Reasoning about Probabilistic and Nondeterministic Processes *

Diego Cazorla, Fernando Cuartero, Valentín Valero, Fernando L. Pelayo

Escuela Politécnica Superior de Albacete
Universidad de Castilla-La Mancha
02071 Albacete, Spain
{dcazorla, fernando, valentin, fpelayo}@info-ab.uclm.es

Abstract. In this paper we present an algebraic language for the specification of probabilistic and nondeterministic processes, PNAL, which is a probabilistic extension of finite EPL (EPL_f) that maintains nondeterminism.
We have defined both an operational and a testing semantics for PNAL, as well as a denotational semantics and a proof system. We show that the equivalence between nondeterministic processes in EPL_f is preserved, and we also prove that the testing semantics, the denotational semantics and the proof system are fully abstract.

1 Introduction

The fundamentals of process algebras were established in the eighties but only qualitative aspects of concurrent systems were considered in these pioneering works. Subsequently quantitative aspects have been proposed, and several extensions to classical process algebra models have been developed including time, priorities and probabilities.

Severeal models for probabilistic processes have been presented in [18, 7, 12, 3, 4, 15]. Nevertheless, the main idea in all these models consists of substituting a nondeterministic behaviour by a probabilistic one in which the different types of behaviours are quantified with probabilities. A precise knowledge of the probabilities associated with each behaviour is therefore required to define a process. However, in practice, this probability distribution could be completely or partially unknown.

It is also necessary therefore to provide operators that allow us to deal with nondeterminism, but, at the same time, taking into account the probabilistic aspects. In the process of modelling a concurrent system we sometimes find nondeterministic behaviours that can be quantified (for instance, the probability of a faulty communication channel failing), whereas on other occasions these nondeterministic behaviours cannot be quantified (for instance, relative speed between

* This work has been partially supported by the CICYT project "Performance Evaluation of Distributed Systems", TIC2000-0701-c02-02.

two processes). This is why we need a language that allows us to model nondeterministic systems with or without quantification, depending on the nature of the system in question.

In line with these ideas, in [21] a probabilistic extension of CCS is presented. The language adds a probabilistic choice operator ($\oplus_p$) indexed with a probability $p \in (0, 1)$, which models the probabilistic internal choice, whereas the original CCS choice (+) is maintained for representing nondeterminism. A testing semantics, based on De Nicola and Hennessy testing, is presented. This work is taken further in [10, 11], where these preorders are redefined and denotationals characterization for may–testing and must–testing are presented.

In [13] a probabilistic extension of CSP (PCSP) that maintains nondeterminism is introduced. The model is provided with an operational semantics, and several denotational semantics are given, based on different kinds of observations (trace, broom and barb semantics), where semantic objects are trees with three kinds of nodes: probabilistic, nondeterministic and action nodes.

Another probabilistic extension of CSP is presented in [14], where a denotational semantics of CSP is defined by applying the probabilistic powerdomain construction of Jones and Plotkin [9] over a directed–complete partial order. Probabilistic processes are considered to be probability distributions over processes of CSP.

Based on ACP, we can cite [1]. In that paper, a probabilistic version of ACP is presented leading to a language that combines probability and nondeterminism. An operational semantics is defined based on the alternating model [7]. In the construction of the term models they use a probabilistic bisimulation showing soundness and completeness of the term model with respect to the proposed axiom systems.

Different approaches, in the field of probabilistic automaton and Markov decision processes, are considered in [17, 16, 19] where the separation between nondeterministic and probabilistic behaviour is achieved by means of adversaries, schedulers or policies, which resolve the nondeterminism.

The specific aim of this paper is to add probabilistic choice to Hennessy's $\mathrm{EPL_f}$ [8], to show how the semantics of that can be defined, and to investigate the resulting process algebra.

The paper is structured as follows. Section 2 introduces our language. In Sections 3 and 4 an operational semantics and a testing semantics are presented. In Section 5 a denotational semantics is given by means of finite probabilistic acceptance trees and we prove that the denotational semantics is fully abstract with respect to the testing semantics. In section 6 a sound and complete proof system is presented. Finally, in section 7 we evaluate the work and compare it with other approaches.

2 PNAL language

The syntax of the language we present, PNAL, is based on $\mathrm{EPL_f}$ (EPL without recursion) [8] and provides three kinds of choice: classical internal choice, denoted by $\oplus$, classical external choice, denoted by $+$, and a probabilistic choice labelled by a probability p: $\otimes_p$.

