sqsubseteq_{\mathbf{PAT}}$ and $\sqsubseteq$ are preorders, we will trivially get $[\![P]\!] =_{\mathbf{PAT}} [\![Q]\!]$ iff $\vdash P \equiv Q$. Finally, by full abstraction, we obtain the desired result $P \approx Q$ iff $\vdash P \equiv Q$.

The first axioms of our system are similar to those in [19] for must testing. They express that internal choice is idempotent, commutative and associative, while external choice is commutative and *Nil* is its identity element. Commutativity and associativity are intended up to a suitable rebalance of probabilities.

(**II**) $P \oplus_p P \equiv P$ (**CI**) $P \oplus_p Q \equiv Q \oplus_{1-p} P$

(**AI**) $P \oplus_p (Q \oplus_q R) \equiv (P \oplus_{p'} Q) \oplus_{q'} R$, where $q' = p + q - p \cdot q$ and $p' = \frac{p}{q'}$

(**CE**) $P +_p Q \equiv Q +_{1-p} P$ (**NE**) $P +_p Nil \equiv P$

Now, we present some *axioms* that are not sound in our probabilistic model, although they were in non-probabilistic testing models. First, in general, the external choice operator is not idempotent as the following example shows.

Example 5.1 Consider the processes $P = a \oplus_{\frac{1}{2}} b$ and $P' = P +_{\frac{1}{2}} P$, and the test $T = a;\omega$. On the one hand we have $pass(P\,,\,T) = \frac{1}{2}$, while on the other hand $pass(P'\,,\,T) = \frac{3}{4}$. □

This situation also appears in models dealing with replication where a choice between the same process is not necessarily equivalent to the original process. However, the result holds for stable process (the proof is easy by considering that stable processes can evolve only by applying the operational rules $(EXT4)$ and $(EXT5)$).

Lemma 5.2 Let P be a stable process and $p \in (0,1)$. Then, $P \approx P +_p P$. □

The following example shows that the external choice operator is not associative (even if we consider a *rebalance* of probabilities similar to the one that we used in the definition of axiom (**AI**)).

Example 5.3 Consider $P = a +_{\frac{1}{2}} (b +_{\frac{1}{2}} Nil)$ and $P' = (a +_{\frac{2}{3}} b) +_{\frac{3}{4}} Nil$, and the test $T = a;\omega +_{\frac{1}{2}} b; Nil$. On the one hand $pass(P\ ,\ T) = \frac{1}{2}$, while on the other hand $pass(P'\ ,\ T) = \frac{2}{3}$. This is so because $P \approx a +_{\frac{1}{2}} b$ while $P' \approx a +_{\frac{2}{3}} b$, and obviously $a +_{\frac{1}{2}} b \not\approx a +_{\frac{2}{3}} b$. □

The lack of associativity appears if any of the corresponding processes is operationally equivalent to *Nil*. This fact could create problems when defining the notion of normal form. However we can easily solve this problem because, by axiom (**NE**), we can remove all the occurrences of *Nil* in the context of a external choice. We will have a restrictive associativity for the external choice, but it will be enough for the purpose of transforming any finite process into normal form.

Lemma 5.4 Let P_1, P_2, P_3 be processes such that for any i, $P_i \longrightarrow$ (i.e. stable processes which are not operationally equivalent to *Nil*). We have $P_1 +_p (P_2 +_q P_3) \approx (P_1 +_{p'} P_2) +_{q'} P_3$, where $q' = p + q - p \cdot q$ and $p' = \frac{p}{q'}$. □

The proof of the previous result is easy just by considering that stable processes can evolve only by performing external transitions and by taking into account that $p = q' \cdot p'$, $q \cdot (1-p) = q' \cdot (1-p')$, and $(1-p) \cdot (1-q) = 1 - q'$. Next we will introduce some axioms dealing with divergence.

$$(\mathbf{D})\ \Omega \sqsubseteq P \qquad (\mathbf{DI})\ P \oplus_p \Omega \sqsubseteq P \qquad (\mathbf{DE})\ P +_p \Omega \equiv \Omega$$

Note that, in contrast with the non-probabilistic case, we have $P \oplus_p \Omega \not\equiv \Omega$. For instance, consider $P = a; Nil$ and $T = a;\omega$. We have $pass(P \oplus_p \Omega\ ,\ T) = p$ while $pass(\Omega\ ,\ T) = 0$.

Now we will consider distributive laws between the choice operators. First, the operator $+_p$ distributes over $\oplus_q$.

$$(\mathbf{DEI})\ P_1 +_p (P_2 \oplus_q P_3) \equiv (P_1 +_p P_2) \oplus_q (P_1 +_p P_3)$$

We can extend the previous axiom to deal with generalized internal choices:

$$(\mathbf{DEIG})\ P +_p (\bigoplus_{i=1}^{n} [p_i]\ P_i) \equiv \bigoplus_{i=1}^{n} [p_i]\ (P +_p P_i)$$

On the contrary, the converse distributivity does not hold in general.

Example 5.5 Let $P = a \oplus_{\frac{1}{2}} (b +_{\frac{1}{2}} c)$ and $Q = (a \oplus_{\frac{1}{2}} b) +_{\frac{1}{2}} (a \oplus_{\frac{1}{2}} c)$. We have $pass(P\ ,\ a;\omega) = \frac{1}{2}$ and $pass(Q\ ,\ a;\omega) = \frac{3}{4}$. □

As in the non-probabilistic case, in order to prove the completeness of the logic system we will define an adequate notion of normal form. Given the fact that our normal forms will contain generalized external choices (instead of binary ones) we will give two axioms for composing generalized external choices.

Let $A = \{a_1, \ldots, a_n\} \subseteq Act$ and $B = \{b_1, \ldots, b_m\} \subseteq Act$, with $A, B \neq \emptyset$. We have

$$(\mathbf{EBE}) \qquad \left(\sum_{i=1}^{n} [p_i]\, a_i; P_i\right) +_p \left(\sum_{j=1}^{m} [q_j]\, b_j; Q_j\right) \equiv \left(\sum_{k=1}^{l} [r_k]\, c_k; R_k\right)$$

where $C = \{c_1, \ldots, c_l\} = A \cup B$ and for any $1 \leq k \leq l$ we have

$$(R_k, r_k) = \begin{cases} (P_i & , p \cdot p_i &) \text{ if } c_k = a_i \in A - B \\ (Q_j & , (1-p) \cdot q_j &) \text{ if } c_k = b_j \in B - A \\ (P_i \oplus_{\frac{p \cdot p_i}{p \cdot p_i + (1-p) \cdot q_j}} Q_j & , p \cdot p_i + (1-p) \cdot q_j &) \text{ if } c_k = a_i = b_j \in A \cap B \end{cases}$$

$$(\mathbf{IBE}) \qquad \left(\sum_{i=1}^{n} [p_i]\, a_i; P_i\right) \oplus_p \left(\sum_{i=1}^{n} [p_i]\, a_i; Q_i\right) \equiv \left(\sum_{i=1}^{n} [p_i]\, a_i; (P_i \oplus_p Q_i)\right)$$

Note that the generalized generalized external choice *generalizes* the prefix operator because the process $a_1; P_1$ can be written as $\sum_{i=1}^{1} [1]\, a_i; P_i$. The next result states that, after the adequate transformations, the binary choice operator can be completely removed from processes, appearing instead generalized external choices. In this case, external choices presenting non-deterministic behavior are converted into internal choices. For instance, the process $(a; P_1) +_p (a; P_2)$ will be transformed into $a; (P_1 \oplus_p P_2)$. The proof is easy by structural induction and by using the axioms (**NE**), (**DEIG**), and (**EBE**).

Proposition 5.6 For any process $P \in \mathrm{PPA}_{fin}$ there exists $P' \in \mathrm{PPA}_{fin}$ such that $P \equiv P'$ and P' does not contain any occurrence of the binary choice operator. □

The following result (which proof is given in the Appendix) states that all the axioms previously presented are sound.

Proposition 5.7 The axioms (**II**), (**CI**), (**AI**), (**CE**), (**NE**), (**D**), (**DI**), (**DE**), (**NE**), (**DEI**), (**EBE**), and (**IBE**) are sound. □

In addition to the previous axioms, we need a set of rules indicating that the relation $\equiv$ fulfills some *good* properties. The inference rules of our logic system are given in Figure 6. Rules (**O1-3**) indicate that $\sqsubseteq$ is an order relation. Rules

(**O1**) $\frac{P \sqsubseteq Q \ \wedge \ Q \sqsubseteq P}{P \equiv Q}$	(**O2**) $\frac{P \equiv Q}{P \sqsubseteq Q, \ Q \sqsubseteq P}$	(**O3**) $\frac{P \sqsubseteq Q \ \wedge \ Q \sqsubseteq R}{P \sqsubseteq R}$
(**C1**) $\frac{P \sqsubseteq Q}{a;P \sqsubseteq a;Q}$	(**C2**) $\frac{P \sqsubseteq Q \ \wedge \ P' \sqsubseteq Q'}{P +_p P' \sqsubseteq Q +_p Q'}$	(**C3**) $\frac{P \sqsubseteq Q \ \wedge \ P' \sqsubseteq Q'}{P \oplus_p P' \sqsubseteq Q \oplus_p Q'}$
(**RE**) $\frac{}{P \equiv P}$	(**OI1**) $\frac{P \sqsubseteq Q}{P \sqsubseteq P \oplus_p Q}$	(**OI2**) $\frac{P \sqsubseteq Q}{P \oplus_p Q \sqsubseteq Q}$

Fig. 6. Inference Rules.

(**C1-3**) say that $\sqsubseteq$ is a precongruence with respect to the basic operators of the language. (**RE**) says that $\equiv$ is reflexive. Finally, (**OI1-2**) indicate that internal choice occupies an intermediate position between the corresponding processes. Soundness of (**O1-3**), (**C1-3**), and (**RE**) rules is trivial with respect to $\sqsubseteq_{\mathbf{PAT}}$ (given the fact that the latter is compositional) while soundness of (**OI1-2**) can be easily shown with respect to $\sqsubseteq_{\mathbf{PAT}}$.

Definition 5.8 Given two processes P and Q, we write $\vdash P \sqsubseteq Q$ (resp. $\vdash P \equiv Q$) if $P \sqsubseteq Q$ (resp. $P \equiv Q$) can be derived from the axioms given before and the rules given in Figure 6. □

Given that the previous axioms and rules are sound, we immediately get

Theorem 5.9 (Soundness for PPA_{fin} **)**
For any $P, \ Q \in \mathrm{PPA}_{fin}$ we have $\vdash P \sqsubseteq Q$ implies $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$. As a corollary, we also have $\vdash P \equiv Q$ implies $[\![P]\!] =_{\mathbf{PAT}} [\![Q]\!]$, and by using full abstraction of $=_{\mathbf{PAT}}$, $\vdash P \equiv Q$ implies $P \approx Q$. □

The last result indicates that if we can derive the equivalence between two *finite* processes then these two processes are testing equivalent. Next, we will prove the reciprocal result: If two finite processes are testing equivalent then the equivalence of these processes with respect to $\equiv$ can be derived from our logic system. As usually, we will consider a notion of *normal form*. Once this notion is introduced, we will have to prove that every PPA_{fin} process can be transformed into a normal form by applying the axioms and rules of our logic system. Our normal forms will be very similar to the ones given in Section 4. Specifically, they will be generalized internal choices of generalized external choices. The actions associated with the generalized external choices *prefix* normal forms. So, normal forms will be again processes having a strict alternation between generalized internal choices and generalized external choices.

Definition 5.10 *Normal forms* are those PPA_{fin} processes defined by means of the following BNF expression:

$$N ::= \bigoplus_{i=1}^{n} [p_i] \sum_{j=1}^{r_i} [p_{i,j}] \, a_{i,j} \, ; N$$

where $n \geq 0$, $\sum p_i \leq 1$, and the following restrictions hold:

- $\forall\, 1 \leq i \leq n :\ p_i > 0 \wedge r_i \geq 0 \wedge$ if $r_i > 0$ then $\sum_{j=1}^{r_i} p_{i,j} = 1 \wedge \forall\, 1 \leq j \leq r_i :\ p_{i,j} > 0$
- $\forall\, 1 \leq i \leq n :\ \forall\, 1 \leq k, l \leq r_i, k \neq l :\ a_{i,k} \neq a_{i,l}$
- $\forall\, 1 \leq u, v \leq n,\ u \neq v : \{(a_{u,j}, p_{u,j})\}_{j=1}^{r_u} \neq \{(a_{v,j}, p_{v,j})\}_{j=1}^{r_v}$

□

Note that, in contrast with [19], we do not force the continuations after the same action in different states to be equal. We will use the process Nil to denote generalized external choices over the empty set of actions (i.e. $r_i = 0$). Moreover, we will use the prefix notation, $a; N$, when the generalized external choice has a unique action (i.e. $r_i = 1$). For the sake of simplicity, we sometimes will use the following alternative notation for normal forms:

$$N ::= \bigoplus_{A \in \mathcal{A}} [p_A] \sum_{(a,p_a) \in A} [p_a]\, a; N_{a,A}$$

where $\mathcal{A}$ is a finite subset of $\mathcal{P}(Act \times (0,1])$ such that for all $A \in \mathcal{A}$, if $A \neq \emptyset$ then $\sum \{\!|\, p_a \,|\, (a, p_a) \in A \,|\!\} = 1$. The next example presents some normal forms processes as well as some processes which are not normal forms.

Example 5.11 The following processes are normal forms: $a; ((b; Nil) \oplus_{\frac{1}{2}} Nil)$, $a; Nil$, $(a; Nil) \oplus_{\frac{1}{3}} (b; Nil)$, and $(a; b; Nil) \oplus_{\frac{1}{2}} ((a; Nil) +_{\frac{1}{4}} (b; Nil))$. On the contrary, the following processes are not normal forms: $(a; Nil) \oplus_{\frac{2}{5}} (a; b; Nil)$, $(a; Nil) +_{\frac{3}{4}} (a; b; Nil)$, and $(a; Nil) +_{\frac{1}{6}} Nil$. □

After introducing normal forms, the next step is to prove their uniqueness. But this result is a trivial consequence of Theorem 3.20, because our normal forms are a particular instance of the ones given by Definition 3.10. We also have to prove that any process can be transformed into normal form (the proof is given in the Appendix).

Theorem 5.12 Let $P \in \mathrm{PPA}_{fin}$. There exists a normal form N, such that $\vdash P \equiv N$. □

Next we present a result stating that if two (semantic) processes are related by $\sqsubseteq_{\mathbf{PAT}}$, then the corresponding syntactic processes are also related by $\sqsubseteq$ (the proof is given in the Appendix).

Lemma 5.13 Let $P, Q \in \mathrm{PPA}_{fin}$. Then, $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ implies $P \sqsubseteq Q$. □

By using the previous result and the equivalence between $=_{\mathbf{PAT}}$ and $\approx$, we immediately obtain

Theorem 5.14 Let $P, Q \in \mathrm{PPA}_{fin}$. Then, $P \approx Q$ implies $\vdash P \equiv Q$. □

During the rest of this section we will extend the previous results for finite processes to deal with the full language PPA. As we did in the denotational treatment of recursive processes, we will work again with the approximation by *finite processes* of recursive processes. These finite approximations are defined as in [19].

Definition 5.15 Let P be a PPA process. For any $n \in \mathbb{N}$, we define the n-th *finite approximation* of P as $P^0 = \Omega$, and for $n \geq 0$:

- $X^{n+1} = X$, if $X \in Id$
- $Nil^{n+1} = Nil$
- $\Omega^{n+1} = \Omega$
- $(a;P)^{n+1} = a;P^{n+1}$
- $(P \oplus_p Q)^{n+1} = P^{n+1} \oplus_p Q^{n+1}$
- $(recX.P)^{n+1} = P^{n+1}\{(recX.P)^n/X\}$
- $(P +_p Q)^{n+1} = P^{n+1} +_p Q^{n+1}$

□

Note that for processes in $P \in \mathrm{PPA}_{fin}$ we have $P^n = P$, for any $n > 0$. At the syntactic level, each finite approximation is a finite process. In particular, we can use our previous study for finite processes when reasoning about finite approximations. In order to cope with recursive processes we first add the following two rules, which are equal to those in the classical testing framework.

$$(\mathbf{R1})\ \frac{}{P\{recX.P/X\} \sqsubseteq recX.P} \qquad (\mathbf{R2})\ \frac{\forall\, n \in \mathbb{N} :\ P^n \sqsubseteq R}{P \sqsubseteq R}$$

Soundness proofs easily follow from the definition of the denotational semantics of recursive processes. Specifically, soundness of (**R1**) is trivial since $[\![P\{recX.P/X\}]\!] = \bigsqcup_{n=1}^{\infty} [\![P^n]\!]$ while (**R2**) is sound because we are working within a *cpo*, and so $[\![P]\!]$ is the least upper bound of $\{[\![P^i]\!]\}_{i=0}^{\infty}$. As we said in the introduction of the paper, we need to add another rule because of *technical* reasons.

$$(\mathbf{R3})\frac{\forall\, n \in \mathbb{N} :\ P \oplus_{\frac{n-1}{n}} \Omega \sqsubseteq R}{P \sqsubseteq R}$$

Lemma 5.16 The rule (**R3**) is sound.

Proof: Let us suppose that for any $n \in \mathbb{N}$ we have $[\![P \oplus_{\frac{n-1}{n}} \Omega]\!] \sqsubseteq_{\mathbf{PAT}} [\![R]\!]$. In other words, for any $n \in \mathbb{N}$, any sequence s, and any state A we have that $p([\![P \oplus_{\frac{n-1}{n}} \Omega]\!], s, A) \leq p([\![R]\!], s, A)$. From the definition of the internal choice semantic function we obtain that for any sequence s and for any state A we have $p([\![P \oplus_{\frac{n-1}{n}} \Omega]\!], s, A) = \frac{n-1}{n} \cdot p([\![P]\!], s, A) + \frac{1}{n} \cdot p([\![\Omega]\!], s, A) = \frac{n-1}{n} \cdot p([\![P]\!], s, A)$. If we consider the two previous expressions, we have that for any sequence of stations s and any state A: $p([\![P]\!], s, A) = \lim_{n\to\infty} \frac{n-1}{n} \cdot p([\![P]\!], s, A) \leq p([\![R]\!], s, A)$, which implies $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![R]\!]$. □

We can extend the soundness result given in Theorem 5.9.

Theorem 5.17 Let $P, Q \in \text{PPA}$. We have $\vdash P \sqsubseteq Q$ implies $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$. □

Now we will prove completeness of the axiomatization. First, we present a result for recursive processes (whose proof is exactly as in [19]) and then we extend Lemma 5.13 for the case when one of the processes is not finite.

Lemma 5.18 Let $P \in \text{PPA}$. For any approximation P^n we have $\vdash P^n \sqsubseteq P$. □

In Lemma 5.13 we showed that $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ implies $P \sqsubseteq Q$ for any pair of finite processes P and Q. Next we deal with the cases when (at least) one of the processes contains recursive definitions.