Let Δ be a set of input actions, $\overline{\Delta} = \{\overline{a} \mid a \in \Delta\}$ their corresponding output actions, and $\mathcal{A}ct = \Delta \cup \overline{\Delta}$. Furthermore we assume a set of process variables Id ranged over by X and Y. Terms of PNAL are defined by:

$$P ::= \mathbf{0} \mid X \mid a.P \mid P \oplus P \mid P + P \mid P \otimes_p P \mid P|P \mid P\backslash A$$

where $a \in \mathcal{A}ct$, $p \in (0,1)$, $X \in Id$, and $A \subset \Delta$.

The classical operators are interpreted in the usual fashion and the probabilistic choice has a *generative* interpretation, i.e., $P \otimes_p Q$ behaves like P with probability p, and like Q with probability $1-p$, this decision been made internally.

In order to deal with probabilities and nondeterminism we have to decide how to solve a situation where both choices appear. In the literature available we have, basically, two alternatives: in [1, 14, 21] probabilistic choices are solved first, whereas in [13] the opposite approach is taken. Note that this decision is not at all meaningless, because, depending on the approach adopted, some properties of classical process algebras will be preserved or not. For example, idempotency of internal choice is only maintained when internal choice is resolved first. Idempotency of external choice is never maintained, because if we take $P = a \otimes_{0.5} b$, we find that $P \,\square\, P$ increments the probability of a, for any model we consider.

In all these models we can identify two kinds of nondeterminism: nondeterminism introduced by internal choice (syntactic nondeterminism), and nondeterminism introduced by external choice and parallel composition between processes prefixed by the same action. Let us consider the following example (using PNAL syntax):

$$(a.P \mid (a.Q \otimes_p b)) \oplus R$$

If we consider Lowe's approach [13], we first resolve internal choice and later (considering the left hand side process) probabilistic choice; at this point we find a nondeterministic behaviour introduced by the process $a.P|a.Q$, and the resolution of this nondeterminism is delayed after the execution of action a. But, in Lowe's words *(sic)* "it is not enough to resolve all nondeterminism in a given state before the probabilistic choices; we have to resolve all nondeterminism before *any* probabilistic choice ... can we find a model that is (faithful to our intuition)?.Such a model would need to ensure that all nondeterministic branching occurs before any other branching, but it is far from obvious how to write down the semantic equations so as to achieve this". This is still, indeed, an open problem.

If we consider the approach taken by Morgan *et al.* [14], probabilistic choice is resolved first, and later on both kinds of nondeterminism can be resolved. Moreover, in that model we find the opposite behaviour, i.e., every probabilistic choice must be resolved before any internal choice. But, in the model in question, probabilistic choice is distributed through all the other operators, so this aim can be achieved easily.

Taking into account both models [13, 14], and the study of asynchronous probabilistic parallel composition introduced in [5] (where "bundle probabilistic transition systems" are introduced), we consider that the more suitable approach is to resolve probabilistic choice first, and then resolve any kind of nondeterminism later on. As we will see in the following sections, this decision leads us to a model that shares some characteristics with the Morgan *et al.* [14] model (e.g., distribution properties of probabilistic choice).

3 Operational semantics

The operational semantics is defined in the usual way, by using a labelled transition system. In order to understand these rules we require a preliminary definition:

Definition 1 *(Probabilistic stability)*
We define in an inductive way the probabilistic stability predicate

i) $\mathbf{0}$ *and* $a.P$ *are probabilistically stable.*
ii) $P \oplus Q$, $P + Q$, $P|Q$ *are probabilistically stable iff both* P *and* Q *are.*
iii) $P \backslash A$ *is probabilistically stable iff* P *is.*

We use $P\downarrow$ *to denote that* P *is probabilistically stable.*

Transition rules are divided into 3 tables. Table 1 defines the probabilistic choice operator behaviour, by using probabilistic transitions, like $P \longrightarrow_p Q$, which means that P may evolve to Q with probability p immediately. Rules P1a and P1b indicate that process $P \otimes_p Q$ behaves like P with probability p and like Q with probability $1 - p$. The remaining rules establish the precedence of the probabilistic choice with respect to the other operators. Processes P and Q are supposed to be probabilistically independent processes in rules P2c, P3c, and P4c, like in [1, 7, 21].