Lemma 5.19 Let $P \in \text{PPA}$ be a non-finite process and $Q \in \text{PPA}_{fin}$. Then, $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ implies $P \sqsubseteq Q$.

Proof: By definition, the finite approximations of P form a chain, where the value $[\![P]\!]$ is the least upper bound of the values $[\![P^n]\!]$. That is, we have that $[\![P^0]\!] \sqsubseteq_{\mathbf{PAT}} [\![P^1]\!] \sqsubseteq_{\mathbf{PAT}} \cdots \sqsubseteq_{\mathbf{PAT}} [\![P^n]\!] \cdots \sqsubseteq_{\mathbf{PAT}} [\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$. Given the fact that the processes P^n and Q are finite, we can apply Lemma 5.13 to deduce $P^n \sqsubseteq Q$, for any n. Finally, by applying (**R2**), we have $P \sqsubseteq Q$. □

Now, let us consider the case where P is finite but Q is not. Given that the usual way to assign semantics to recursive processes is by means of their finite approximations, the most straight way for proving $P \sqsubseteq Q$ would be to guarantee that there exists m such that the *m-th* finite approximation of Q fulfills $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q^m]\!]$. Then, given the fact that P and Q^m are finite, we could apply Lemma 5.13, deducing $P \sqsubseteq Q^m$. Besides, we have $Q^m \sqsubseteq Q$ and so, by applying (**O3**), we would obtain $P \sqsubseteq Q$. If finite processes were mapped into compact (also called finite) elements of the semantic domain then the existence of such an m would be guaranteed. This is so because if R is a compact element and $R \sqsubseteq_{\mathbf{PAT}} \sqcup R^n$ then there exists R^i such that $R \sqsubseteq_{\mathbf{PAT}} R^i$. Unfortunately, this is not the case as the following example shows.

Example 5.20 Consider $P = recX.((a; Nil) \oplus_{\frac{1}{2}} X)$, and $Q = a; Nil$. It is easy to check that the finite approximations of P are given by $P^n = (a; Nil) \oplus_{1-\frac{1}{2^n}} \Omega$. By definition we have $[\![P]\!] = \sqcup [\![P^n]\!]$. Thus, we trivially get $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} \sqcup [\![P^n]\!]$. Besides, $[\![P]\!]$ *describes* a syntactic finite process because $[\![P]\!] =_{\mathbf{PAT}} [\![Q]\!]$. So, we should be able to conclude $P \equiv Q$. By the previous lemma we have $P \sqsubseteq Q$ but there does not exist m such that $[\![Q]\!] \sqsubseteq_{\mathbf{PAT}} [\![P^m]\!]$. If there would exist such an n then we get $1 = p([\![Q]\!], \epsilon, \{(a, 1)\}) \leq p([\![P^m]\!], \epsilon, \{(a, 1)\}) = 1 - \frac{1}{2^m}$, which is not possible. So, we have found a finite (syntactic) process, $a; Nil$ whose semantics is the least upper bound of the infinite nontrivial chain $\{[\![P^n]\!]\}_{n=1}^{\infty}$. □

The previous example shows that, in general, we must use another way in order to deduce $P \sqsubseteq Q$ from $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$. This is the reason why the rule (**R3**) was included in our logic system. This is an important difference with respect to [19] where finite processes are mapped into compact elements. Note that if we delete probabilities, the previous example is not correct in the classical testing theory. This is so because Ω is a *zero* of the non-probabilistic internal choice operator. That is, in the non-probabilistic setting all the processes P^n would be equivalent to Ω. Thus, a non-probabilistic version of (**R3**) is not sound for non-probabilistic testing. Let us remark that the only compact element of the semantic domain is the one corresponding to divergence. Note that for any process P (semantically) different from Ω we can always construct a succession, for instance $P^n = P \oplus_{\frac{n}{n+1}} \Omega$, such that P is *lower* than the limit (actually $[\![P]\!] = \sqcup [\![P^n]\!]$) while for any n we have $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![P^n]\!]$ does not hold. The proof of the following result (which can be found in the Appendix) shows how to deal with this situation.

Lemma 5.21 Let $P \in$ PPA be a finite process and $Q \in$ PPA be a recursive one. Then, $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ implies $P \sqsubseteq Q$. □

Theorem 5.22 Let $P, Q \in$ PPA. Then, $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ implies $P \sqsubseteq Q$.

Proof: If either P or Q is finite then the result has been proven in the previous lemmas. If both of them are infinite then, by Lemma 5.21, we have $P^n \sqsubseteq Q$ for any finite approximation P^n of P. So, by applying rule (**R2**), we conclude $P \sqsubseteq Q$. □

Corollary 5.23 Let $P, Q \in$ PPA. Then, $[\![P]\!] =_{\mathbf{PAT}} [\![Q]\!]$ implies $\vdash P \equiv Q$. □

The proof of the following result is immediate from Corollaries 4.12 and 5.23.

Corollary 5.24 Let $P, Q \in$ PPA. Then, $P \approx Q$ implies $\vdash P \equiv Q$. □

Finally, we obtain the desired result.

Corollary 5.25 Let $P, Q \in$ PPA. We have $P \approx Q$ iff $\vdash P \equiv Q$. □

We finish this section by showing an interesting alternative to the inclusion of rule (**R3**). As suggested in [11], we could avoid this rule just by considering a different definition of the finite approximations of a process. These new approximations would explicitly include the information contained in that rule. Specifically, the formal definition of the new finite approximations would be $P'_0 = \Omega$ and $P'_{n+1} = P^n \oplus_{\frac{n}{n+1}} \Omega$, where the processes P^n are the *old* finite approximations given by Definition 5.15. Using this alternative definition, the information given by the rules (**R2**) and (**R3**) would be contained in the rule (**R2**). In this case the rule (**R3**) would be redundant and so it could be omitted in the axiomatization.

$$\frac{P >\!\!\longrightarrow_p P' \ \wedge\ \mathtt{stable}(Q)}{P \parallel_A^{p_1} Q >\!\!\longrightarrow_p P' \parallel_A^{p_1} Q} \qquad \frac{Q >\!\!\longrightarrow_p Q' \ \wedge\ \mathtt{stable}(P)}{P \parallel_A^{p_1} Q >\!\!\longrightarrow_p P \parallel_A^{p_1} Q'} \qquad \frac{P >\!\!\longrightarrow_p P' \ \wedge\ Q >\!\!\longrightarrow_q Q'}{P \parallel_A^{p_1} Q >\!\!\longrightarrow_{p\cdot q} P' \parallel_A^{p_1} Q'}$$

$$\frac{P \xrightarrow{b}_p P' \ \wedge\ \mathtt{stable}(Q) \ \wedge\ b \notin A}{P \parallel_A^{p_1} Q \xrightarrow{b}_{p_1 \cdot r_1} P' \parallel_A^{p_1} Q} \qquad \frac{Q \xrightarrow{b}_p Q' \ \wedge\ \mathtt{stable}(P) \ \wedge\ b \notin A}{P \parallel_A^{p_1} Q \xrightarrow{b}_{(1-p_1)\cdot r_1} P \parallel_A^{p_1} Q'}$$

$$\frac{P \xrightarrow{a}_p P' \wedge Q \xrightarrow{a}_q Q' \wedge a \in A}{P \parallel_A^{p_1} Q \xrightarrow{a}_{r_2} P' \parallel_A^{p_1} Q'}$$

where $r_1 = \frac{p}{\mu(P,Q,A,p_1)}$ and $r_2 = \frac{p\cdot q}{\mu(P,Q,A,p_1)}$.

Fig. 7. Rules for the operator $\parallel_A^{p_1}$.

6 Extensions of the Language

In this section we discuss the inclusion of new operators in our language. Specifically, a *parallel composition* operator and a *hiding* operator. Regarding the parallel operator, there is no clear agreement about what is the *appropriate* definition in a probabilistic setting (see [13] for a discussion on the topic). We will consider a simple parallel operator with two parameters: A synchronization set and a probability. The probability is used to assign more *weight* to one of the components when performing interleaving actions. Unfortunately, the simplicity of our operator implies that it does not fulfill the good properties presented in [13]: Our operator is neither *respectful* nor *stochastic*.

The operational semantics of this operator is given in Figure 7. The first three rules are similar to those for the external choice: Internal transitions are performed first. The next two rules consider *interleaving* actions. The last rule deals with *synchronization* actions. Note that the last three rules are applied only if both processes are stable. We have a *normalization factor*. This function is similar to the one considered for the composition of processes and tests. It is formally defined as:

$$\begin{aligned}\mu(P,Q,A,p_1) = {} & \textstyle\sum_{a\in A} \{\!|\, p\cdot q \mid \exists P', Q' :\ P \xrightarrow{a}_p P' \wedge Q \xrightarrow{a}_q Q' \,|\!\} \\ & + p_1 \cdot \textstyle\sum_{a\notin A} \{\!|\, p \mid \exists P' :\ P \xrightarrow{a}_p P' \,|\!\} \\ & + (1-p_1) \cdot \textstyle\sum_{a\notin A} \{\!|\, p \mid \exists Q' :\ Q \xrightarrow{a}_p Q' \,|\!\}\end{aligned}$$

After extending the operational semantics, we have that the definition of both the testing equivalence and the alternative characterization based on acceptance sets may also deal with processes having occurrences of the parallel operator. Next we define the semantic function associated with the operator $\parallel_A^p$. This allows to add this operator to the semantic treatment described in Section 4. The function $\_ \parallel_A^p \_ :: \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$, for

$p \in (0,1)$ and $A \subseteq Act$, returns an acceptance tree representing the *parallel composition synchronizing in A* of the corresponding trees, *with respect to the probability p*. We will give an auxiliary function which joins two states according to a synchronization set and a probability. The idea is similar to Definition 4.4 but the definition is more involved.

Definition 6.1 Let X, Y be states, $A \subseteq Act$ and $p \in (0,1)$. We define the *union* of the states X and Y with *associated probability* p and *synchronization set* A as follows:

$$X \parallel_A^p Y = \{(a, p_a) \mid a \in (Act(X) \cap Act(Y) \cap A) \cup (Act(X) - A) \cup (Act(Y) - A)\}$$

where the probability p_a is given by

$$p_a = \begin{cases} \dfrac{p \cdot pro(a,X) + (1-p) \cdot pro(a,Y)}{\mu(X,Y,A,p)} & \text{if } a \notin A \\ \\ \dfrac{pro(a,X) \cdot pro(a,Y)}{\mu(X,Y,A,p)} & \text{if } a \in A \end{cases}$$

and $\mu(X,Y,A,p)$ is given by:

$$\begin{aligned} \mu(X,Y,A,p) &= \textstyle\sum_{a \in A} \{\!\{\, pro(a,X) \cdot pro(a,Y) \,\}\!\} \\ &\quad + p \cdot \textstyle\sum_{a \notin A} \{\!\{\, pro(a,X) \,\}\!\} + (1-p) \cdot \sum_{a \notin A} \{\!\{\, pro(a,Y) \,\}\!\} \end{aligned}$$

□

Note the overloading of the symbols $\parallel_A^p$, which is used both for denoting the parallel composition of processes and the union of states, and μ, which denotes both the normalization factor for syntactic processes and the one for *normalizing* the union of states. As we did for the external choice, we must distinguish between the root of the new tree and the continuations after the root. In order to define the root of the new tree we consider the union, using the function $\parallel_A^p$, of the initial states of both trees.

$$p(R_1 \parallel_A^p R_2, \epsilon, X) = \sum_{X = B \parallel_A^p C} p(R_1, \epsilon, B) \cdot p(R_2, \epsilon, C)$$

That is, there is an outgoing arc labelled by X from the root of the new tree iff there exist an outgoing arc from the root of the tree R_1, labelled by a state B, and an outgoing arc from the root of the tree R_2, labelled by the state C, such that $X = B \parallel_A^p C$. We have that this semantic function is strict in both arguments, that is, $[\![P \parallel_A^p \Omega]\!] = [\![\Omega \parallel_A^p P]\!] = [\![\Omega]\!]$, for any $A \subseteq Act$ and $0 < p < 1$. If the synchronization set is empty then *Nil* is an identity element: $[\![P \parallel_\emptyset^p Nil]\!] = [\![Nil \parallel_\emptyset^p P]\!] = [\![P]\!]$, for any $0 < p < 1$.

Next we define the *rest* of the tree, that is, how to compute $p(R_1 \parallel_A^p R_2, s, X)$ from the trees R_1 and R_2. The idea is similar to the one for the external

$$\textbf{(CP)}\quad P \parallel_A^p Q \equiv Q \parallel_A^{1-p} P \qquad \textbf{(DPIG)}\ P \parallel_A^p (\bigoplus_{i=1}^{n}[p_i]\ P_i) \equiv \bigoplus_{i=1}^{n}[p_i]\ (P \parallel_A^p P_i)$$

$$\textbf{(EP)}\quad (\sum_{i=1}^{n}[p_i]\ a_i; P_i) \parallel_X^p (\sum_{j=1}^{m}[q_j]\ b_j; Q_j) \equiv \sum_{k=1}^{l}[\tfrac{r_k}{\mu(P,Q,X,p)}]\ c_k; R_k$$

$$\textbf{(EPN)}\ (\sum_{i=1}^{n}[p_i]\ a_i; P_i) \parallel_X^p \mathit{Nil} \equiv \sum_{a_k \in A - X}[\tfrac{p_k}{\mu(P,\mathit{Nil},X,p)}]\ a_k; P_k$$

$$\textbf{(DP)}\quad P \parallel_A^p \Omega \equiv \Omega \qquad \textbf{(C4)}\ \frac{P \sqsubset Q\ \wedge\ P' \sqsubset Q'}{P \parallel_A^p P' \sqsubseteq Q \parallel_A^p Q'}$$

Fig. 8. Extension of the axiomatization.

choice operator. In order to define *continuations* we use the semantic function $R/(A,a)$ (see Definition 4.1). Formally,

$$p(R_1 \parallel_A^p R_2, \langle B\, b\rangle \circ s', X) = \sum_{B=C \parallel_A^p D} p(R_1, \epsilon, C) \cdot p(R_2, \epsilon, D) \cdot$$

$$\begin{cases} q_1 \cdot p(R_1/(C,b) \parallel_A^p R_2, s', X) & \\ + \ q_2 \cdot p(R_1 \parallel_A^p R_2/(D,b), s', X) & \text{if } b \in (Act(C) - A) \cup (Act(D) - A) \\ p(R_1/(C,b) \parallel_A^p R_2/(D,b), s', X) & \text{if } b \in A \cap Act(C) \cap Act(D) \\ 0 & \text{otherwise} \end{cases}$$

where $q_1 = \frac{p \cdot pro(b,C)}{p \cdot pro(b,C) + (1-p) \cdot pro(b,D)}$ and $q_2 = \frac{(1-p) \cdot pro(b,D)}{p \cdot pro(b,C) + (1-p) \cdot pro(b,D)}$.

Now, we will show monotony and continuity of this function (the proof is essentially as the one for external choice and we omit it). First, we need the forthcoming Lemma 6.3 (the proof is immediate). This result indicates that the operator $/(A,a)$ is *somehow* monotonous. Let us remark that this operator is not monotonous in general, as the following example shows.

Example 6.2 Consider the following processes: $P_1 = (a; (b \oplus_{\frac{1}{3}} \Omega)) \oplus_{\frac{1}{3}} \Omega$ and $P_2 = (a; (b \oplus_{\frac{1}{4}} \Omega)) \oplus_{\frac{1}{2}} \Omega$. It is easy to check that $[\![P_1]\!] \sqsubseteq_{\mathbf{PAT}} [\![P_2]\!]$. Nevertheless, $p([\![P_1]\!]/(\{(a,1)\},a), \epsilon, \{(b,1)\}) = \frac{1}{3} > p([\![P_2]\!]/(\{(a,1)\},a), \epsilon, \{(b,1)\}) = \frac{1}{4}$. □

Lemma 6.3 Let $R_1, R_2 \in \mathbf{PAT}_{Act}$ $(R_1 \sqsubseteq_{\mathbf{PAT}} R_2)$. Then, for any state A such that $p(R_1, A) > 0$, any action $a \in Act(A)$, any sequence s, and any state X, we have $p(R_1, \epsilon, A) \cdot p(R_1/(A,a), s, X) \leq p(R_2, \epsilon, A) \cdot p(R_2/(A,a), s, X)$. □

Proposition 6.4 The functions $\_ \parallel_A^p \_ :: \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ are monotonous and continuous in both arguments, for any $0 < p < 1$ and any $A \subseteq Act$. □

Next we will extend the axiomatization given in the previous section. As it is the case in *interleaving* models of non-probabilistic processes, the parallel operator can be considered as derived from the rest of operators. This notion of *derivation* allows us to transform the application of the parallel operator on two normal forms into a *head normal form* (that is, there are no occurrences of the parallel operator in the *head* of the expression). So, by applying reiteratively these axioms, we can completely remove the parallel operator if it does not appear within the scope of a recursion. If it appears in such a scope, its occurrences can be *sinked*. Our axioms indicate that the parallel operator is commutative and distributes over the internal choice. We have an *expansion law* similar to the one appearing in non-probabilistic process algebras. We also have an axiom indicating that the parallel operator is strict. Finally, we have the usual congruence rule. These axioms and the rule appear in Figure 8. The proofs of soundness for axioms (**CP**) and (**DPIG**) are easy. Axiom (**EPN**) is a particular case of (**EP**). The proofs for (**DP**) and (**C4**) are trivial (in the last case with respect to $\sqsubseteq_{\mathbf{PAT}}$). Next, we formulate the expansion law (the proof of soundness is given in the Appendix of the paper).

• Let $A = \{a_1, \ldots, a_n\} \subseteq Act$ and $B = \{b_1, \ldots, b_m\} \subseteq Act$. If we consider the processes $P = \sum_{i=1}^{n} [p_i]\, a_i; P_i$ and $Q = \sum_{j=1}^{m} [q_j]\, b_j; Q_j$ then we have

$$(\mathbf{EP}) \qquad P \parallel_X^p Q \equiv R$$

where $R = \sum_{k=1}^{l} [\frac{r_k}{\mu(P,Q,X,p)}]\, c_k; R_k$, $C = \{c_1, \ldots, c_l\} = (A \cup B) - X \cup (A \cap B \cap X)$,

$$r_k = \begin{cases} p_i \cdot q_j & \text{if } c_k = a_i = b_j \in X \\ p \cdot p_i & \text{if } c_k = a_i \in (A - B) - X \\ (1-p) \cdot q_j & \text{if } c_k = b_j \in (B - A) - X \\ p \cdot p_i + (1-p) \cdot q_j & \text{if } c_k = a_i = b_j \in (A \cap B) - X \end{cases}$$

$$R_k = \begin{cases} P_i \parallel_X^p Q_j & \text{if } c_k = a_i = b_j \in X \\ P_i \parallel_X^p Q & \text{if } c_k = a_i \in (A - B) - X \\ P \parallel_X^p Q_j & \text{if } c_k = b_j \in (B - A) - X \\ (P_i \parallel_X^p Q) \oplus_{\frac{p \cdot p_i}{p \cdot p_i + (1-p) \cdot q_j}} (P \parallel_X^p Q_j) & \text{if } c_k = a_i = b_j \in (A \cap B) - X \end{cases}$$

Proposition 6.5 The axiom (**EP**) is sound. □

The inclusion of a *hiding* operator in our language is not so easy. Even though there are proposals for such an operator in probabilistic process algebras (e.g. [7,5,2]) we cannot reuse them in our setting. The problem is that in order

to include hiding as a derived operator, as we did for the parallel operator, we should have a *distributivity* axiom similar to that in classical must testing: $((a;c)+b)\backslash a \equiv c \oplus (c+b)$. If we do an interpretation of hidden/internal actions similar to that in other probabilistic models, then the probabilistic extension of such an axiom is not sound in PPA. Consider the processes $P = (a;c) +_{\frac{1}{2}} b$, and $P' = P\backslash a$. Using a syntax *a la* CCS, we would have that P' should be a process as $(\tau;c) +_{\frac{1}{2}} b$. We will show that there do not exist r and s, with $0 < r, s < 1$, such that $P'' = c \oplus_r (c +_s b) \approx P'$. If we compose the process P' and the test $T_1 = b;\omega$, we should obtain $pass(P' , T_1) = \frac{1}{2}$, as it is the case in other probabilistic testing models with internal actions (e.g. [9]). So, $r = \frac{1}{2}$. Consider now the test $T_2 = b +_{\frac{1}{2}} (c;\omega)$. We have $pass(P'' , T_2) = \frac{1}{2} + \frac{1}{2} \cdot s$. Once again, if we do an intuitive interpretation of hiding, we would have $pass(P' , T_2) = \frac{2}{3}$, and so $s = \frac{1}{3}$. Finally, consider the test $T_3 = b +_{\frac{2}{3}} (c;\omega)$. On the one hand we have $pass(P' , T_3) = \frac{3}{4}$, while on the other hand we obtain $pass(P'' , T_3) = \frac{3}{5}$. So, we deduce that the processes P' and P'' cannot be testing equivalent.

We obtain a very interesting result if we use *prenormalization factors* in the previous reasoning. In few words, a prenormalization factor redistributes probabilities among the *available* actions of both sides of the parallel (we omit the formal definition). For example, using a prenormalization policy, the process $a +_{\frac{1}{8}} b \;||^{\frac{1}{2}}_{\{a,b\}}\; a +_{\frac{1}{2}} c$ has the transitions $\xrightarrow{a}_{\frac{1}{2}}$ and $\xrightarrow{c}_{\frac{1}{2}}$. In this case, the probability with which P' would pass the test T_2 is given by $pass(P' , T_2) = \frac{1}{2}$. This happens because c, offered by the test, cannot be immediately performed by P'. So, the probability associated with c in the test must be *transferred* to b in the first step of the composition. So, in this first step the situation is similar to consider that the test offers b with probability 1. But this result leads s to be equal to 0 and, taking into account that in our model probabilities belong to the interval $(0, 1)$, we have that this value is not valid. We think that this last result, apparently strange, can help to include a hiding operator in our setting. Specifically, P' should be equivalent to $P'' = c \oplus_{\frac{1}{2}} (c +_0 b)$, where $+_0$ denotes a (right hand side) priority operator. Intuitively, the process $(c +_0 b)$ behaves in the following way: If the environment offers either b or both b and c then b is performed with probability 1; if the environment offers only c then this action is performed with probability 1. Unfortunately, our preliminary studies showed that the inclusion of priorities extremely complicates our model. For example, in addition to the current six rules, the operational semantics for the parallel operator had six more rules dealing with the different cases in which both processes perform priority transitions.

Another possibility consists in including a *primitive* hiding operator, that is, it cannot be derived from the rest of the operators. This solution complicates too much the semantics of our language. In particular, normal forms would be more complex because the hide operator must be explicitly considered.

7 Conclusions and Future Work

In this paper we have introduced a complete testing theory for probabilistic processes. We have followed the presentation given in [19]. First, we have considered a language featuring two probabilistic choice operators and recursion. We have defined a testing equivalence for this language and we have studied alternative characterizations of this semantics. We have also shown how our language can be extended with a parallel operator and we have argued that any attempt to include a hiding operator would lead either to consider this operator as primitive or to add priorities into the model. Regarding future work, our underlying probabilistic model is similar to that of discrete time Markov chains (if we *forget* actions). We are currently working on the extension of our results to the semi-Markov processes setting. Specifically, [26] represents the first step towards the definition of a complete testing theory of stochastic processes with general distributions. Given the fact that that language combines probabilistic choices with probability distributions (inducing delays), we are trying to extend the results appearing in this paper to that model.

Acknowledgements. I would like to thank the referees of this paper for the thorough reading and the useful remarks.

References

[1] S. Andova. Process algebra with probabilistic choice. In *5th AMAST Workshop on Real-Time and Probabilistic Systems, LNCS 1601*, pages 111–130. Springer, 1999.

[2] S. Andova and J.C.M. Baeten. Abstraction in probabilistic process algebra. In *TACAS 2001, LNCS 2031*, pages 204–219. Springer, 2001.

[3] J.C.M. Baeten, J.A. Bergstra, and S.A. Smolka. Axiomatizing probabilistic processes: ACP with generative probabilities. *Information and Computation*, 121(2):234–255, 1995.

[4] J.A. Bergstra, A. Ponse, and S.A. Smolka, editors. *Handbook of Process Algebra*. North Holland, 2001.

[5] M. Bravetti and A. Aldini. Expressing processes with different action durations through probabilities. In *PAPM-PROBMIV 2001, LNCS 2165*, pages 168–183. Springer, 2001.

[6] D. Cazorla. *PNAL: Un modelo algebraico para procesos probabilísticos y no deterministas*. PhD thesis, Universidad de Castilla-La Mancha, 2001.

[7] D. Cazorla, F. Cuartero, V. Valero, and F.L. Pelayo. A process algebra for probabilistic and nondeterministic processes. *Information Processing Letters*, 80:15–23, 2001.

[8] I. Christoff. Testing equivalences and fully abstract models for probabilistic processes. In *CONCUR'90, LNCS 458*, pages 126–140. Springer, 1990.

[9] R. Cleaveland, Z. Dayar, S.A. Smolka, and S. Yuen. Testing preorders for probabilistic processes. *Information and Computation*, 154(2):93–148, 1999.

[10] R. Cleaveland, G. Lüttgen, and V. Natarajan. Priority in process algebra. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of process algebra*, chapter 12. North Holland, 2001.

[11] F. Cuartero. *CSP probabilístico (PCSP). Un modelo probabilístico de procesos concurrentes*. PhD thesis, Universidad Complutense de Madrid, 1993.

[12] F. Cuartero, D. de Frutos, and V. Valero. A sound and complete proof system for probabilistic processes. In *4th International AMAST Workshop on Real-Time Systems, Concurrent and Distributed Software, LNCS 1231*, pages 340–352. Springer, 1997.

[13] P.R. D'Argenio, H. Hermanns, and J.-P. Katoen. On generative parallel composition. In *PROBMIV'98, Electronics Notes in Theoretical Computer Science 22*. Elsevier, 1999.

[14] R. de Nicola and M. Hennessy. CCS without τ's. In *TAPSOFT'87, LNCS 249*, pages 138–152. Springer, 1987.

[15] R. van Glabbeek, S.A. Smolka, and B. Steffen. Reactive, generative and stratified models of probabilistic processes. *Information and Computation*, 121(1):59–80, 1995.

[16] C. Gregorio and M. Núñez. Denotational semantics for probabilistic refusal testing. In *PROBMIV'98, Electronics Notes in Theoretical Computer Science 22*. Elsevier, 1999.

[17] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6:512–535, 1994.

[18] M. Hennessy. Acceptance trees. *Journal of the ACM*, 32(4):896–928, 1985.

[19] M. Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.

[20] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[21] B. Jonsson and W. Yi. Compositional testing preorders for probabilistic processes. In *10th IEEE Symposium on Logic In Computer Science*, pages 431–443. IEEE-CS Press, 1995.

[22] B. Jonsson, W. Yi, and K.G. Larsen. Probabilistic extensions of process algebras. In J.A. Bergstra, A. Ponse, and S.A. Smolka, editors, *Handbook of process algebra*, chapter 11. North Holland, 2001.

[23] C.-C. Jou and S.A. Smolka. Equivalences, congruences and complete axiomatizations for probabilistic processes. In *CONCUR'90, LNCS 458*, pages 367–383. Springer, 1990.

[24] M. Kwiatkowska and G.J. Norman. A testing equivalence for reactive probabilistic processes. In *EXPRESS'98, Electronic Notes in Theoretical Computer Science 16*. Elsevier, 1998.

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

[26] N. López and M. Núñez. A testing theory for generally distributed stochastic processes. In *CONCUR 2001, LNCS 2154*, pages 321–335. Springer, 2001.

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

[28] K. Narayan Kumar, R. Cleaveland, and S.A. Smolka. Infinite probabilistic and nonprobabilistic testing. In *18th Conference on Foundations of Software Technology and Theoretical Computer Science, LNCS 1530*, pages 209–220. Springer, 1998.

[29] R. de Nicola and M.C.B. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[30] M. Núñez. An axiomatization of probabilistic testing. In *5th AMAST Workshop on Real-Time and Probabilistic Systems, LNCS 1601*, pages 130–150. Springer, 1999.

[31] M. Núñez and D. de Frutos. Testing semantics for probabilistic LOTOS. In *Formal Description Techniques VIII*, pages 365–380. Chapman & Hall, 1995.

[32] M. Núñez, D. de Frutos, and L. Llana. Acceptance trees for probabilistic processes. In *CONCUR'95, LNCS 962*, pages 249–263. Springer, 1995.

[33] M. Núñez and D. Rupérez. Fair testing through probabilistic testing. In *Formal Description Techniques for Distributed Systems and Communication Protocols (XII), and Protocol Specification, Testing, and Verification (XIX)*, pages 135–150. Kluwer Academic Publishers, 1999.

[34] R. Segala. Testing probabilistic automata. In *CONCUR'96, LNCS 1119*, pages 299–314. Springer, 1996.

[35] E.W. Stark and S.A. Smolka. A complete axiom system for finite-state probabilistic processes. In *Proof, Language and Interaction: Essays in Honour of Robin Milner*. MIT Press, 2000.

[36] S.-H. Wu, S.A. Smolka, and E.W. Stark. Composition and behaviors of probabilistic I/O automata. *Theoretical Computer Science*, 176(1-2):1–37, 1997.

[37] W. Yi and K.G. Larsen. Testing probabilistic and nondeterministic processes. In *Protocol Specification, Testing and Verification XII*, pages 47–61. North Holland, 1992.

Appendix: Proofs of the Main Results

Lemma 2.10 $P \approx Q$ iff for all *finite* test T we have $pass(P, T) = pass(Q, T)$.

Proof: The left to right implication is obvious. The proof of the right to left implication is done by contradiction. We suppose that P and Q pass any finite test with the same probability, but that there exists a recursive test T such that $p = pass(P, T)$, $q = pass(Q, T)$, and $p \neq q$. So,

$$p = \lim_{n\to\infty} \sum_{S \in Su(P,T)} \{Pr(S) \,|\, \texttt{length}(S) < n\} \neq \lim_{n\to\infty} \sum_{S \in Su(Q,T)} \{Pr(S) \,|\, \texttt{length}(S) < n\} = q$$

Let $\texttt{actions}(S)$ be equal to the number of transitions appearing in S labeled by an action belonging to the set $Act \cup \{\omega\}$. If we have the previous inequality it is obvious that we also have

$$p = \lim_{n\to\infty} \sum_{S \in Su(P,T)} \{Pr(S) \,|\, \texttt{actions}(S) < n\} \neq \lim_{n\to\infty} \sum_{S \in Su(Q,T)} \{Pr(S) \,|\, \texttt{actions}(S) < n\} = q$$

If these two limits are different, then there exist n_0 such that for all $n \geq n_0$ we have

$$\sum_{S \in Su(P,T)} \{Pr(S) \,|\, \texttt{actions}(S) < n\} \neq \sum_{S \in Su(Q,T)} \{Pr(S) \,|\, \texttt{actions}(S) < n\}$$

Let us consider the finite test T' resulting from T by unwinding n_0 times every occurrence of recursion in T, and then replacing any occurrence of recursion by Nil. We have that T' can perform the same sequences of transitions S such that $\texttt{actions}(S) \leq n_0$ as T.[11] So, $P \mid T'$ (resp. $Q \mid T'$) can perform the same computations as $P \mid T$ such that $\texttt{actions}(S) \leq n_0$ (resp. $Q \mid T$), and so we get, by the last inequality, that the probabilities with which P and Q pass T' are different (note that T' is finite).

Lemma 3.18 Let f and f' be two rational functions of $n \geq 0$ variables $x_1, x_2 \ldots x_n$, defined as follows:

$$f = \sum_{i \in I} \frac{c_i}{1 + \sum_{j=1}^{n} d_{j,i} \cdot x_j} \qquad f' = \sum_{i' \in I'} \frac{c'_{i'}}{1 + \sum_{j=1}^{n} d'_{j,i'} \cdot x_j}$$

where I, I' are finite sets of indices; $c_i, c'_{i'} > 0$; for each distinct $r, s \in I$, the tuples $(d_{1,r}, d_{2,r}, \ldots, d_{n,r})$ and $(d_{1,s}, d_{2,s}, \ldots, d_{n,s})$ are distinct; and for each distinct $r, s \in I'$, the tuples $(d'_{1,r}, d'_{2,r}, \ldots, d'_{n,r})$ and $(d'_{1,s}, d'_{2,s}, \ldots, d'_{n,s})$ are distinct. If $f = f'$, then there exists a bijection $h : I \longrightarrow I'$ such that $d_{j,i} = d'_{j,h(i)}$ and $c_i = c'_{h(i)}$ for all $i \in I$ and $1 \leq j \leq n$. So, the expressions defining f and f' are identical up to commutativity.

Proof: By now, let us assume that there does not exist a null tuple in the definition of f, that is, $\forall i \in I \; \exists j \in \{1 \ldots n\} : \; d_{j,i} \neq 0$.

Let $p_i = 1 + \sum_{j=1}^{n} d_{j,i} \cdot x_j$, $p'_{i'} = 1 + \sum_{j=1}^{n} d'_{j,i'} \cdot x_j$, and let $P = \prod_{i \in I} p_i \cdot \prod_{i' \in I'} p'_{i'}$. We

[11] Sequences corresponding to T may have additional internal transitions corresponding to unwinding recursions. However, these transitions have probability 1.

have, $f \cdot P = f' \cdot P$, that is:

$$\sum_{i \in I} c_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k \cdot \prod_{k' \in I'} p'_{k'} = \sum_{i' \in I'} c'_{i'} \cdot \prod_{k \in I} p_k \cdot \prod_{\substack{k' \in I' \\ k' \neq i'}} p'_{k'}$$

For each $i \in I$, let $\bar{E}_i \subseteq \mathbb{R}^n$ be the set of roots of the polynomial p_i. Then, for any $\bar{e}_i \in \bar{E}_i$ we have

$$(f \cdot P)(\bar{e}_i) = c_i \cdot \overbrace{\prod_{\substack{k \in I \\ k \neq i}} p_k(\bar{e}_i) \cdot \prod_{k' \in I'} p'_{k'}(\bar{e}_i)}^{R_i(\bar{e}_i)} = = (f' \cdot P)(\bar{e}_i) = 0$$

We are assuming $c_i > 0$, for any $i \in I$. So, for each $\bar{e}_i \in \bar{E}_i$ we have that $R_i(\bar{e}_i) = \prod_{\substack{k \in I \\ k \neq i}} p_k \cdot \prod_{k' \in I'} p'_{k'} = 0$.

Geometrically speaking, the sets $\bar{E}_i$ represent the points of an hyperplane in $\mathbb{R}^n$. Besides, the expression $R_i(\bar{e}_i) = 0$ represents, for each i, the union of the hyperplanes corresponding to each $k' \in I'$ and to each $k \in I$ such that $k \neq i$. Note that this union is finite. So, if $R_i(x) = 0$ holds for all the points belonging to a hyperplane (in this case E_i, or equivalently the roots of the polynomial p_i) then we have two possibilities: Either there exists $k \in I$ (with $k \neq i$) such that the hyperplane associated with p_k is $\bar{E}_i$, or there exists $k' \in I'$ such that the hyperplane associated with $p'_{k'}$ is $\bar{E}_i$. However, if two polynomials define the same hyperplane, then their coefficients must be proportional. Given the fact that all of the involved polynomials has as independent term the value 1, this proportionality coefficient must be equal to 1. In this situation, the first possibility is not possible, because the hypothesis of the lemma says that all the tuples appearing in f are pairwise different. So, there must exist $k' \in I'$ such that $p_i = p'_{k'}$. Iterating the same reasoning for every $i \in I$, we obtain a function $h : I \longrightarrow I'$ such that $d_{j,i} = d'_{j,h(i)}$ for every $i \in I$ and $1 \leq j \leq n$. Because all of the tuples appearing in f and f' are pairwise different, we trivially have that h is injective. For the sake of simplicity, reordering the indices of I', we can assume that $h(i) = i$. This allows to decompose I' as $I' = I \cup J$ where $I \cap J = \emptyset$, such that we obtain

$$f = \sum_{i \in I} \frac{c_i}{p_i} \qquad\qquad f' = \sum_{i \in I} \frac{c'_i}{p_i} + \sum_{j \in J} \frac{c'_j}{p'_j}$$

where by the hypothesis of the lemma, the polynomials p'_j (for $j \in J$) are different from p_i. We still have to prove that $J = \emptyset$. Let $Q = \prod_{i \in I} p_i$. Let us consider $f \cdot Q$ and $f' \cdot Q$:

$$f \cdot Q = \sum_{i \in I} c_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k \qquad\qquad f' \cdot Q = \sum_{i \in I} c'_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k + \sum_{j \in J} \frac{c'_j}{p'_j} \cdot \prod_{i \in I} p_i$$

For each $i \in I$, let $\bar{e}_i \in \bar{E}_i$ be such that $p_i(\bar{e}_i) = 0$ and such that for any $k \in (I - \{i\}) \cup J$ we have $p_k(\bar{e}_i) \neq 0$. Let us remark that such $\bar{e}_i$ always exists

because the set of hyperplanes appearing in $(I - \{i\}) \cup J$ is finite, and so the union of their intersections with the hyperplane $\bar{E}_i$ cannot be equal to $\bar{E}_i$. This element fulfills the following:

$$(f \cdot Q)(\bar{e}_i) = c_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k(\bar{e}_i) = (f' \cdot Q)(\bar{e}_i) = c'_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k(\bar{e}_i) + \sum_{j \in J} \frac{c'_j}{p'_j(\bar{e}_i)} \cdot \prod_{i \in I} p_i(\bar{e}_i)$$

We are assuming $c_i, c'_i > 0$. Besides, we have $p'_j(\bar{e}_i) \neq 0$ (for any $j \in J$), $\prod_{\substack{k \in I \\ k \neq i}} p_k(\bar{e}_i) \neq 0$, and $\prod_{i \in I} p_i(\bar{e}_i) = 0$. So, we obtain

$$(f \cdot Q)(\bar{e}_i) = c_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k(\bar{e}_i) = (f' \cdot Q)(\bar{e}_i) = c'_i \cdot \prod_{\substack{k \in I \\ k \neq i}} p_k(\bar{e}_i)$$

which implies $c_i = c'_i$, for each $i \in I$. Consequently, we can rewrite f and f' as

$$f = \sum_{i \in I} \frac{c_i}{p_i} \qquad f' = \sum_{i \in I} \frac{c_i}{p_i} + \sum_{j \in J} \frac{c'_j}{p'_j}$$

In order to show that $J = \emptyset$ let us consider the values $f(\bar{0})$ and $f'(\bar{0})$. We obtain $f(\bar{0}) = \sum_{i \in I} c_i$ while $f'(\bar{0}) = \sum_{i \in I} c_i + \sum_{j \in J} c'_j$. Given the fact that $f = f'$ and the values c'_j $(j \in J)$ are greater than zero, we have $\sum_{j \in J} c'_j = 0$, which implies $J = \emptyset$. From this result we immediately obtain that the function h is in fact a bijection.