The behaviour of the remaining operators is presented in Tables 2 and 3, where we assume that processes are now probabilistically stable. The second table defines the rules for unlabelled transitions, i.e., those representing an internal evolution, $P \longrightarrow Q$, with the usual meaning that P may evolve internally to Q without performing any visible action. Rules I1a and I1b show how the internal choice operator behaves. Rules I2a, I2b, I3a and I3b indicate that internal choice distributes over the external choice and parallel composition. Rule I3c indicates that an unbarred action may synchronize with its homologous barred action and the obtained process evolves internally. Finally, rule I4 shows the operational behaviour for the restriction operator.

The third table defines the inference rules for transitions labelled with actions in $\mathcal{A}ct$, $P \xrightarrow{a} Q$, with the usual meaning that P can perform and a and move to Q.

Definition 2 *The operational semantics of PNAL is defined as the multiset of transitions we can derive by using the rules in Tables 1–3.*

Throughout the paper we will use $\rightsquigarrow$ instead of $\longrightarrow_p$, $\longrightarrow$ and $\xrightarrow{a}$ in order to represent any of these transitions.

P1a $\dfrac{}{P \otimes_p Q \longrightarrow_p P}$	**P1b** $\dfrac{}{P \otimes_p Q \longrightarrow_{1-p} Q}$	**P2a** $\dfrac{P \longrightarrow_p P', Q\!\downarrow}{P \oplus Q \longrightarrow_p P' \oplus Q}$
P2b $\dfrac{P\!\downarrow, Q \longrightarrow_q Q'}{P \oplus Q \longrightarrow_q P \oplus Q'}$	**P2c** $\dfrac{P \longrightarrow_p P', Q \longrightarrow_q Q'}{P \oplus Q \longrightarrow_{p \cdot q} P' \oplus Q'}$	**P3a** $\dfrac{P \longrightarrow_p P', Q\!\downarrow}{P + Q \longrightarrow_p P' + Q}$
P3b $\dfrac{P\!\downarrow, Q \longrightarrow_q Q'}{P + Q \longrightarrow_q P + Q'}$	**P3c** $\dfrac{P \longrightarrow_p P', Q \longrightarrow_q Q'}{P + Q \longrightarrow_{p \cdot q} P' + Q'}$	**P4a** $\dfrac{P \longrightarrow_p P', Q\!\downarrow}{P \mid Q \longrightarrow_p P' \mid Q}$
P4b $\dfrac{P\!\downarrow, Q \longrightarrow_q Q'}{P \mid Q \longrightarrow_q P \mid Q'}$	**P4c** $\dfrac{P \longrightarrow_p P', Q \longrightarrow_q Q'}{P \mid Q \longrightarrow_{p \cdot q} P' \mid Q'}$	**P5** $\dfrac{P \longrightarrow_p P'}{P \backslash A \longrightarrow_p P' \backslash A}$

Table 1. Probabilistic transitions

I1a $\dfrac{}{P \oplus Q \longrightarrow P}$	**I1b** $\dfrac{}{P \oplus Q \longrightarrow Q}$	**I2a** $\dfrac{P \longrightarrow P'}{P + Q \longrightarrow P' + Q}$
I2b $\dfrac{Q \longrightarrow Q'}{P + Q \longrightarrow P + Q'}$	**I3a** $\dfrac{P \longrightarrow P'}{P \mid Q \longrightarrow P' \mid Q}$	**I3b** $\dfrac{Q \longrightarrow Q'}{P \mid Q \longrightarrow P \mid Q'}$
I3c $\dfrac{P \xrightarrow{a} P', Q \xrightarrow{\overline{a}} Q'}{P \mid Q \longrightarrow P' \mid Q'}$	**I4** $\dfrac{P \longrightarrow P'}{P \backslash A \longrightarrow P' \backslash A}$	

Table 2. Unlabelled transitions

A1 $\dfrac{}{a.P \xrightarrow{a} P}$	**A2a** $\dfrac{P \xrightarrow{a} P'}{P + Q \xrightarrow{a} P'}$	**A2b** $\dfrac{Q \xrightarrow{a} Q'}{P + Q \xrightarrow{a} Q'}$
A3a $\dfrac{P \xrightarrow{a} P'}{P \mid Q \xrightarrow{a} P' \mid Q}$	**A3b** $\dfrac{Q \xrightarrow{a} Q'}{P \mid Q \xrightarrow{a} P \mid Q'}$	**A4** $\dfrac{P \xrightarrow{a} P'}{P \backslash A \xrightarrow{a} P' \backslash A}$ $(a, \overline{a} \notin A)$

Table 3. Transitions labelled with actions

4 Testing semantics

A test is just a process which additionally may perform a special action w to report a successful state. A purely probabilistic process P will pass a test T with a certain probability p. But, when dealing with nondeterministic processes, i.e., including internal choices, we will have to compute a set of probabilities to pass each test, in order to capture all the possibilities to do that.