Now, we will consider the case when there exists a null tuple in the definition of f. Note that there can exists at most a null tuple (because the tuples are pairwise different). That is, there exists $k \in I$ such that for any $j \in \{1 \ldots n\}$ we have $d_{k,j} = 0$. In this case, we cannot use the previous reasoning consisting in finding a root of the associated polynomial (i.e. p_k). Nevertheless, we can perform this reasoning for the rest of the elements in I. We get the following expressions for f and f':

$$f = \sum_{i \in I - \{k\}} \frac{c_i}{p_i} + c_k \qquad f' = \sum_{i \in I - \{k\}} \frac{c_i}{p_i} + \sum_{j \in J} \frac{c'_j}{p'_j}$$

where $I' = (I - \{k\}) \cup J$. We still have to prove that $J = \{k'\}$, $c'_{k'} = c_k$ and $p'_{k'} = 1$. Given the fact that $f = f'$, we obtain $c_k = \sum \frac{c'_j}{p'_j}$. Because c_k is a constant, the right hand side of the previous expression must be constant, so that all the polynomials p'_j must be equal [12] to 1. So, the associated tuples will be null. Taking into account the hypothesis of the lemma (the tuples are

[12] Let us suppose that there exists a polynomial being not equal to 1. If we pass all the constant terms from the right hand side to the left one, and we compute the sum of the remaining fractions in the right hand side, we obtain a polynomial in the denominator which is of higher degree than the one in the numerator. If we move the denominator to the left hand side of the expression, we have the equality between two polynomials with a different degree, which is not possible.

pairwise different) there will be such a (unique) tuple in J, which finishes the proof.

Theorem 3.20 Let P and P' be processes. $\widehat{\mathcal{A}}(P) \approx \widehat{\mathcal{A}}(P')$ iff $\widehat{\mathcal{A}}(P) = \widehat{\mathcal{A}}(P')$.

Proof: The right to left implication is trivial since if two normal forms are (syntactically) equal, up to commutativity, then they pass any test with the same probability. For the left to right implication, let us consider the union of the alphabets [13] of the processes $\alpha(P) \cup \alpha(P') = \{a_1, \ldots, a_n\} \subseteq Act$. Note that this set is finite. Now, let us consider $\widehat{\mathcal{A}}(P)$ and $\widehat{\mathcal{A}}(P')$:

$$\widehat{\mathcal{A}}(P) = \bigoplus_{i=1}^{m} [p_i] \sum_{j=1}^{n} [p_{i,j}]\ a_j\,;C_{i,j} \qquad\qquad \widehat{\mathcal{A}}(P') = \bigoplus_{i=1}^{m'} [p'_i] \sum_{j=1}^{n} [p'_{i,j}]\ a_j\,;C'_{i,j}$$

where, for the sake of simplicity, we have considered $p_{i,j} = 0$ if the action a_j did not appear originally in the *i-th summand* of $\widehat{\mathcal{A}}(P)$ and analogously for $\widehat{\mathcal{A}}(P')$. Let $A_i = \{(a_j, p_{i,j}) \,|\, p_{i,j} > 0\}$ and $A'_i = \{(a_j, p'_{i,j}) \,|\, p'_{i,j} > 0\}$. Depending on the processes $\widehat{\mathcal{A}}(P)$ and $\widehat{\mathcal{A}}(P')$, we distinguish three cases:

(1) Both of them are finite.
(2) One of them is finite but the other one is not.
(3) Both of them are infinite.

We will start with the first case, doing a proof by complete induction:

Base Step: We will prove that if two normal forms pass all the tests with the same probability then their *first floors* are equal, that is

$$m = m' \ \wedge\ \forall\, 1 \leq i \leq m :\ (p_i = p'_i \ \wedge\ \forall\, 1 \leq j \leq n :\ p_{i,j} = p'_{i,j})$$

Let us suppose that both normal forms pass with the same probability any test. In particular, for any probability distribution $\bar{q} = \langle q_1, q_2, \ldots q_n, q_{n+1} \rangle$ we consider the test

$$T^{\bar{q}} = \sum_{j=1}^{n+1} [q_j]\ a_j; Nil$$

with $a_{n+1} = \omega$. If we compose this test with the process $\widehat{\mathcal{A}}(P)$, we have

$$pass(\widehat{\mathcal{A}}(P)\,,\ T^{\bar{q}}) \ = \sum_{i=1}^{m} p_i \cdot \frac{q_{n+1}}{q_{n+1} + \sum_{j=1}^{n} p_{i,j} \cdot q_j}$$

This is so because internal transitions are performed first, each of them with probability equal to p_i. Afterwards, the test will be passed only if the action ω is performed (with a probability equal to the quotient of the value associated

[13] The alphabet of a process is defined by induction: $\alpha(Nil) = \alpha(\Omega) = \alpha(X) = \emptyset$; $\alpha(a; P) = \{a\} \cup \alpha(P)$; $\alpha(P +_p Q) = \alpha(P \oplus_p Q) = \alpha(P) \cup \alpha(Q)$; $\alpha(recX.P) = \alpha(P)$.

with ω, i.e. q_{n+1}, by the corresponding normalization factor. Dividing and multiplying in the previous expression by q_{n+1} we get

$$pass(\widehat{\mathcal{A}}(P)\ ,\ T^{\bar{q}}) \ = \sum_{i=1}^{m} p_i \cdot \frac{1}{1 + \sum_{j=1}^{n} p_{i,j} \cdot \frac{q_j}{q_{n+1}}}$$

The same reasoning leads us to the following expression for $\widehat{\mathcal{A}}(P')$

$$pass(\widehat{\mathcal{A}}(P')\ ,\ T^{\bar{q}}) \ = \sum_{i=1}^{m'} p'_i \cdot \frac{1}{1 + \sum_{j=1}^{n} p'_{i,j} \cdot \frac{q_j}{q_{n+1}}}$$

Let $q'_j = \frac{q_j}{q_{n+1}}$. We are assuming that both normal forms are testing equivalent. In particular, $pass(\widehat{\mathcal{A}}(P)\ ,\ T^{\bar{q}}) \ = \ pass(\widehat{\mathcal{A}}(P')\ ,\ T^{\bar{q}})$. So, we obtain

$$\sum_{i=1}^{m} p_i \cdot \frac{1}{1 + \sum_{j=1}^{n} p_{i,j} \cdot q'_j} = \sum_{i=1}^{m'} p'_i \cdot \frac{1}{1 + \sum_{j=1}^{n} p'_{i,j} \cdot q'_j}$$

We can apply Lemma 3.18, [14] and we obtain $m = m'$, and for any $1 \leq i \leq m$ and $1 \leq j \leq n$ we have $p_i = p'_i$ and $p_{i,j} = p'_{i,j}$. So, we have proven that both normal forms have the same first floor.

Inductive Step: We will suppose that both normal forms have the same *first floor*, but (at least) one of the *continuations* is different. That is, there exists a state A_i and an action $a_j \in Act(A_i)$, such that $C_{i,j} \neq C'_{i,j}$. We will give a test such that our normal forms pass it with different probabilities. So, this will be a proof by contrapositive.

Let us consider the initial states of $\widehat{\mathcal{A}}(P)$ having (at least) a continuation after an action different from the one corresponding to $\widehat{\mathcal{A}}(P')$. We will chose one of these states being *minimal* in the following sense:

> Let A_j be a state of the process $\widehat{\mathcal{A}}(P)$ such that there exists $a_k \in Act(A_j)$, such that $C_{j,k} \neq C'_{j,k}$, and such that for any state A_r of the process $\widehat{\mathcal{A}}(P)$, with $Act(A_r) \subset Act(A_j)$, we have $C_{r,s} = C'_{r,s}$, for any $a_s \in Act(A_r)$.

[14] This statement needs additional explanation. The hypothesis of Lemma 3.18 require the functions to be equal for any value of the variables (i.e. for any tuple in $\mathbb{R}^n$), while we only have this equality for the values $q'_1, \ldots, q'_n$, such that $q'_i > 0$ and $\sum q'_i = \frac{1-q_{n+1}}{q_{n+1}}$. If we consider the limit when q_{n+1} tends to zero, we can get arbitrarily high values of the variables q'_i, and so we obtain that these two functions are equal for any positive value of the variables q'_i. Note that these functions are rational. So, if they are equal for any value of the positive region of $\mathbb{R}^n$ then they are also equal for any value of $\mathbb{R}^n$. So, we can apply Lemma 3.18.

Note that such a state always exists. Obviously, the chosen state must be different from the empty set because there are no continuations after the empty state. For the sake of simplicity, we suppose $Act(A_j) = \{a_1, \ldots, a_k, \ldots, a_r\} \subseteq \{a_1, \ldots, a_n\} = \alpha(P) \cup \alpha(P')$. Once we have fixed the minimal state A_j, we will split the immediately reachable states in three groups: States whose set of actions is contained in the actions of A_j; states whose set of actions is equal to that of A_j; the rest of states, that is, those states having actions not belonging to those in A_j. Formally:

$$A^1 = \{i \mid \exists p, Q: \ \widehat{\mathcal{A}}(P) \stackrel{\epsilon}{\Longrightarrow}_p Q \ \wedge \ S(Q) = A_i \ \wedge \ Act(A_i) \subset Act(A_j)\}$$
$$A^2 = \{i \mid \exists p, Q: \ \widehat{\mathcal{A}}(P) \stackrel{\epsilon}{\Longrightarrow}_p Q \ \wedge \ S(Q) = A_i \ \wedge \ Act(A_i) = Act(A_j)\}$$
$$A^3 = \{i \mid \exists p, Q: \ \widehat{\mathcal{A}}(P) \stackrel{\epsilon}{\Longrightarrow}_p Q \ \wedge \ S(Q) = A_i \ \wedge \ i \notin A^1 \cup A^2\}$$

These sets will be used when we consider the composition of processes and tests. We supposed $C_{j,k} \neq C'_{j,k}$. So, by induction hypothesis, there exists a test T distinguishing them: $pass(C_{j,k}\ ,\ T) \neq pass(C'_{j,k}\ ,\ T)$. At least one of these values must be different from zero. Let us suppose that $pass(C_{j,k}\ ,\ T) > 0$ (the symmetrical case is equivalent). Then, for any probability distribution $\bar{q} = \langle q_1, q_2, \ldots q_r \rangle$ and any $0 < \delta < 1$, we consider the test

$$T^{\bar{q}}_{k,\delta} = (\sum_{s=1}^{r} [q_s]\ a_s; T_s) +_\delta (\sum_{s=r+1}^{n} [\frac{1}{n-r}]\ a_s; Nil) \qquad \text{where } T_s = \begin{cases} T & \text{if } s = k \\ Nil & \text{otherwise} \end{cases}$$

Note that if $n = r$ then the second sum does not appear. If we consider the composition of these tests and $\widehat{\mathcal{A}}(P)$ we obtain:

$$pass(\widehat{\mathcal{A}}(P)\ ,\ T^{\bar{q}}_{k,\delta}) = \sum_{i \in A^1} p_i \cdot \frac{\delta \cdot q_k \cdot p_{i,k} \cdot \ pass(C_{i,k}\ ,\ T)}{\sum_{s=1}^{r} \delta \cdot p_{i,s} \cdot q_s} \tag{1}$$

$$+ \sum_{i \in A^2} p_i \cdot \frac{\delta \cdot q_k \cdot p_{i,k} \cdot \ pass(C_{i,k}\ ,\ T)}{\sum_{s=1}^{r} \delta \cdot p_{i,s} \cdot q_s} \tag{2}$$

$$+ \sum_{i \in A^3} p_i \cdot \frac{\delta \cdot q_k \cdot p_{i,k} \cdot \ pass(C_{i,k}\ ,\ T)}{\sum_{s=1}^{r} \delta \cdot p_{i,s} \cdot q_s + \sum_{s=r+1}^{n} (1-\delta) \cdot p_{i,s} \cdot \frac{1}{n-r}} \tag{3}$$

Note that in the previous expression, in the three cases, if for any i we have $a_k \notin A_i$ then the continuation $C_{i,k}$ does not exist. This fact does not cause any problem because $p_{i,k}$ would be equal to 0. So, the corresponding summand would disappear. If $n = r$ then the last summand, that is expression (3), does not appear, and the δ's appearing in the other two expressions disappear. Symmetrically, we also have:

$$pass(\widehat{\mathcal{A}}(P')\,,\,T^{\bar{q}}_{k,\delta}) = \sum_{i\in A^1} p_i \cdot \frac{\delta\cdot q_k\cdot p_{i,k}\cdot\ pass(C'_{i,k}\,,\,T)}{\sum_{s=1}^{r}\delta\cdot p_{i,s}\cdot q_s} \tag{4}$$

$$+\sum_{i\in A^2} p_i \cdot \frac{\delta\cdot q_k\cdot p_{i,k}\cdot\ pass(C'_{i,k}\,,\,T)}{\sum_{s=1}^{r}\delta\cdot p_{i,s}\cdot q_s} \tag{5}$$

$$+\sum_{i\in A^3} p_i \cdot \frac{\delta\cdot q_k\cdot p_{i,k}\cdot\ pass(C'_{i,k}\,,\,T)}{\sum_{s=1}^{r}\delta\cdot p_{i,s}\cdot q_s + \sum_{s=r+1}^{n}(1-\delta)\cdot p_{i,s}\cdot\frac{1}{n-r}} \tag{6}$$

We will do the proof by contradiction. That is, we will suppose that both normal forms pass all the tests with the same probability and we will get $C_{j,k} = C'_{j,k}$. If both normal forms pass with the same probability any test, in particular, they will pass with the same probability the tests of the form $T^{\bar{q}}_{k,\delta}$, that is $pass(\widehat{\mathcal{A}}(P)\,,\,T^{\bar{q}}_{k,\delta}) = pass(\widehat{\mathcal{A}}(P')\,,\,T^{\bar{q}}_{k,\delta})$, for any $0<\delta<1$ and any probability distribution $\bar{q}$. Because the previous equality holds for any $0<\delta<1$, it also holds if we consider the limit when δ tends to zero:

$$\lim_{\delta\to 0} pass(\widehat{\mathcal{A}}(P)\,,\,T^{\bar{q}}_{k,\delta}) = \lim_{\delta\to 0} pass(\widehat{\mathcal{A}}(P')\,,\,T^{\bar{q}}_{k,\delta}) \qquad (\forall\,\bar{q}) \tag{7}$$

If we look at the previous expression, we observe that if δ tends to zero then expressions (3) and (6) tends also to zero. Note that for each $i\in A^3$, the numerator would tend to zero, while the denominator would tend to $\sum_{s=r+1}^{n} p_{i,s}\cdot\frac{1}{n-r} > 0$. Moreover, because of the election of the state A_j, the summands (1) and (4) have the same value because they have identical continuations, the processes $C_{i,k}$ and $C'_{i,k}$, for any $i\in A^1$. Therefore, if the equality (7) holds, and taking into account the previous discussion, then for any probability distribution $\bar{q} = \langle q_1, q_2, \ldots q_r\rangle$ the equality between expressions (2) and (5) must hold. If we divide each summand by $\delta\cdot q_k\cdot p_{i,k}$ we obtain:

$$\sum_{i\in A^2} p_i\cdot\frac{pass(C_{i,k}\,,\,T)}{1+\sum\limits_{\substack{s=1\\ s\neq k}}^{r}\frac{p_{i,s}}{p_{i,k}}\cdot\frac{q_s}{q_k}} = \sum_{i\in A^2} p_i\cdot\frac{pass(C'_{i,k}\,,\,T)}{1+\sum\limits_{\substack{s=1\\ s\neq k}}^{r}\frac{p_{i,s}}{p_{i,k}}\cdot\frac{q_s}{q_k}}$$

If we remove null summands and we let $q'_s = \frac{q_s}{q_k}$, we obtain:

$$\sum_{\substack{pass(C_{i,k}\,,\,T)\neq 0\\ i\in A^2}} p_i\cdot\frac{pass(C_{i,k}\,,\,T)}{1+\sum\limits_{\substack{s=1\\ s\neq k}}^{r}\frac{p_{i,s}}{p_{i,k}}\cdot q'_s} = \sum_{\substack{pass(C'_{i,k}\,,\,T)\neq 0\\ i\in A^2}} p_i\cdot\frac{pass(C'_{i,k}\,,\,T)}{1+\sum\limits_{\substack{s=1\\ s\neq k}}^{r}\frac{p_{i,s}}{p_{i,k}}\cdot q'_s}$$

Now, we can apply Lemma 3.18, because all the summands are different from zero, and applying Lemma 3.19 to the summands of the form $\frac{p_{i,s}}{p_{i,k}}$ we have that

the corresponding tuples are different. Taking into account $pass(C_{j,k}\,,\,T) > 0$ and $j \in A^2$, there exists $j' \in A^2$ verifying the following conditions:

(1) $pass(C_{j,k}\,,\,T) = pass(C'_{j',k}\,,\,T)$.
(2) $\forall\ 1 \leq s \leq r,\ s \neq k :\ \frac{p_{j,s}}{p_{j,k}} = \frac{p_{j',s}}{p_{j',k}}$.

Note that $\sum p_{j,s} = 1 = \sum p_{j',s}$. So, by applying again Lemma 3.19, we obtain $p_{j,s} = p_{j',s}$, for any $1 \leq s \leq r$. Given the fact that these tuples correspond to states of a process, and so there are no repetitions, we must have $j = j'$. But if $j = j'$ then $pass(C_{j,k}, T) = pass(C'_{j,k}, T)$, which contradicts our assumption. This finishes the proof for the case when both processes are finite.

If only one of the processes is finite, let us suppose that its *depth* is equal to n (i.e. any sequence longer than n returns $\emptyset$ as acceptance sets). Then, we can consider the $n+1$ first *floors* of the infinite process, and we can use the same reasoning that for finite process. Note that in its $n+1th$ floor, the infinite process must have some continuations, because otherwise it would be finite.