As usual, in order to relate two processes using the testing semantics, we take one of them as a specification and the other one as its implementation. We will say that a process is a good implementation of a specification if the range of probabilities with which the implementation passes every test is included in the range of probabilities with which the specification passes the same test.

We use $P\|T$ to stand for $(P|T)\backslash\mathcal{A}ct$. $P\|T$ is a process that can only perform probabilistic transitions $\longrightarrow_p$, unlabelled transitions $\longrightarrow$ or $w-$transitions $\xrightarrow{w}$. These processes are called test processes and will be represented by S, S', etc.

Given a process P and a test T, $\phi_{(P,T)}$ will represent the set of probabilities for accepting T considering each possible evolution of $P\|T$. In order to define $\phi_{(P,T)}$ we need some preliminary definitions, following similar ideas to those presented in [21].

Definition 3 *A resolution R of $P\|T$ is a maximal subtree of $P\|T$ such that, whenever $S \hookrightarrow S_1$, $S \hookrightarrow S_2$, we have $S_1 = S_2$, where $\hookrightarrow$ represents either $\longrightarrow$ or $\xrightarrow{w}$.*

Therefore, every nondeterministic node in a resolution has a unique output arc.

Definition 4 *A computation C of a test process S is a maximal sequence of transitions, $S \leadsto S_1 \dots S_{i-1} \leadsto S_i \dots$ where $\leadsto$ represents either $\longrightarrow_p$, $\longrightarrow$ or $\xrightarrow{w}$.*

We will use the following notation: $\mathcal{R}(S)$ denotes the set of resolutions of S, $\mathcal{C}(R)$ denotes the set of computations of resolution R, $Pr(C)$ denotes the probability to reach a successful state along the computation C, and $Pr(R)$ denotes the probability to reach a successful state in the resolution R.

Definition 5 *A computation $C = S \leadsto S_1 \cdots\ S_{i-1} \leadsto S_i \dots \leadsto S_n$ is a successful computation if $S_n \xrightarrow{w}$ and $S_k \stackrel{w}{\not\longrightarrow}$ for all $k < n$. We use $\widehat{\mathcal{C}}(R)$ to denote the set of successful computations of R. We may compute the probability for C to be a successful computation as follows:*

$$\begin{array}{l} Pr(S_n \xrightarrow{w} S_{n+1}) = 1 \ \textit{(success)} \\ Pr(S_i \longrightarrow C) = Pr(C) \\ Pr(S_i \longrightarrow_p C) = p \cdot Pr(C) \end{array}$$

And the probability to get a success in a resolution: $Pr(R) = \displaystyle\sum_{C \in \widehat{\mathcal{C}}(R)} Pr(C)$

Definition 6 *Given a process P and a test T, we say that P* may pass *T with the set of probabilities $\phi_{(P,T)} = \{Pr(R) \mid R \in \mathcal{R}(P\|T)\}$.*

Now we may introduce the *test* preorder over processes, by using $\phi_{(P,T)}$.

Definition 7 *We define the test preorder as follows:*

$$\begin{array}{l} P \sqsubseteq_{test} Q \ \textit{if}\ \forall T \ \textit{s.t.}\ \phi_{(P,T)} \neq \{0\},\ \textit{then}\ \phi_{(Q,T)} \neq \{0\}, \\ \qquad\qquad \sup\{\phi_{(P,T)}\} \geq \sup\{\phi_{(Q,T)}\}, \textit{and}\ \inf\{\phi_{(P,T)}\} \leq \inf\{\phi_{(Q,T)}\} \end{array}$$

This preorder impose that whenever Q cannot accept a test T, P cannot accept it either. This has been introduced because, according to the definition of test preorder given in [10], the processes $P = a \oplus c$ and $Q = a$ are related under the test preorder, i.e., $P \sqsubseteq_{test} Q$. On the contrary in Hennessy's testing P *may* $c.w$ but Q $\cancel{may}$ $c.w$ so $P \not\sqsubseteq_{may} Q$. According to our definition both situations are captured.

The *test* relation has the following interpretation: $P \sqsubseteq_{test} Q$ if Q is a good implementation of the specification P. The corresponding equivalence is defined as follows:

Definition 8 *$P \equiv_{test} Q$ if and only if $P \sqsubseteq_{test} Q$ and $Q \sqsubseteq_{test} P$*

5 Denotational semantics

The aim of this section is to provide our language with a fully abstract denotational semantics with respect to the testing semantics. Our domain will be a set of finite probabilistic acceptance trees, fPAT. Let us informally describe the structure of these trees:

- We will consider trees with two kinds of nodes: probabilistic ($\otimes$), and nondeterministic ($\bullet$).
- The root node is a probabilistic one and probabilistic and nondeterministic nodes appear in an alternative way.
- Nondeterministic nodes are labelled with an acceptance set (as defined in [8]), while probabilistic nodes are unlabelled.
- Branches leaving a probabilistic node are labelled with probabilities p_i, with $\sum_i p_i = 1$. Every branch leaving a nondeterministic node is labelled with one action in the acceptance set, but there are not two branches labelled with the same action.
- Terminal nodes are always nondeterministic, and are labelled with the acceptance set $\{\emptyset\}$.

We can see that these trees are quite similar to those used in [8], but now we have to extend these trees in order to include the probabilistic information associated with every reachable state. Let us introduce now acceptance sets and probabilistic acceptance sets.

Definition 9 *An acceptance set $\mathcal{A}$ is a subset of $\mathcal{P}(\mathcal{A}ct)$. Let $\boldsymbol{D}(\mathcal{A})$ be the domain of acceptance sets for a given set of actions $\mathcal{A}ct$. We say that $(p, \mathcal{A})$ is a "probability–acceptance set pair" if $(p, \mathcal{A}) \in (0, 1] \times \boldsymbol{D}(\mathcal{A})$.*

Let $\mathcal{A}$ be an acceptance set. We define $A(\mathcal{A})$, the set of actions belonging to $\mathcal{A}$, as follows:

$$A(\mathcal{A}) = \{a \in \mathcal{A}ct \,|\, \exists B \in \mathcal{A} \text{ s.t. } a \in B\}$$

Definition 10 *A probabilistic acceptance set, $\mathcal{PA}$, is defined as follows:*

$$\mathcal{PA} = \{(p_i, \mathcal{A}_i) \in (0, 1] \times \boldsymbol{D}(\mathcal{A}) \,|\, \sum_i p_i = 1 \wedge \mathcal{A}_i \neq \mathcal{A}_j \text{ if } i \neq j\}$$

The domain of probabilistic acceptance sets is represented by $\boldsymbol{D}(\mathcal{PA})$.

Definition 11 *We define the probability distribution, induced by a probabilistic acceptance set, as the function* $\pi : \boldsymbol{D}(\mathcal{A}) \times \boldsymbol{D}(\mathcal{PA}) \longrightarrow [0,1]$ *given by*

$$\pi(\mathcal{A}, \mathcal{PA}) = \begin{cases} p & \textit{if } (p, \mathcal{A}) \in \mathcal{PA} \\ 0 & \textit{otherwise} \end{cases}$$

Definition 12 *Let* $\mathcal{PA}$ *be a probabilistic acceptance set. We define the set of acceptance sets belonging to* $\mathcal{PA}$ *as follows:*
$Accept(\mathcal{PA}) = \{\mathcal{A} \in \boldsymbol{D}(\mathcal{A}) \mid \exists p \textit{ s.t. } (p, \mathcal{A}) \in \mathcal{PA}\}$

We will use the following notation:
$\overline{C} = \{\overline{a} \mid a \in C\}$ and $\mathcal{A}\backslash C = \{B - (C \cup \overline{C}) \mid B \in \mathcal{A}\}$.

Besides the definitions above, we will need two additional operators borrowed from [8]: operators c and u. Due to the lack of space, we will not give here their formal definitions. Instead, we will remember their informal meaning: c is the convex closure of a set, and u is the pointwise union of sets of sets.

We may now define (see Table 4) the probabilistic acceptance set of each process at the top level, $\mathcal{PA}(P)$. Parallel composition will be studied separately because extra explanation is needed.