If both processes are infinite but different then the difference between them must appear in *finite time*. That is, there exists a finite sequence s such that $\mathcal{A}(\widehat{\mathcal{A}}(P), s) \neq \mathcal{A}(\widehat{\mathcal{A}}(P'), s)$. So we can use the previous reasoning, considering the processes until depth $|s|$.

Theorem 4.3 $(\mathbf{PAT}_{Act}, \sqsubseteq_{\mathbf{PAT}})$ is a *complete partial order* (cpo).

Proof: We must prove the existence of both minimal element and least upper bound for each chain. First, let us show that the tree $R \in \mathbf{PAT}_{Act}$ defined as $p(R, s, A) = 0$, for any s and A is the minimal element. Indeed, if $R' \in \mathbf{PAT}_{Act}$ we have $p(R, s, A) = 0 \leq p(R', s, A)$, for any s and A. So, we have $R \sqsubseteq_{\mathbf{PAT}} R'$.

In order to show the existence of least upper bound, let $\{R_n\}_{n\in\mathbb{N}}$ be a chain in $\mathbf{PAT}_{Act}$. The element $\sqcup R_n$ is given by $p(\sqcup R_n, s, A) = \lim_{n\in\mathbb{N}} p(R_n, s, A)$. Let us show that $\sqcup R_n$ is well defined. The elements R_n form a chain. So, we have that the values $p(R_n, s, A)$ are an increasing succession bounded by 1. Therefore, there exists the limit of this succession, which will be less than or equal to 1. Moreover, if there would exist an internal node of the limit tree such that the sum of the probabilities associated with its outgoing arcs is equal to $1 + \epsilon$ (with $\epsilon > 0$), then there exists $j \in \mathbb{N}$ such that in that internal node of the tree R_j, the sum of the probabilities is equal to $1 + \delta$, with $0 < \delta \leq \epsilon$. But this is not possible because the elements of the chain are trees. So, $\sqcup R_n$ is well defined. Once defined, we must prove that it is the least upper bound of the chain. First, we show that $\sqcup R_n$ is an upper bound of the chain. Let R_j be an element of the chain. We must show that for any sequence s and any state A, $p(R_j, s, A) \leq p(\sqcup R_n, s, A)$ holds. But this is trivial since $p(\sqcup R_n, s, A) = \lim_{n\in\mathbb{N}} p(R_n, s, A) \geq p(R_j, s, A)$. Finally, we will prove that $\sqcup R_n$

is in fact the least upper bound. Let $R \in \mathbf{PAT}_{Act}$ be such that $R_n \sqsubseteq_{\mathbf{PAT}} R$, for any $n \in \mathbb{N}$. We have the following implications:

$$\begin{aligned} R_n \sqsubseteq_{\mathbf{PAT}} R\ (\forall\, n \in \mathbb{N}) &\Longrightarrow p(R_n, s, A) \leq p(R, s, A)\ (\forall s, A\ \wedge\ \forall\, n \in \mathbb{N}) \\ &\Longrightarrow \lim_{n\in\mathbb{N}}\ p(R_n, s, A) \leq p(R, s, A)\ (\forall s, A) \\ &\Longrightarrow p(\sqcup R_n, s, A) \leq p(R, s, A)\ (\forall s, A) \\ &\Longrightarrow \sqcup R_n \sqsubseteq_{\mathbf{PAT}} R \end{aligned}$$

Proposition 4.7 For any $a \in Act$, the function $a;\_ :: \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ is monotonous and continuous. Moreover, for any $0 < p < 1$, the functions $\_\oplus_p\_, \_+_p\_ :: \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ are monotonous and continuous in both arguments.

Proof: In order to show the monotony of $a;\_ :: \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$, let $R, R' \in \mathbf{PAT}_{Act}$ be such that $R \sqsubseteq_{\mathbf{PAT}} R'$. We must prove $a; R \sqsubseteq_{\mathbf{PAT}} a; R'$, or equivalently, that for any s, A we have $p(R, s, A) \leq p(R', s, A)$. We distinguish three cases:

- $s = \epsilon\ \wedge\ A = \{(a, 1)\}$. We have $p(a; R, \epsilon, A) = 1 \leq 1 = p(a; R', \epsilon, A)$.
- $s = \langle \{(a, 1)\}\, a \rangle \circ s'$. For each state A we have $p(a; R, s, A) = p(R, s', A) \leq p(R', s', A) = p(a; R', s, A)$.
- For the rest of the sequences, we have $p(a; R, s, A) = 0 \leq 0 = p(a; R', s, A)$.

In order to show the continuity of $a;\_ :: \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$, let $\{R_n\}_{n\in\mathbb{N}}$ be a chain in $\mathbf{PAT}_{Act}$. We have that $\{a; R_n\}_{n\in\mathbb{N}}$ are also a chain (because we have showed that the prefix operator is monotonous). We must prove that for any sequence s and state A the following identity holds: $p(a; \sqcup\{R_n\}_{n\in\mathbb{N}}, s, A) = p(\sqcup\{a; R_n\}_{n\in\mathbb{N}}, s, A)$. This follows from the following chain of equalities:

$$p(a; \sqcup\{R_n\}_{n\in\mathbb{N}}, s, A) = \begin{cases} 1 & \text{if } s = \epsilon\ \wedge\ A = \{(a, 1)\} \\ p(\sqcup\{R_n\}_{n\in\mathbb{N}}, s', A) & \text{if } s = \langle \{(a, 1)\}\, a \rangle \circ s' \\ 0 & \text{otherwise} \end{cases}$$

$$= \begin{cases} 1 & \text{if } s = \epsilon\ \wedge\ A = \{(a, 1)\} \\ \lim\limits_{n\in\mathbb{N}} p(R_n, s', A) & \text{if } s = \langle \{(a, 1)\}\, a \rangle \circ s' \\ 0 & \text{otherwise} \end{cases}$$

$$= \lim_{n\in\mathbb{N}} \begin{cases} 1 & \text{if } s = \epsilon\ \wedge\ A = \{(a, 1)\} \\ p(R_n, s', A) & \text{if } s = \langle \{(a, 1)\}\, a \rangle \circ s' \\ 0 & \text{otherwise} \end{cases}$$

$$= \lim_{n\in\mathbb{N}} p(a; R_n, s, A) = p(\sqcup\{a; R_n\}_{n\in\mathbb{N}}, s, A)$$

Next we will prove the result for $\_ \oplus_p \_ :: \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$. Because of symmetry, it is enough to perform the proof for one of the arguments. We choose the first one. We start with monotony. Let $R_1, R_2 \in \mathbf{PAT}_{Act}$ be such that $R_1 \sqsubseteq_{\mathbf{PAT}} R_2$. We must prove that $R_1 \oplus_p R \sqsubseteq_{\mathbf{PAT}} R_2 \oplus_p R$ holds for any $R \in \mathbf{PAT}_{Act}$, or equivalently, that for any sequence s and state A we have $p(R_1 \oplus_p R, s, A) \leq p(R_2 \oplus_p R, s, A)$. But this is straightforward since we have the following: $p(R_1 \oplus_p R, s, A) = p \cdot p(R_1, s, A) + (1-p) \cdot p(R, s, A) \leq p \cdot p(R_2, s, A) + (1-p) \cdot p(R, s, A) = p(R_2 \oplus_p R, s, A)$.

For the proof of continuity, let $\{R_n\}_{n \in \mathbb{N}}$ be a chain in $\mathbf{PAT}_{Act}$. We have that the trees $\{R_n \oplus_p R\}_{n \in \mathbb{N}}$ are also a chain (because we have showed that the internal choice operator is monotonous). We must prove that for any sequence s and state A, we have $p(\sqcup\{R_n\}_{n \in \mathbb{N}} \oplus_p R, s, A) = p(\sqcup\{R_n \oplus_p R\}_{n \in \mathbb{N}}, s, A)$. This follows from the following chain of equalities

$$\begin{aligned} p(\sqcup\{R_n\}_{n \in \mathbb{N}} \oplus_p R, s, A) &= p \cdot p(\sqcup\{R_n\}_{n \in \mathbb{N}}, s, A) + (1-p) \cdot \mathrm{prob}(R, s, A) \\ &= p \cdot (\lim_{n \in \mathbb{N}} p(R_n, s, A)) + (1-p) \cdot p(R, s, A) \\ &= \lim_{n \in \mathbb{N}} (p \cdot p(R_n, s, A) + (1-p) \cdot p(R, s, A)) \\ &= \lim_{n \in \mathbb{N}} p(R_n \oplus_p R, s, A) = p(\sqcup\{R_n \oplus_p R\}_{n \in \mathbb{N}}, s, A) \end{aligned}$$

Finally, the proof for the functions $\_ +_p \_ :: \mathbf{PAT}_{Act} \times \mathbf{PAT}_{Act} \longrightarrow \mathbf{PAT}_{Act}$ is again shown only for the first argument. In order to prove monotony, let $R_1, R_2 \in \mathbf{PAT}_{Act}$ be such that $R_1 \sqsubseteq_{\mathbf{PAT}} R_2$. We have to prove that for any tree $R \in \mathbf{PAT}_{Act}$, $R_1 +_p R \sqsubseteq_{\mathbf{PAT}} R_2 +_p R$ holds, or equivalently, that for any sequence s and state A, $p(R_1 +_p R, s, A) \leq p(R_2 +_p R, s, A)$ holds. We will prove it by induction on the length of s. If $s = \epsilon$ we have:

$$\begin{aligned} p(R_1 +_p R, \epsilon, A) &= \sum_{A = B \cup_p C} p(R_1, \epsilon, B) \cdot p(R, \epsilon, C) \\ &\leq \sum_{A = B \cup_p C} p(R_2, \epsilon, B) \cdot p(R, \epsilon, C) = p(R_2 +_p R, \epsilon, A) \end{aligned}$$

If $s = \langle A\, a \rangle \circ s'$, where $s_B = \langle B\, a \rangle \circ s'$ and $s_C = \langle C\, a \rangle \circ s'$, we have:

$$\begin{aligned} p(R_1 +_p R, s, X) = & \sum_{A = B \cup_p C} \tfrac{p \cdot pro(a,B)}{p \cdot pro(a,B) + (1-p) \cdot pro(a,C)} \cdot p(R_1, s_B, X) \cdot p(R, C) \\ & + \sum_{A = B \cup_p C} \tfrac{(1-p) \cdot pro(a,C)}{p \cdot pro(a,B) + (1-p) \cdot pro(a,C)} \cdot p(R, s_C, X) \cdot p(R_1, B) \\ \leq & \sum_{A = B \cup_p C} \tfrac{p \cdot pro(a,B)}{p \cdot pro(a,B) + (1-p) \cdot pro(a,C)} \cdot p(R_2, s_B, X) \cdot p(R, C) \\ & + \sum_{A = B \cup_p C} \tfrac{(1-p) \cdot pro(a,C)}{p \cdot pro(a,B) + (1-p) \cdot pro(a,C)} \cdot p(R, s_C, X) \cdot p(R_2, B) \\ = & \; p(R_2 +_p R, s, X) \end{aligned}$$

Finally, for showing the continuity of the external choice semantic function, let $\{R_n\}_{n\in\mathbb{N}}$ be a chain in $\mathbf{PAT}_{Act}$. We have that the trees $\{R_n +_p R\}_{n\in\mathbb{N}}$ are also a chain (we have proved that the external choice operator is monotonous). We must prove that for any sequence s and state A, the following equality holds: $p(\sqcup\{R_n\}_{n\in\mathbb{N}} +_p R, s, A) = p(\sqcup\{R_n +_p R\}_{n\in\mathbb{N}}, s, A)$. We will perform the proof by induction. For $s = \epsilon$ we have

$$\begin{aligned}
p(\sqcup\{R_n\}_{n\in\mathbb{N}} +_p R, \epsilon, A) &= \sum_{A=B\cup_p C} p(\sqcup\{R_n\}_{n\in\mathbb{N}}, \epsilon, B)\cdot p(R,\epsilon,C)\\
&= \sum_{A=B\cup_p C} (\lim_{n\in\mathbb{N}} p(R_n,\epsilon,B))\cdot p(R,\epsilon,C)\\
&= \lim_{n\in\mathbb{N}} \sum_{A=B\cup_p C} p(R_n,\epsilon,B)\cdot p(R,\epsilon,C)\\
&= \lim_{n\in\mathbb{N}} p(R_n +_p R,\epsilon,A) = p(\sqcup\{R_n +_p R\}_{n\in\mathbb{N}},\epsilon,A)
\end{aligned}$$

If $s = \langle A\, a\rangle \circ s'$, where $s_B = \langle B\, a\rangle \circ s'$ and $s_C = \langle C\, a\rangle \circ s'$, we have:

$$p(\sqcup\{R_n\}_{n\in\mathbb{N}} +_p R, s, X) =$$

$$\begin{aligned}
&= \sum_{A=B\cup_p C} \tfrac{p\cdot pro(a,B)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(\sqcup\{R_n\}_{n\in\mathbb{N}}, s_B, X)\cdot p(R,C)\\
&\quad + \sum_{A=B\cup_p C} \tfrac{(1-p)\cdot pro(a,C)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(R, s_C, X)\cdot p(\sqcup\{R_n\}_{n\in\mathbb{N}}, B)\\
&= \sum_{A=B\cup_p C} \tfrac{p\cdot pro(a,B)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot (\lim_{n\in\mathbb{N}} p(R_n, s_B, X))\cdot p(R,C)\\
&\quad + \sum_{A=B\cup_p C} \tfrac{(1-p)\cdot pro(a,C)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(R, s_C, X)\cdot (\lim_{n\in\mathbb{N}} p(R_n, B))\\
&= \lim_{n\in\mathbb{N}} \sum_{A=B\cup_p C} \tfrac{p\cdot pro(a,B)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(R_n, s_B, X)\cdot p(R,C)\\
&\quad + \lim_{n\in\mathbb{N}} \sum_{A=B\cup_p C} \tfrac{(1-p)\cdot pro(a,C)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(R, s_C, X)\cdot p(R_n, B)\\
&= \lim_{n\in\mathbb{N}} \left(\begin{array}{l} \displaystyle\sum_{A=B\cup_p C} \tfrac{p\cdot pro(a,B)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(R_n, s_B, X)\cdot p(R,C)\\ + \displaystyle\sum_{A=B\cup_p C} \tfrac{(1-p)\cdot pro(a,C)}{p\cdot pro(a,B)+(1-p)\cdot pro(a,C)}\cdot p(R, s_C, X)\cdot p(R_n, B) \end{array} \right)\\
&= \lim_{n\in\mathbb{N}} p(R_n +_p R, s, A) = p(\sqcup\{R_n +_p R\}_{n\in\mathbb{N}}, s, A)
\end{aligned}$$

Theorem 4.9 For any process P and any sequence s, we have

$$p([\![P]\!], s, A) = \sum_{P'} \{\!|\ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge\ S(P') = A\ |\!\}$$

Proof: We will do the proof by a double induction: Structural induction and induction on the length of the sequence s.

$\boxed{P = Nil}$

Let us consider $s = \epsilon$. The process Nil is stable and $S(Nil) = \emptyset$. Thus, we have $\sum_{Q'} \{\!\{ p_i \mid Nil \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = \emptyset \}\!\} = 1$. Moreover, we also have $\sum_{Q'} \{\!\{ p_i \,|\, Nil \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \wedge A \neq \emptyset \}\!\} = 0$. Besides, $p([\![Nil]\!], \epsilon, \emptyset) = 1$, and $p([\![Nil]\!], \epsilon, A) = 0$, for any $A \neq \emptyset$.

If $s = \langle B\, b\rangle \circ s'$ we have $Nil \stackrel{s}{\not\Longrightarrow}$. So, $\sum_{Q'} \{\!\{ p_i | Nil \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \}\!\} = 0$, for any state A. Besides, if $s \neq \epsilon$ then $p([\![Nil]\!], s, A) = 0$ for any state A.

$\boxed{P = \Omega}$

Ω cannot become stabilized (i.e. $\Omega \stackrel{s}{\not\Longrightarrow}$). This implies $P \stackrel{s}{\not\Longrightarrow}$ for any sequence s. So, $\sum_{Q'} \{\!\{ p_i \,|\, \Omega \stackrel{s}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = A \}\!\} = 0$, for any state A. Besides, for any sequence s and state A we have $p([\![\Omega]\!], s, A) = 0$.

$\boxed{P = a; P'}$

Let us consider $s = \epsilon$. The process P is stable and such that $S(P) = \{(a, 1)\}$. So, $\sum_{Q'} \{\!\{ p_i \mid P \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = \{(a,1)\} \}\!\} = 1$ while for any other state we have $\sum_{Q'} \{\!\{ p_i \mid P \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = A \ \wedge \ A \neq \{(a,1)\} \}\!\} = 0$. Besides, $p([\![P]\!], \epsilon, \{(a,1)\}) = 1$, while if $A \neq \{(a,1)\}$ then $p([\![P]\!], \epsilon, A) = 0$.

Now, we consider $s = \langle B\, b\rangle \circ s'$. We have that P can perform only the transition $P \stackrel{a}{\longrightarrow}_1 P'$. Therefore, if $B \neq \{(a,1)\}$ then $P \stackrel{s}{\not\Longrightarrow}$. So, for any state A we have $\sum_{Q'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = A \}\!\} = 0$. Similarly, for any state A we have $p([\![P]\!], s, A) = 0$. Let us suppose now that $B = \{(a,1)\}$ and so $b = a$. We have $P \stackrel{\langle B\, a\rangle \circ s'}{\Longrightarrow}_p Q$ iff $P' \stackrel{s'}{\Longrightarrow}_p Q$. So, $\sum_{Q'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = A \}\!\} = \sum_{Q'} \{\!\{ p_i \,|\, P' \stackrel{s'}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \}\!\}$. On the one hand, by induction hypothesis, we have $\sum_{Q'} \{\!\{ p_i \,|\, P' \stackrel{s'}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = A \}\!\} = p([\![P']\!], s', A)$ while on the other hand, by the definition of $[\![a; P']\!]$, we have $p([\![P']\!], s', A) = p([\![P]\!], s, A)$. Finally, putting together the previous equalities, we obtain the desired result: $\sum_{Q'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} Q' \ \wedge \ S(Q') = A \}\!\} = p([\![P]\!], s, A)$.

$\boxed{P = P_1 \oplus_p P_2}$

P can perform two internal transitions: $P \stackrel{}{\rightarrowtail}_p P_1$ and $P \stackrel{}{\rightarrowtail}_{1-p} P_2$. So, the probability with which P performs the sequence s is equal to the addition of the probabilities with which P_1 and P_2 perform this sequence, multiplied

by p and $1-p$, respectively. That is,

$$\sum_{Q'} \{\!|\, p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}= \; p \cdot \sum_{Q'} \{\!|\, p_i \mid P_1 \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\} + (1-p) \cdot \sum_{Q'} \{\!|\, p_i \mid P_2 \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$$

Besides, by the definition of the semantic function $\oplus_p$, we have $p([\![P]\!], s, A) = p \cdot p([\![P_1]\!], s, A) + (1-p) \cdot p([\![P_2]\!], s, A)$, while by induction hypothesis we obtain both $p([\![P_1]\!], s, A) = \sum_{Q'} \{\!|\, p_i \mid P_1 \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$ and $p([\![P_2]\!], s, A) = \sum_{Q'} \{\!|\, p_i \mid P_2 \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$. So, we finally get $p([\![P]\!], s, A) = \sum_{Q'} \{\!|\, p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$.

$\boxed{P = P_1 +_p P_2}$

If one of the process cannot become stable (i.e. $P_1 \stackrel{\epsilon}{\not\Longrightarrow} \vee P_2 \stackrel{\epsilon}{\not\Longrightarrow}$) then P cannot become stable. So, $\sum_{Q'} \{\!|\, p_i | P \stackrel{s}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\} = 0$ for any state A and sequence s. Besides, by the definition of $+_p$, we have $p([\![P]\!], s, A) = 0$ for any state A and sequence s. So, the result holds in this case.

Let us now assume that both of them can be stabilized. We will perform again the proof by induction on the length of s.

$\underline{s = \epsilon}$: Process P can perform the empty sequence once P_1 and P_2 become stable. Let us consider the values $p_A^{P_1} = \sum_{Q'} \{\!|\, p_i \mid P_1 \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$ and $p_A^{P_2} = \sum_{Q'} \{\!|\, p_i \mid P_2 \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$. The states reachable by P after the empty sequence are those given by Definition 4.4, considering the corresponding reachable states of P_1 and P_2. That is, we have $\sum_{Q'} \{\!|\, p_i \mid P \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\} = \sum_{A_1, A_2} \{\!|\, p_{A_1}^{P_1} \cdot p_{A_2}^{P_2} \mid A = A_1 \cup_p A_2 \,|\!\}$. Besides, $p([\![P_1 +_p P_2]\!], \epsilon, A) = \sum_{A = B \cup_p C} p([\![P_1]\!], \epsilon, B) \cdot p([\![P_2]\!], \epsilon, C)$. By induction hypothesis we have $p([\![P_1]\!], \epsilon, B) = p_B^{P_1}$ and $p([\![P_2]\!], \epsilon, C) = p_C^{P_2}$. So, we obtain the required result: $p([\![P_1 +_p P_2]\!], \epsilon, A) = \sum_{Q'} \{\!|\, p_i \mid P \stackrel{\epsilon}{\Longrightarrow}_{p_i} Q' \wedge S(Q') = A \,|\!\}$.

$\underline{s = \langle B\, b \rangle \circ s'}$: Process P can perform the sequence s if the process P_1 performs a generalized internal transition, becoming a process P_1' having as associated state B_1, the process P_2 performs a generalized internal transition, becoming a process P_2' having as associated state B_2, and $B = B_1 \cup_p B_2$. That is, if $P_1 \rightarrowtail^*_{p_1} P_1'$ and $P_2 \rightarrowtail^*_{p_2} P_2'$, then $P \rightarrowtail^*_{p_1 \cdot p_2} P_1' +_p P_2'$ (applying rules $(EXT1)$, $(EXT2)$, and $(EXT3)$). Now, we have three possibilities:

$\diamond\ P_1' \stackrel{b}{\longrightarrow} \wedge P_2' \stackrel{b}{\not\longrightarrow}$

In this case, P performs b from P_1' and then performs the sequence s'. That

is, we have the following:

- $P_1 \rightarrowtail^*_{p_1} P_1' \xrightarrow{b}_{p_1'} P_1'' \stackrel{s'}{\Longrightarrow}_{p_1''} Q' \ \wedge\ S(P_1') = B_1 \ \wedge\ S(Q') = A$
- $P_2 \rightarrowtail^*_{p_2} P_2' \ \wedge\ S(P_2') = B_2 \ \wedge\ P_2' \not\xrightarrow{b} \ \wedge B = B_1 \cup_p B_2$

We have $P \rightarrowtail^*_{p_1 \cdot p_2} P_1' +_p P_2' \xrightarrow{b}_{p \cdot p_1'} P_1'' \stackrel{s'}{\Longrightarrow}_{p_1''} Q'$, so [15]

$$P \stackrel{s}{\Longrightarrow}_q Q' \ \wedge\ S(Q') = A \text{ iff } q = p_1 \cdot p_2 \cdot \frac{p \cdot p_1'}{p \cdot \sum_{Q'} \{\!\{\ q_i \mid P_1' \xrightarrow{b}_{q_i} Q'\ \}\!\}} \cdot p_1''$$
$$\text{iff } P_1 \stackrel{s_{B_1}}{\Longrightarrow}_{q'} Q' \ \wedge\ P_2 \rightarrowtail^*_{p_2} P_2'$$

where $s_{B_1} = \langle B_1\, b\rangle \circ s'$ and $q' = p_1 \cdot \frac{p_1'}{\sum_{Q'} \{\!\{\ q_i \mid P_1' \xrightarrow{b}_{q_i} Q'\ \}\!\}} \cdot p_1''$. We have that $S(P_1') = B_1$ implies $\sum_{Q'} \{\!\{\ q_i \mid P_1' \xrightarrow{b}_{q_i} Q'\ \}\!\} = pro(b, B_1)$. If we group the summands of this form, we obtain [16]

$$\sum_{\substack{B = B_1 \cup_p B_2 \\ b \in B_1 \ \wedge\ b \notin B_2}} \left(\begin{array}{l} \displaystyle\sum_{Q'} \{\!\{\ q_i \mid P_1 \stackrel{s_{B_1}}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A\ \}\!\} \\ \cdot \displaystyle\sum_{Q'} \{\!\{\ q_i \mid P_2 \stackrel{\epsilon}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = B_2\ \}\!\} \end{array} \right) \tag{8}$$

$\diamond\ P_1' \not\xrightarrow{b} \ \wedge\ P_2' \xrightarrow{b}$

Using a similar reasoning we obtain:

$$\sum_{\substack{B = B_1 \cup_p B_2 \\ b \notin B_1 \ \wedge\ b \in B_2}} \left(\begin{array}{l} \displaystyle\sum_{Q'} \{\!\{\ q_i \mid P_1 \stackrel{\epsilon}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = B_1\ \}\!\} \\ \cdot \displaystyle\sum_{Q'} \{\!\{\ q_i \mid P_2 \stackrel{s_{B_2}}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A\ \}\!\} \end{array} \right) \tag{9}$$

$\diamond\ P_1' \xrightarrow{b} \ \wedge\ P_2' \xrightarrow{b}$

In this case, P will perform either one of the transitions labeled by b from the ones that P_1' can perform or one of the corresponding to P_2', multiplying the probabilities associated with these transitions by p and $1-p$, respectively. Depending on which process performs b, we will have two cases:

[15] Formally, we should distinguish two cases in the expression that follows: $B_2 = \emptyset$ and $B_2 \neq \emptyset$. In the first case, instead of the transition $\xrightarrow{b}_{p \cdot p_1'}$, we would have $\xrightarrow{b}_{p_1'}$. This does not cause any problem because when considering $P \stackrel{s}{\Longrightarrow}_q$, the value of p would not influence the probability q. The same comment is valid in the following case for $(1-p) \cdot p_2'$.

[16] In the rest of the proof $b \in X$ (resp. $b \notin X$) stands for $pro(b, X) > 0$ (resp. $pro(b, X) = 0$).

$\triangle$ P_1' performs action b.

In this case, we have:

- $P_1 \rightarrowtail^*_{p_1} P_1' \xrightarrow{b}_{p_1'} P_1'' \stackrel{s'}{\Longrightarrow}_{p_1''} Q' \ \wedge\ S(P_1') = B_1 \ \wedge\ S(Q') = A$
- $P_2 \rightarrowtail^*_{p_2} P_2' \ \wedge\ S(P_2') = B_2 \ \wedge\ P_2' \xrightarrow{b} \ \wedge B = B_1 \cup_p B_2$

So, $P \rightarrowtail^*_{p_1 \cdot p_2} P_1' +_p P_2' \xrightarrow{b}_{p \cdot p_1'} P_1'' \stackrel{s'}{\Longrightarrow}_{p_1''} Q'$. We have:

$$
\begin{aligned}
& P \stackrel{s}{\Longrightarrow}_q Q' \ \wedge\ S(Q') = A \\
& \quad \text{iff } q = p_1 \cdot p_2 \cdot \frac{p \cdot p_1'}{p \cdot \sum_{Q'} \{\!|\, q_i \mid P_1' \xrightarrow{b}_{q_i} Q' \,|\!\} + (1-p) \cdot \sum_{Q'} \{\!|\, q_i \mid P_2' \xrightarrow{b}_{q_i} Q' \,|\!\}} \cdot p_1'' \\
& \quad \text{iff } P_1 \stackrel{s_{B_1}}{\Longrightarrow}_{q'} Q' \ \wedge\ P_2 \rightarrowtail^*_{p_2} P_2' \\
& \qquad \wedge\, q = \frac{p \cdot \sum_{Q'} \{\!|\, q_i \mid P_1' \xrightarrow{b}_{q_i} Q' \,|\!\}}{p \cdot \sum_{Q'} \{\!|\, q_i \mid P_1' \xrightarrow{b}_{q_i} Q' \,|\!\} + (1-p) \cdot \sum_{Q'} \{\!|\, q_i \mid P_2' \xrightarrow{b}_{q_i} Q' \,|\!\}} \cdot q' \cdot p_2
\end{aligned}
$$

where $s_{B_1} = \langle B_1\, b \rangle \circ s'$ and $q' = p_1 \cdot \frac{p_1'}{\sum_{Q'} \{\!|\, q_i \mid P_1' \xrightarrow{b}_{q_i} Q' \,|\!\}} \cdot p_1''$. Note that $S(P_1') = B_1$ and $S(P_2') = B_2$ imply $\sum_{Q'} \{\!|\, q_i \mid P_1' \xrightarrow{b}_{q_i} Q' \,|\!\} = pro(b, B_1)$ and $\sum_{Q'} \{\!|\, q_i \mid P_2' \xrightarrow{b}_{q_i} Q' \,|\!\} = pro(b, B_2)$. Finally, if we group these summands, we obtain:

$$
\begin{aligned}
\sum_{\substack{B = B_1 \cup_p B_2 \\ b \in B_1 \ \wedge\ b \in B_2}} & \frac{p \cdot pro(b,B_1)}{p \cdot pro(b,B_1) + (1-p) \cdot pro(b,B_2)} \cdot \sum_{Q'} \{\!|\, q_i \mid P_1 \stackrel{s_{B_1}}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A \,|\!\} \\
& \cdot \sum_{Q'} \{\!|\, q_i \mid P_2 \stackrel{\epsilon}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = B_2 \,|\!\} \quad (10)
\end{aligned}
$$

$\triangle$ P_2' performs action b.

Using a similar reasoning we get:

$$
\begin{aligned}
\sum_{\substack{B = B_1 \cup_p B_2 \\ b \in B_1 \ \wedge\ b \in B_2}} & \frac{(1-p) \cdot pro(b,B_2)}{p \cdot pro(b,B_1) + (1-p) \cdot pro(b,B_2)} \cdot \sum_{Q'} \{\!|\, q_i \mid P_2 \stackrel{s_{B_2}}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A \,|\!\} \\
& \cdot \sum_{Q'} \{\!|\, q_i \mid P_1 \stackrel{\epsilon}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = B_1 \,|\!\} \quad (11)
\end{aligned}
$$

Putting together the previous expressions we obtain $(8) + (9) + (10) + (11) = \sum_{Q'} \{\!|\, q_i \mid P \stackrel{s}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A \,|\!\}$. Besides, by the definition of the external choice semantic operator, we have

$$
\begin{aligned}
p([\![P_1 +_p P_2]\!], s, A) = & \sum_{B = B_1 \cup_p B_2} \frac{p \cdot pro(b,B_1)}{p \cdot pro(b,B_1) + (1-p) \cdot pro(b,B_2)} \cdot p([\![P_1]\!], s_{B_1}, A) \cdot p([\![P_2]\!], \epsilon, B_2) \\
& + \sum_{B = B_1 \cup_p B_2} \frac{(1-p) \cdot pro(b,B_2)}{p \cdot pro(b,B_1) + (1-p) \cdot pro(b,B_2)} \cdot p([\![P_2]\!], s_{B_2}, A) \cdot p([\![P_1]\!], \epsilon, B_1)
\end{aligned}
$$

where $s_{B_1} = \langle B_1\, b\rangle \circ s'$ and $s_{B_2} = \langle B_2\, b\rangle \circ s'$. As before, the previous expression can be split into four sums:

$$
\begin{aligned}
p([\![P_1 +_p P_2]\!], s, A) = \\
&\sum_{\substack{B = B_1 \cup_p B_2 \\ b\in B_1 \ \wedge\ b\notin B_2}} p([\![P_1]\!], s_{B_1}, A)\cdot p([\![P_2]\!], \epsilon, B_2) \\
+ &\sum_{\substack{B = B_1 \cup_p B_2 \\ b\notin B_1 \ \wedge\ b\in B_2}} p([\![P_2]\!], s_{B_2}, A)\cdot p([\![P_1]\!], \epsilon, B_1) \\
+ &\sum_{\substack{B = B_1 \cup_p B_2 \\ b\in B_1 \ \wedge\ b\in B_2}} \tfrac{p\cdot pro(b,B_1)}{p\cdot pro(b,B_1)+(1-p)\cdot pro(b,B_2)}\cdot p([\![P_1]\!], s_{B_1}, A)\cdot p([\![P_2]\!], \epsilon, B_2) \\
+ &\sum_{\substack{B = B_1 \cup_p B_2 \\ b\in B_1 \ \wedge\ b\in B_2}} \tfrac{(1-p)\cdot pro(b,B_2)}{p\cdot pro(b,B_1)+(1-p)\cdot pro(b,B_2)}\cdot p([\![P_2]\!], s_{B_2}, A)\cdot p([\![P_1]\!], \epsilon, B_1)
\end{aligned}
$$

where $s_{B_1} = \langle B_1\, b\rangle \circ s'$ and $s_{B_2} = \langle B_2\, b\rangle \circ s'$. Note that in the first two summands, the factors $\frac{p\cdot pro(b,B_1)}{p\cdot pro(b,B_1)+(1-p)\cdot pro(b,B_2)}$ and $\frac{(1-p)\cdot pro(b,B_2)}{p\cdot pro(b,B_1)+(1-p)\cdot pro(b,B_2)}$ have disappeared. The reason is that in the first case $pro(b, B_1) = 0$ while in the second case we have $pro(b, B_2) = 0$. By induction hypothesis we obtain:

$$
\begin{aligned}
p([\![P_1]\!], s_{B_1}, A) &= \textstyle\sum_{Q'} \{\!\{\, q_i \mid P_1 \stackrel{s_{B_1}}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A \,\}\!\} \\
p([\![P_1]\!], \epsilon, B_1) &= \textstyle\sum_{Q'} \{\!\{\, q_i \mid P_1 \stackrel{\epsilon}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = B_1 \,\}\!\} \\
p([\![P_2]\!], s_{B_2}, A) &= \textstyle\sum_{Q'} \{\!\{\, q_i \mid P_2 \stackrel{s_{B_2}}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A \,\}\!\} \\
p([\![P_2]\!], \epsilon, B_2) &= \textstyle\sum_{Q'} \{\!\{\, q_i \mid P_2 \stackrel{\epsilon}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = B_2 \,\}\!\}
\end{aligned}
$$

which easily implies the desired result:

$$
\sum_{Q'} \{\!\{\, q_i \mid P \stackrel{s}{\Longrightarrow}_{q_i} Q' \ \wedge\ S(Q') = A \,\}\!\} = p([\![P_1 +_p P_2]\!], s, A)
$$

Finally, we consider recursion. Let $P = recX.P'(X)$. In order to simplify the presentation, we will show the proof only for the case when $P'(X)$ is finite (that is, it does not contain occurrences of recursion). Given the fact that the P_n are finite, we have $p([\![P_n]\!], s, A) = \sum_{P_n''} \{\!\{\, p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge\ S(P_n'') = A \,\}\!\}$. So, we deduce

$$
\lim_{n\in\mathbb{N}} \sum_{P_n''} \{\!\{\, p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge\ S(P_n'') = A \,\}\!\} = \lim_{n\in\mathbb{N}} p([\![P_n]\!], s, A) \tag{12}
$$

We must show that this value is equal to $\sum_{P'} \{\!\{\, p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge\ S(P') = A \,\}\!\}$, that is, that the operational semantics is continuous in some sense. Applying

Lemma 4.8 from right to left, we obtain

$$\begin{array}{l} \lim_{n\in\mathbb{N}} \sum_{P_n''} \{\!\{ p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge \ S(P_n'') = A \}\!\} \\ \leq \sum_{P'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} \end{array} \tag{13}$$

It still remains to prove the converse. We have that for any $\delta > 0$ there exists a finite multiset of computations of the form $P \stackrel{s}{\Longrightarrow}_q P'$, namely F_δ, such that

$$\sum_{F_\delta} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} > \sum_{P'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} - \delta$$

Given the fact that F_δ is a finite multiset of computations $P \stackrel{s}{\Longrightarrow}_q P'$, by applying Lemma 4.8 from left to right, we obtain that there exist n such that the computations belonging to F_δ correspond to computations $P_n \stackrel{s}{\Longrightarrow}_q P_n''$. So,

$$\sum_{F_\delta} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} \leq \sum_{P_n''} \{\!\{ p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge \ S(P_n'') = A \}\!\}$$

Considering the limit when n tends to infinite we obtain

$$\sum_{P'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} - \delta < \lim_{n\in\mathbb{N}} \sum_{P_n''} \{\!\{ p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge \ S(P_n'') = A \}\!\}$$

Moreover, considering the limit when δ tends to zero we conclude

$$\begin{array}{l} \sum_{P'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} \\ \leq \lim_{n\in\mathbb{N}} \sum_{P_n''} \{\!\{ p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge \ S(P_n'') = A \}\!\} \end{array} \tag{14}$$

Combining (13) and (14) we obtain $\sum_{P'} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P' \ \wedge \ S(P') = A \}\!\} = \lim_{n\in\mathbb{N}} \sum_{P_n''} \{\!\{ p_i \mid P_n \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge \ S(P_n'') = A \}\!\}$. Finally, considering the previous equality together with expression (12) and taking into account that, by definition, $p([\![P]\!], s, A) = \lim p([\![P_n]\!], s, A)$, we obtain the desired result: $p([\![P]\!], s, A) = \sum_{P_n''} \{\!\{ p_i \mid P \stackrel{s}{\Longrightarrow}_{p_i} P_n'' \ \wedge \ S(P_n'') = A \}\!\}$.

The general case in which the processes can contain recursions inside P' is a little bit more involved. We need to *unfold* all the recursions at the same time and perform an additional induction on the number of times that recursion has been unwound.

Proposition 5.7 The axioms (**II**), (**CI**), (**AI**), (**CE**), (**NE**), (**D**), (**DI**), (**DE**), (**NE**), (**DEI**), (**EBE**), and (**IBE**) are sound.

Proof: The proofs for the first five axioms are easy with respect to the testing semantics $\approx$. Let us remind that the set of probabilistic barbs have the

same discriminatory power as the whole family of tests. As an example, let us consider the soundness proof for (**II**). Let $T \in \mathcal{PB}$. Given the fact that this test cannot perform internal transitions, the only possible transitions that the composition of the process $R = P \oplus_p P$ and the test T can perform are $R \mid T \succ\!\!\rightarrow_p P \mid T$ and $R \mid T \succ\!\!\rightarrow_{1-p} P \mid T$. Thus, we deduce $pass(R\,,\,T) = p \cdot pass(P\,,\,T) + (1-p) \cdot pass(P\,,\,T) = pass(P\,,\,T)$.

Soundness proofs for (**D**), (**DI**), and (**DE**) are trivial with respect to $\sqsubseteq_{\mathbf{PAT}}$.

Next we show the soundness of (**DEI**). Let $T \in \mathcal{PB}$. Applying reiteratively the rules $(EXT1)$, $(EXT2)$ and $(EXT3)$ we obtain:

$$P_i \succ\!\!\rightarrow^*_{r_i} P_i' \Longrightarrow (P_1 +_p P_2 \succ\!\!\rightarrow^*_{r_1 \cdot r_2} P_1' +_p P_2') \ \wedge\ (P_1 +_p P_3 \succ\!\!\rightarrow^*_{r_1 \cdot r_3} P_1' +_p P_3')$$

$$\Longrightarrow \begin{cases} (P_1 +_p P_2) \oplus_q (P_1 +_p P_3) \succ\!\!\rightarrow^*_{q \cdot r_1 \cdot r_2} (P_1' +_p P_2') \\ (P_1 +_p P_2) \oplus_q (P_1 +_p P_3) \succ\!\!\rightarrow^*_{(1-q) \cdot r_1 \cdot r_3} (P_1' +_p P_3') \end{cases}$$

$$P_i \succ\!\!\rightarrow^*_{r_i} P_i' \Longrightarrow \left(P_2 \oplus_q P_3 \succ\!\!\rightarrow^*_{q \cdot r_2} P_2'\right) \ \wedge\ \left(P_2 \oplus_q P_3 \succ\!\!\rightarrow^*_{(1-q) r_2} P_3'\right)$$

$$\Longrightarrow \begin{cases} P_1 +_p (P_2 \oplus_q P_3) \succ\!\!\rightarrow^*_{r_1 \cdot q \cdot r_2} (P_1' +_p P_2') \\ P_1 +_p (P_2 \oplus_q P_3) \succ\!\!\rightarrow^*_{r_1 \cdot (1-q) \cdot r_3} (P_1' +_p P_3') \end{cases}$$

From the previous results we obtain

$$(P_1 +_p P_2) \oplus_q (P_1 +_p P_3) \succ\!\!\rightarrow^*_r (P_1' +_p P_2') \text{ iff } P_1 +_p (P_2 \oplus_q P_3) \succ\!\!\rightarrow^*_r (P_1' +_p P_2')$$
$$(P_1 +_p P_2) \oplus_q (P_1 +_p P_3) \succ\!\!\rightarrow^*_r (P_1' +_p P_3') \text{ iff } P_1 +_p (P_2 \oplus_q P_3) \succ\!\!\rightarrow^*_r (P_1' +_p P_3')$$

For any $i \in \{1,2,3\}$, let us consider the following multisets of pairs (process, probability): $\tilde{P}_i = \{\!\!\{ (P_i', r_i) \mid P_i \succ\!\!\rightarrow^*_{r_i} P_i' \}\!\!\}$. If we combine the previous results we obtain

$$\begin{aligned} & pass((P_1 +_p P_2) \oplus_q (P_1 +_p P_3)\,,\,T) \\ & \quad = \textstyle\sum \{\!\!\{ q \cdot r_1 \cdot r_2 \cdot pass(P_1' +_p P_2'\,,\,T) \mid (P_1', r_1) \in \tilde{P}_1 \ \wedge\ (P_2', r_2) \in \tilde{P}_2 \}\!\!\} \\ & \quad + \textstyle\sum \{\!\!\{ (1-q) \cdot r_1 \cdot r_3 \cdot pass(P_1' +_p P_3'\,,\,T) \mid (P_1', r_1) \in \tilde{P}_1 \ \wedge\ (P_3', r_3) \in \tilde{P}_3 \}\!\!\} \\ & \quad = \textstyle\sum \{\!\!\{ r_1 \cdot q \cdot r_2 \cdot pass(P_1' +_p P_2'\,,\,T) \mid (P_1', r_1) \in \tilde{P}_1 \ \wedge\ (P_2', r_2) \in \tilde{P}_2 \}\!\!\} \\ & \quad + \textstyle\sum \{\!\!\{ r_1 \cdot (1-q) \cdot r_3 \cdot pass(P_1' +_p P_3'\,,\,T) \mid (P_1', r_1) \in \tilde{P}_1 \ \wedge\ (P_3', r_3) \in \tilde{P}_3 \}\!\!\} \\ & \quad = pass(P_1 +_p (P_2 \oplus_q P_3)\,,\,T) \end{aligned}$$

Now we show soundness of (**EBE**). First we introduce some notation. Given a probabilistic barb T, let $\tilde{T} = \{t_1, \ldots, t_u\}$ be the set of its initial actions.

Given a set of actions $C \subseteq Act$, the set $\{t_i \mid t_i \in C \cap \tilde{T}\}$ is denoted by T_C. Let us consider the following processes:

$$P = \sum_{i=1}^{n} [p_i]\, a_i; P_i \qquad Q = \sum_{j=1}^{m} [q_j]\, b_j; Q_j \qquad R = \sum_{k=1}^{l} [r_k]\, c_k; R_k$$

where for any $1 \leq k \leq l$ we have that r_k, c_k, and R_k are given in the presentation of axiom (**EBE**). Let $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$. In the following, $p(P, a_i)$ stands for p_i while $p(Q, b_j)$ stands for q_j. Applying rules $(EXT4)$ and $(EXT5)$ we have that $P \xrightarrow{a}_q P'$ implies $P +_p Q \xrightarrow{a}_{p \cdot q} P'$ and $Q \xrightarrow{a}_q P'$ implies $P +_p Q \xrightarrow{a}_{(1-p) \cdot q} P'$. We will show that for any probabilistic barb $T \in \mathcal{PB}$ we have $pass(P +_p Q\ ,\ T) = pass(R\ ,\ T)$.

If T is a probabilistic barb as $T = \sum_{i=1}^{u} [s_i]\ (t_i; Nil) +_s \omega$ then we have

$pass(P +_p Q\ ,\ T) =$

$$\frac{1-s}{(1-s) + \sum\limits_{t_i \in T_A} s \cdot s_i \cdot p \cdot p(P,t_i) \ + \sum\limits_{t_i \in T_B} s \cdot s_i \cdot (1-p) \cdot p(Q,t_i)} =$$

$$\frac{1-s}{(1-s) + \sum\limits_{t_i \in T_{A-B}} s \cdot s_i \cdot p \cdot p(P,t_i) + \sum\limits_{t_i \in T_{B-A}} s \cdot s_i \cdot (1-p) \cdot p(Q,t_i) + \sum\limits_{t_i \in T_{A \cap B}} s \cdot s_i \cdot (p \cdot p(P,t_i) + (1-p) \cdot p(Q,t_i))} =$$

$pass(R\ ,\ T)$

Let T be a probabilistic barb as $T = \sum_{i=1}^{u} [s_i]\ t_i; T_i$, where if $i = u$ then $T_i = T'$ and $T_i = Nil$ otherwise. We distinguish four cases:

$\diamond\ \exists\ 1 \leq i \leq n :\ a_i = t_u \in A - B.$

$pass(P +_p Q\ ,\ T) =$

$$\frac{s_u \cdot p \cdot p_i \cdot\ pass(P_i\ ,T')}{\sum\limits_{t_i \in T_A} s_i \cdot p \cdot p(P,t_i) \ + \sum\limits_{t_i \in T_B} s_i \cdot (1-p) \cdot p(Q,t_i)} =$$

$$\frac{s_u \cdot p \cdot p_i \cdot\ pass(P_i\ ,T')}{\sum\limits_{t_i \in T_{A-B}} s_i \cdot p \cdot p(P,t_i) \ + \sum\limits_{t_i \in T_{B-A}} s_i \cdot (1-p) \cdot p(Q,t_i) \ + \sum\limits_{t_i \in T_{A \cap B}} s_i \cdot (p \cdot p(P,t_i) + (1-p) \cdot p(Q,t_i))} =$$

$pass(R\ ,\ T)$

$\diamond\ \exists\ 1 \leq j \leq m :\ b_j = t_u \in B - A$. Symmetrical to the previous case.

$\diamond\ \exists\ 1 \leq i \leq n,\ 1 \leq j \leq m :\ a_i = b_j = t_u \in A \cap B.$

$$pass(P +_p Q\ ,\ T)\ =$$

$$\frac{s_u{\cdot}p{\cdot}p_i{\cdot}\ pass(P_i\ ,T') + s_u{\cdot}(1{-}p){\cdot}q_j{\cdot}\ pass(Q_j\ ,T')}{\sum\limits_{t_i \in T_A} s_i{\cdot}p{\cdot}p(P,t_i)\ +\ \sum\limits_{t_i \in T_B} s_i{\cdot}(1{-}p){\cdot}p(Q,t_i)} =$$

$$\frac{s_u{\cdot}(p{\cdot}p_i{+}(1{-}p){\cdot}q_j){\cdot}\ pass(P_i \oplus_{\frac{p\cdot p_i}{p\cdot p_i+(1-p)\cdot q_j}} Q_j\ ,T')}{\sum\limits_{t_i \in T_{A-B}} s_i{\cdot}p{\cdot}p(P,t_i)\ +\ \sum\limits_{t_i \in T_{B-A}} s_i{\cdot}(1{-}p){\cdot}p(Q,t_i)\ +\ \sum\limits_{t_i \in T_{A\cap B}} s_i{\cdot}(p{\cdot}p(P,t_i){+}(1{-}p){\cdot}p(Q,t_i))} =$$

$$pass(R\ ,\ T)$$

$\diamond\ t_u \notin A \cup B$. In this case, $pass(P +_p Q\ ,\ T)\ =\ pass(R\ ,\ T)\ = 0.$

Soundness of (**IBE**). Let us consider the following processes:

$$P = \sum_{i=1}^{n} [p_i]\ a_i; P_i \qquad Q = \sum_{i=1}^{n} [p_i]\ a_i; Q_i \qquad R = \sum_{i=1}^{n} [p_i]\ a_i; (P_i \oplus_p Q_i)$$

Let $A = \{a_1, \ldots, a_n\}$. We will show that for any probabilistic barb T we have $pass(P \oplus_p Q\ ,\ T)\ =\ pass(R\ ,\ T)$. In the following, $p(a_i)$ stands for p_i.

If T is a probabilistic barb as $T = \sum\limits_{i=1}^{u} [s_i]\ (t_i; Nil) +_s \omega$ then

$$pass(P \oplus_p Q\ ,\ T)\ = p \cdot\ pass(P\ ,\ T)\ + (1-p) \cdot\ pass(Q\ ,\ T)$$

$$= \frac{p{\cdot}(1{-}s)}{(1{-}s){+}\sum\limits_{t_i \in T_A} s{\cdot}s_i{\cdot}p(t_i)} + \frac{(1{-}p){\cdot}(1{-}s)}{(1{-}s){+}\sum\limits_{t_i \in T_A} s{\cdot}s_i{\cdot}p(t_i)} = \frac{(1{-}s)}{(1{-}s){+}\sum\limits_{t_i \in T_A} s{\cdot}s_i{\cdot}p(t_i)}$$

$$= pass(R\ ,\ T)$$

Let T be a probabilistic barb as $T = \sum\limits_{i=1}^{u} [s_i]\ t_i; T_i$, where if $i = u$ then $T_i = T'$, and $T_i = Nil$ otherwise. We consider two cases:

$\diamond\ \exists\ 1 \leq i \leq n :\ a_i = t_u.$

$$pass(P \oplus_p Q\ ,\ T)\ = p \cdot\ pass(P\ ,\ T)\ + (1-p) \cdot\ pass(Q\ ,\ T)$$

$$= \frac{p{\cdot}s_u{\cdot}p_i{\cdot}\ pass(P_i\ ,T')}{\sum\limits_{t_i \in T_A} s_i{\cdot}p(t_i)} + \frac{(1{-}p){\cdot}s_u{\cdot}p_i{\cdot}\ pass(Q_i\ ,T')}{\sum\limits_{t_i \in T_A} s_i{\cdot}p(t_i)} = \frac{s_u{\cdot}p_i{\cdot}\ pass(P_i \oplus_p Q_i\ ,T')}{\sum\limits_{t_i \in T_A} s_i{\cdot}p(t_i)}$$

$$= pass(R\ ,\ T)$$

$\diamond\ t_u \notin A$. In this case, $pass(P \oplus_p Q\ ,\ T)\ =\ pass(R\ ,\ T)\ = 0.$

Theorem 5.12 Let $P \in \mathrm{PPA}_{fin}$. There exists a normal form N, such that $\vdash P \equiv N$.

Proof: The proof will be done by structural induction. If $P = Nil$ or $P = \Omega$ then P is already a normal form (just considering the normal forms corresponding to the sets $\mathcal{A} = \{\emptyset\}$ and $\mathcal{A} = \emptyset$, respectively). If $P = a; P'$ then by induction hypothesis P' can be transformed into a normal form N'. Considering $\mathcal{A} = \{\{(a,1)\}\}$, $N_{a,A} = N'$, and applying rule (**C1**) we obtain a normal form N such that $P \equiv N$.

If $P = P_1 \oplus_p P_2$ then, by induction hypothesis, P_1 and P_2 can be transformed into normal forms N_1 and N_2, respectively, such that $P_1 \equiv N_1 \ \wedge \ P_2 \equiv N_2$, where $N_1 = \bigoplus_{A \in \mathcal{A}} [p_A] \sum_{(a,p_a) \in A} [p_a]\, a; P_{a,A}$ and $N_2 = \bigoplus_{B \in \mathcal{B}} [q_B] \sum_{(b,q_b) \in B} [q_b]\, b; Q_{b,B}$. Applying (**C3**) and (**O1-2**) we obtain $P_1 \oplus_p P_2 \equiv N_1 \oplus_p N_2$. By applying (**IBE**), if necessary, and given the fact that any generalized internal choice can be decomposed into binary internal choices and vice versa, we obtain the normal form $N = \bigoplus_{C \in \mathcal{C}} [r_C] \sum_{(c,r_c) \in C} [r_c]\, c; R_{c,C}$, where $\mathcal{C} = \mathcal{A} \cup \mathcal{B}$ and for any $C \in \mathcal{C}$ we have three possibilities:

$$
\begin{array}{lll}
C = A \in \mathcal{A} - \mathcal{B} & \Rightarrow & r_C = p \cdot p_A \ \wedge \ \forall\, c \in C : \ R_{c,C} = P_{c,A} \\
C = B \in \mathcal{B} - \mathcal{A} & \Rightarrow & r_C = (1-p) \cdot q_B \ \wedge \ \forall\, c \in C : \ R_{c,C} = Q_{c,B} \\
C = A = B \in \mathcal{A} \cap \mathcal{B} & \Rightarrow & r_C = p \cdot p_A + (1-p) \cdot q_B \ \wedge \\
 & & \forall\, c \in C : \ R_{c,C} = P_{c,A} \oplus_{\frac{p \cdot p_A}{r_C}} Q_{c,B}
\end{array}
$$

In the first two cases we obtain a normal form. In the third case we can apply induction hypothesis to $P_{c,A}$ and $Q_{c,B}$. Consequently, we get a normal form N such that $N_1 \oplus_p N_2 \equiv N$. Finally, applying rules (**O1-3**), we obtain $P_1 \oplus_p P_2 \equiv N$.

If $P = P_1 +_p P_2$ then, by induction hypothesis, we have $P_1 \equiv N_1 \ \wedge \ P_2 \equiv N_2$, where $N_1 = \bigoplus_{A \in \mathcal{A}} [p_A] \sum_{(a,p_a) \in A} [p_a]\, a; P_{a,A}$ and $N_2 = \bigoplus_{B \in \mathcal{B}} [q_B] \sum_{(b,q_b) \in B} [q_b]\, b; Q_{b,B}$. Applying rules (**C3**) and (**O1-2**) we obtain $P_1 +_p P_2 \equiv N_1 +_p N_2$. Then, applying reiteratively (**DEIG**), we obtain

$$
P' = \bigoplus_{\substack{A \in \mathcal{A} \\ B \in \mathcal{B}}} [p_A \cdot q_B] \left(\Big(\sum_{(a,p_a) \in A} [p_a]\, a; P_{a,A} \Big) +_p \Big(\sum_{(b,q_b) \in B} [q_b]\, b; Q_{b,B} \Big) \right)
$$

Let us study the processes $\big(\sum_{(a,p_a) \in A} [p_a]\, a; P_{a,A} \big) +_p \big(\sum_{(b,q_b) \in B} [q_b]\, b; Q_{b,B} \big)$. If either A or B is empty then, applying axiom (**NE**), we obtain the other process. If both are non-empty then, applying axiom (**EBE**), we obtain that

the previous process is equivalent to $R_{A,B} = \sum_{(c,r_c)\in C} [r_c]\ c; R_{c,A,B}$, where $C = \{(c_1, r_1), \ldots, (c_l, r_l) \mid \exists\ q :\ (c_i, q) \in A \cup B\}$ and

$$r_c = \begin{cases} p \cdot q & \text{if } \exists\ q :\ (c,q) \in A \ \wedge\ \nexists\ r :\ (c,r) \in B \\ (1-p) \cdot q & \text{if } \exists\ q :\ (c,q) \in B \ \wedge\ \nexists\ r :\ (c,r) \in A \\ p \cdot p_1 + (1-p) \cdot p_2 & \text{if } \exists\ p_1, p_2 :\ (c,p_1) \in A\ \wedge\ (c,p_2) \in B \end{cases}$$

$$R_{c,A,B} = \begin{cases} P_{c,A} & \text{if } \exists\ q :\ (c,q) \in A \ \wedge\ \nexists\ r :\ (c,r) \in B \\ Q_{c,B} & \text{if } \exists\ q :\ (c,q) \in B \ \wedge\ \nexists\ r :\ (c,r) \in A \\ P_{c,A} \oplus_{\frac{p\cdot p_1}{p\cdot p_1 + (1-p)\cdot p_2}} Q_{c,B} & \text{if } \exists\ p_1, p_2 :\ (c,p_1) \in A\ \wedge\ (c,p_2) \in B \end{cases}$$

As for the internal choice case, the processes $R_{c,A,B}$ are either a normal form, in the first two cases, or they can be transformed into a normal form by applying induction hypothesis. But the new process is not yet (necessarily) a normal form, because if we consider the processes appearing when composing two generalized external choices by an external choice, we can generate in different ways the set C associated with the process R. That is, given a set C, there can exist several $A \in \mathcal{A}$ and $B \in \mathcal{B}$ such that their combination produces the same set C. In that case, we must join all the generalized external choices having associated with them the same states. The probabilities associated with the actions of the new generalized external choices are computed from the ones in the original sets, by using the function $\cup_p$ given in Definition 4.4, so that applying axiom (**IBE**) we obtain

$$P'' = \bigoplus_{C\in\mathcal{C}} [r_C] \sum_{(c,r_c)\in C} [r_c]\ c; \Big(\bigoplus_{\substack{A\in\mathcal{A},\, B\in\mathcal{B} \\ C = A \cup_p B}} \left[\frac{p_A \cdot q_B}{r_C}\right] R_{c,A,B} \Big)$$

where $\mathcal{C} = \{A \cup_p B \mid A \in \mathcal{A}\ \wedge\ B \in \mathcal{B}\}$, the processes $R_{c,A,B}$ are defined as before, and $r_C = \sum \{\!|\ p_A \cdot q_B \mid \exists\ A \in \mathcal{A}, B \in \mathcal{B} :\ A \cup_p B = C\ |\!\}$. Finally, by induction hypothesis, processes as

$$\bigoplus_{\substack{A\in\mathcal{A},\, B\in\mathcal{B} \\ C = A \cup_p B}} \left[\frac{p_A \cdot q_B}{r_C}\right] R_{c,A,B}$$

can be transformed into a normal form, obtaining from P'' a process P''' such that, applying rules (**O1-3**), we conclude $P_1 +_p P_2 \equiv P'''$, where P''' is already a normal form.

Lemma 5.13 Let $P, Q \in \mathrm{PPA}_{fin}$. Then, $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ implies $P \sqsubseteq Q$.

Proof: Theorem 5.12 states that P and Q can be transformed into normal forms. Thus, we can restrict ourselves to the study of equivalent normal forms. Let us consider P and Q as:

$$P = \bigoplus_{A \in \mathcal{A}} [p_A] \sum_{(a,p_a) \in A} [p_a]\ a; P_{a,A} \qquad Q = \bigoplus_{B \in \mathcal{B}} [q_B] \sum_{(b,q_b) \in B} [q_b]\ b; Q_{b,B}$$

where $\mathcal{A}, \mathcal{B} \subseteq \mathcal{P}(\Sigma \times (0,1])$. Note that $p(P, \epsilon, A) = p_A$ and $p(Q, \epsilon, B) = q_B$. We will do the proof by structural induction on the *complexity* of the processes. By complexity we mean the depth of processes, and if two processes have the same depth, we consider that a process is more complex than another one, if the reachable states of the latter are contained in the ones of the former. We can suppose $P \neq \Omega$ because if $P = \Omega$ then the result is immediate (applying axiom (**D**)). We have three possibilities:

- The sets $\mathcal{A}$ and $\mathcal{B}$ different.

We assume $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$. This implies $\mathcal{A} \subseteq \mathcal{B}$. So, there exists a state B' such that $B' \in \mathcal{B} - \mathcal{A}$ (because $\mathcal{A}$ and $\mathcal{B}$ are different). Moreover, the rest of states belonging to $\mathcal{B}$ must have a probability associated with them in the process Q greater than or equal to the corresponding to P. In particular, we have that the probability of P diverging in its first step is greater than or equal to $q_{B'}$, because

$$\sum_{A \in \mathcal{A}} p_A \leq \sum_{A \in \mathcal{A}} q_A < \sum_{A \in \mathcal{A}} q_A + q_{B'} \leq \sum_{B \in \mathcal{B}} q_B \leq 1$$

We can rewrite, using axiom (**AI**), P and Q as:

$$P \equiv \left(\bigoplus_{A \in \mathcal{A}} [\tfrac{p_A}{1-q_{B'}}] \sum_{(a,p_a) \in A} [p_a]\ a; P_{a,A} \right) \ \oplus_{1-q_{B'}}\ \Omega$$

$$Q \equiv \left(\bigoplus_{B \in \mathcal{B}-B'} [\tfrac{q_B}{1-q_{B'}}] \sum_{(b,q_b) \in B} [q_b]\ b; Q_{b,B} \right) \ \oplus_{1-q_{B'}}\ \left(\sum_{(b',q_{b'}) \in B'} [q_{b'}]\ b'; Q_{b',B'} \right)$$

By applying axiom (**D**), we have $\Omega \sqsubseteq \sum_{(b',q_{b'}) \in B} [q_{b'}]\ b'; Q_{b',B'}$. Again, because $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$, we obtain

$$[\![\bigoplus_{A \in \mathcal{A}} \left[\frac{p_A}{1-q_{B'}}\right] \sum_{(a,p_a) \in A} [p_a]\ a; P_{a,A}]\!] \sqsubseteq_{\mathbf{PAT}} [\![\bigoplus_{B \in \mathcal{B}-B'} \left[\frac{q_B}{1-q_{B'}}\right] \sum_{(b,q_b) \in B} [q_b]\ b; Q_{b,B}]\!]$$

and by induction hypothesis, because the states of the right hand side process are contained in those of Q, we have

$$\bigoplus_{A \in \mathcal{A}} \left[\frac{p_A}{1-q_{B'}}\right] \sum_{(a,p_a) \in A} [p_a]\ a; P_{a,A} \sqsubseteq \bigoplus_{B \in \mathcal{B}-B'} \left[\frac{q_B}{1-q_{B'}}\right] \sum_{(b,q_b) \in B} [q_b]\ b; Q_{b,B}$$

so that applying rule (**C3**) we finally obtain $P \sqsubseteq Q$.

• $\mathcal{A} = \mathcal{B}$ and $\exists\, C :\ p_C \neq q_C$.

Assuming $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ we have $p_A \leq q_A$, for any $A \in \mathcal{A}$. This implies $p_C < q_C$. Performing a distinction similar to the previous case, we can rewrite our processes as:

$$P \equiv (\bigoplus_{A \in \mathcal{A}-C} [\tfrac{p_A}{1-q_C}] \sum_{(a,p_a)\in A} [p_a]\ a; P_{a,A}) \oplus_{1-q_C} ((\sum_{(c,p_c)\in C} [p_c]\ c; P_{c,C}) \oplus_{\frac{p_C}{q_C}} \Omega)$$

$$Q \equiv (\bigoplus_{A \in \mathcal{A}-C} [\tfrac{q_A}{1-q_C}] \sum_{(a,p_a)\in A} [p_a]\ a; Q_{a,A}) \oplus_{1-q_C} ((\sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}) \oplus_{\frac{p_C}{q_C}} (\sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}))$$

Note that by using rules (**OI1**), (**OI2**), and (**O1**) we can infer

$$\vdash (\sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}) \oplus_{\frac{p_C}{q_C}} (\sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}) \equiv (\sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C})$$

Applying axiom (**D**) we have $\Omega \sqsubseteq \sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}$ and given the fact that $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$ we obtain $[\![\sum_{(c,p_c)\in C} [p_c]\ c; P_{c,C}]\!] \sqsubseteq_{\mathbf{PAT}} [\![\sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}]\!]$. So, applying induction hypothesis, we get

$$\sum_{(c,p_c)\in C} [p_c]\ c; P_{c,C} \sqsubseteq \sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}$$

and applying rule (**C3**) we obtain

$$\sum_{(c,p_c)\in C} [p_c]\ c; P_{c,C} \oplus_{\frac{p_C}{q_C}} \Omega \ \sqsubseteq \sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C} \oplus_{\frac{p_C}{q_C}} \sum_{(c,p_c)\in C} [p_c]\ c; Q_{c,C}$$

Once again, given the fact that $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$, we have

$$[\![\bigoplus_{A \in \mathcal{A}-C} [\frac{p_A}{1-q_C}] \sum_{(a,p_a)\in A} [p_a]\ a; P_{a,A}]\!] \sqsubseteq_{\mathbf{PAT}} [\![\bigoplus_{A \in \mathcal{A}-C} [\frac{q_A}{1-q_C}] \sum_{(a,p_a)\in A} [p_a]\ a; Q_{a,A}]\!]$$

and applying induction hypothesis

$$\bigoplus_{A \in \mathcal{A}-C} [\frac{p_A}{1-q_C}] \sum_{(a,p_a)\in A} [p_a]\ a; P_{a,A} \sqsubseteq \bigoplus_{A \in \mathcal{A}-C} [\frac{q_A}{1-q_C}] \sum_{(a,p_a)\in A} [p_a]\ a; Q_{a,A}$$

Finally, applying rule (**C3**) to the previous results, we obtain $P \sqsubseteq Q$.

•$\mathcal{A} = \mathcal{B}$ and $\forall\, C \in \mathcal{A} :\ p_C = q_C$.

We assume $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!]$. Then, for any $A \in \mathcal{A}$ and $(a, p_a) \in A$ we have $[\![P_{a,A}]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q_{a,A}]\!]$. So, applying induction hypothesis, we obtain $P_{a,A} \sqsubseteq Q_{a,A}$. Then, applying axiom (**C1**), we have $a; P_{a,A} \sqsubseteq a; Q_{a,A}$, for any action a and state A. So, applying reiteratively axiom (**C2**), for any A we have

$$\sum_{(a,p_a)\in A} [p_a]\, a; P_{a,A} \sqsubseteq \sum_{(a,p_a)\in A} [p_a]\, a; Q_{a,A}$$

Finally, if we reiteratively apply axiom (**C3**) we obtain the desired result

$$P = \bigoplus_{A\in\mathcal{A}} [p_A] \sum_{(a,p_a)\in A} [p_a]\, a; P_{a,A} \sqsubseteq \bigoplus_{A\in\mathcal{A}} [p_A] \sum_{(a,p_a)\in A} [p_a]\, a; Q_{a,A} = Q$$

Lemma 5.21 Let $P \in$ PPA be a finite process and $Q \in$ PPA be a recursive one. Then, $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!] \Longrightarrow\ P \sqsubseteq Q$.

Proof: Suppose $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q]\!] = \sqcup [\![Q^n]\!]$. If there exists $m \in \mathbb{N}$ such that $[\![P]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q^m]\!]$ then the proof can be done as indicated after the presentation of Lemma 5.19 in the body of the paper.

Let us suppose that there does not exist such an m. For any sequence s and state A we have $p([\![P]\!], s, A) \leq p([\![Q]\!], s, A) = \lim_n p([\![Q^n]\!], s, A)$. Let us consider those sequences s and those states A such that $p([\![P]\!], s, A) > 0$. We have $(1 - \frac{1}{k}) \cdot p([\![P]\!], s, A) < \lim_n p([\![Q^n]\!], s, A)$, for any $k > 0$. Note that $(1 - \frac{1}{k}) \cdot p([\![P]\!], s, A) = p([\![P \oplus_{1-\frac{1}{k}} \Omega]\!], s, A)$. Given the fact that P is finite, we have that the set of pairs (s, A) such that $p([\![P]\!], s, A) > 0$ is finite. So, for each $k \in \mathbb{N}$ there exists $n_k \in \mathbb{N}$ such that $p([\![P \oplus_{1-\frac{1}{k}} \Omega]\!], s, A) \leq p([\![Q^{n_k}]\!], s, A)$, for any sequence s and state A such that $p([\![P]\!], s, A) > 0$. If $p([\![P]\!], s', A') = 0$ then the previous result also holds, so that we have $[\![P \oplus_{1-\frac{1}{k}} \Omega]\!] \sqsubseteq_{\mathbf{PAT}} [\![Q^{n_k}]\!]$. Given the fact that the processes $P \oplus_{1-\frac{1}{k}} \Omega$ and Q^{n_k} are finite, we can apply Lemma 5.13. So, $P \oplus_{1-\frac{1}{k}} \Omega \sqsubseteq Q^{n_k}$, for any $k \in \mathbb{N}$. Considering that for any $n \in \mathbb{N}$ we have $Q^n \sqsubseteq Q$, we deduce $P \oplus_{1-\frac{1}{k}} \Omega \sqsubseteq Q$ for any $k \in \mathbb{N}$. Finally, applying rule (**R3**), we obtain $P \sqsubseteq Q$.

Proposition 6.5 The axiom (**EP**) is sound.

Proof: First we introduce some additional notation. For a probabilistic barb T, let $\tilde{T} = \{t_1, \ldots, t_u\}$ be the set of its initial actions. Given a set of actions $C \subseteq Act$, the set $\{t_i \mid t_i \in C \cap \tilde{T}\}$ is denoted by T_C. Finally, we consider $p_i = p(P, a_i)$ and $q_j = p(Q, b_j)$.

Let us note that if $a \notin X$ then we have both that $P \stackrel{a}{\longrightarrow}_q P'$ implies $P \parallel^p_X Q \stackrel{a}{\longrightarrow}_{p_1} P' \parallel^p_X Q$ and that $Q \stackrel{a}{\longrightarrow}_q Q'$ implies $P \parallel^p_X Q \stackrel{a}{\longrightarrow}_{p_2} P \parallel^p_X Q'$, where $p_1 = \frac{p \cdot q}{\mu(P,Q,X,p)}$ and $p_2 = \frac{(1-p)\cdot q}{\mu(P,Q,X,p)}$. Besides, if $a \in X$ then we have that $P \stackrel{a}{\longrightarrow}_{p_1} P' \ \wedge\ Q \stackrel{a}{\longrightarrow}_{p_2} Q'$ implies $P \parallel^p_X Q \stackrel{a}{\longrightarrow}_{\frac{p_1 \cdot p_2}{\mu(P,Q,X,p)}} P' \parallel^p_X Q'$.

We will prove that $pass(P \parallel^p_X Q\ ,\ T) = pass(R\ ,\ T)$ for any $T \in \mathcal{PB}$.

If T is a probabilistic barb as $T = \sum_{i=1}^{u} [s_i]\ (t_i; Nil) +_s \omega$ then we deduce

$$pass(P \parallel_X^p Q\ ,\ T)\ =$$

$$\frac{1-s}{(1-s)+\sum\limits_{t_i \in T_{A-X}} s \cdot s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)} + \sum\limits_{t_i \in T_{B-X}} s \cdot s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)} + s \cdot R_{A \cap B \cap X}} =$$

$$\frac{1-s}{(1-s)+\sum\limits_{t_i \in T_{(A-B)-X}} s \cdot s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)} + \sum\limits_{t_i \in T_{(B-A)-X}} s \cdot s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)} + s \cdot R_{(A \cap B)-X} + s \cdot R_{A \cap B \cap X}} =$$

$$pass(R\ ,\ T)$$

where we have considered that $R_{(A \cap B)-X} = \sum\limits_{t_i \in T_{(A \cap B)-X}} s_i \cdot \frac{(p \cdot p(P,t_i) + (1-p) \cdot p(Q,t_i))}{\mu(P,Q,X,p)}$ and $R_{A \cap B \cap X} = \sum\limits_{t_i \in T_{A \cap B \cap X}} s_i \cdot \frac{p(P,t_i) \cdot p(Q,t_i)}{\mu(P,Q,X,p)}$

Let T be a probabilistic barb as $T = \sum_{i=1}^{u} [s_i]\ t_i; T_i$, where if $i = u$ then $T_i = T'$, and $T_i = Nil$ otherwise. We must consider several cases depending on the different sets of actions to which t_u may belong:

$\diamond\ \exists\ 1 \leq i \leq n :\ a_i = t_u \in (A - B) - X.$

$$pass(P \parallel_X^p Q\ ,\ T)\ =$$

$$\frac{s_u \cdot \frac{p \cdot p_i}{\mu(P,Q,X,p)} \cdot pass(P_i \parallel_X^p Q\ , T')}{\sum\limits_{t_i \in T_{A-X}} s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)} + \sum\limits_{t_i \in T_{B-X}} s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)} + R_{A \cap B \cap X}} =$$

$$\frac{s_u \cdot \frac{p \cdot p_i}{\mu(P,Q,X,p)} \cdot pass(P_i \parallel_X^p Q\ , T')}{\sum\limits_{t_i \in T_{(A-B)-X}} s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)} + \sum\limits_{t_i \in T_{(B-A)-X}} s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)} + R_{(A \cap B)-X} + R_{A \cap B \cap X}} =$$

$$pass(R\ ,\ T)$$

$\diamond\ \exists\ 1 \leq j \leq m :\ b_j = t_u \in (B - A) - X.$ Symmetrical to the previous case.

$\diamond\ \exists\ 1 \leq i \leq n,\ 1 \leq j \leq m :\ a_i = b_j = t_u \in (A \cap B) - X.$

$$pass(P \parallel^p_X Q\ ,\ T)\ =$$

$$\frac{s_u \cdot \frac{p \cdot p_i}{\mu(P,Q,X,p)} \cdot\ pass(P_i \parallel^p_X Q\ , T') + s_u \cdot \frac{(1-p) \cdot q_j}{\mu(P,Q,X,p)} \cdot\ pass(P \parallel^p_X Q_j\ , T')}{\sum\limits_{t_i \in T_{A-X}} s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)}\ +\ \sum\limits_{t_i \in T_{B-X}} s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)}\ + R_{A \cap B \cap X}} =$$

$$\frac{s_u \cdot \frac{p \cdot p_i + (1-p) \cdot q_j}{\mu(P,Q,X,p)} \cdot\ pass((P_i \parallel^p_X Q) \oplus_{q'} (P \parallel^p_X Q_j)\ , T')}{\sum\limits_{t_i \in T_{(A-B)-X}} s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)}\ +\ \sum\limits_{t_i \in T_{(B-A)-X}} s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)}\ + R_{(A \cap B)-X} + R_{A \cap B \cap X}} =$$

$$pass(R\ ,\ T)$$

where $q' = \frac{p \cdot p_i}{p \cdot p_i + (1-p) \cdot q_j}$.

$\diamond\ \exists\ 1 \leq i \leq n,\ 1 \leq j \leq m :\ a_i = b_j = t_u \in A \cap B \cap X.$

$$pass(P \parallel^p_X Q\ ,\ T)\ =$$

$$\frac{s_u \cdot \frac{p_i \cdot q_j}{\mu(P,Q,X,p)} \cdot\ pass(P_i \parallel^p_X Q_j\ , T')}{\sum\limits_{t_i \in T_{A-X}} s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)}\ +\ \sum\limits_{t_i \in T_{B-X}} s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)}\ + R_{A \cap B \cap X}} =$$

$$\frac{s_u \cdot \frac{p_i \cdot q_j}{\mu(P,Q,X,p)} \cdot\ pass(P_i \parallel^p_X Q_j\ , T')}{\sum\limits_{t_i \in T_{(A-B)-X}} s_i \cdot \frac{p \cdot p(P,t_i)}{\mu(P,Q,X,p)}\ +\ \sum\limits_{t_i \in T_{(B-A)-X}} s_i \cdot \frac{(1-p) \cdot p(Q,t_i)}{\mu(P,Q,X,p)}\ + R_{(A \cap B)-X} + R_{A \cap B \cap X}} =$$

$$pass(R\ ,\ T)$$

$\diamond\ t_u \notin ((A \cup B) - X) \cup (A \cap B \cap X)$. In this case, $pass(P \parallel^p_A Q\ ,\ T)\ = 0 = pass(R\ ,\ T)$.