$\mathcal{PA}(\mathbf{0}) = \{(1, \{\emptyset\})\} \qquad \mathcal{PA}(a.t) = \{(1, \{\{a\}\})\}$

$\mathcal{PA}(t_1 \otimes_q t_2) = \{(p, \mathcal{A}) \mid \mathcal{A} \in Accept(\mathcal{PA}(t_1)) \vee \mathcal{A} \in Accept(\mathcal{PA}(t_2)),$
$\quad p = q \cdot \pi(\mathcal{A}, \mathcal{PA}(t_1)) + (1-q) \cdot \pi(\mathcal{A}, \mathcal{PA}(t_2))\}$

$\mathcal{PA}(t_1 \oplus t_2) = \{(p, \mathcal{A}) \mid \exists \mathcal{B} \in Accept(\mathcal{PA}(t_1)), \exists \mathcal{C} \in Accept(\mathcal{PA}(t_2)) \text{ s.t. } \mathcal{A} = c(\mathcal{B} \cup \mathcal{C}),$
$\quad p = \sum_{\mathcal{A} = c(\mathcal{B} \cup \mathcal{C})} \pi(\mathcal{B}, \mathcal{PA}(t_1)) \cdot \pi(\mathcal{C}, \mathcal{PA}(t_2))\}$

$\mathcal{PA}(t_1 + t_2) = \{(p, \mathcal{A}) \mid \exists \mathcal{B} \in Accept(\mathcal{PA}(t_1)), \exists \mathcal{C} \in Accept(\mathcal{PA}(t_2)) \text{ s.t. } \mathcal{A} = \mathcal{B} \; u \; \mathcal{C},$
$\quad p = \sum_{\mathcal{A} = \mathcal{B} \; u \; \mathcal{C}} \pi(\mathcal{B}, \mathcal{PA}(t_1)) \cdot \pi(\mathcal{C}, \mathcal{PA}(t_2))\}$

$\mathcal{PA}(t\backslash C) = \{(p, \mathcal{A}) \mid \mathcal{A} = \mathcal{B}\backslash C \wedge p = \sum_{\mathcal{A} = \mathcal{B}\backslash C} \pi(t, \mathcal{B})\}$

Table 4. Probabilistic acceptance set of a semantic process

The probabilistic acceptance set defined allows us to distinguish between the different reachable states, and also provides us with the probability associated with every state.

Let us now introduce the semantic process $t/(\mathcal{A}, a)$, which models the behaviour of t, once it has executed the action a at the state given by $\mathcal{A}$. We define $t/(\mathcal{A}, a)$ as follows:

- If t is a process such that either $a \notin A(\mathcal{A})$ or $\mathcal{A} \notin \mathit{Accept}(\mathcal{PA}(t))$ then $t/(\mathcal{A}, a)$ is not defined.
- If $t = a.t_1$, the only reachable state is $\{\{a\}\}$ with probability 1. Then $t/(\{\{a\}\}, a) = t_1$.
- $t = t_1 \otimes_q t_2$. We must distinguish several subcases:
  - $\mathcal{A} \in \mathit{Accept}(\mathcal{PA}(t_1))$ and $\mathcal{A} \notin \mathit{Accept}(\mathcal{PA}(t_2))$. In this case only t_1 can execute the action a, so $t/(\mathcal{A}, a) = t_1/(\mathcal{A}, a)$.
  - $\mathcal{A} \notin \mathit{Accept}(\mathcal{PA}(t_1))$ and $\mathcal{A} \in \mathit{Accept}(\mathcal{PA}(t_2))$. It is the symmetric case: $t/(\mathcal{A}, a) = t_2/(\mathcal{A}, a)$.
  - $\mathcal{A} \in \mathit{Accept}(\mathcal{PA}(t_1))$ and $\mathcal{A} \in \mathit{Accept}(\mathcal{PA}(t_2))$. Now a can be made either by t_1 or t_2, then: $t/(\mathcal{A}, a) = t_1/(\mathcal{A}, a) \otimes_q t_2/(\mathcal{A}, a)$
- $t = t_1 \oplus t_2$. We have that, according to the definition of $\mathcal{PA}$ for the internal choice operator, each one of its states, $\mathcal{A}$, is obtained as the convex closure of a union of states, $\mathcal{A} = c(\mathcal{B} \cup \mathcal{C})$, where $\mathcal{B} \in \mathit{Accept}(\mathcal{PA}(t_1))$ and $\mathcal{C} \in \mathit{Accept}(\mathcal{PA}(t_2))$. Then, we can distinguish the following cases:
  - $a \in A(\mathcal{B})$ and $a \notin A(\mathcal{C})$. Then, it is t_1 which executes a, and thus: $t/(\mathcal{A}, a) = t_1/(\mathcal{B}, a)$.
  - $a \notin A(\mathcal{B})$ and $a \in A(\mathcal{C})$. Symmetric case: $t/(\mathcal{A}, a) = t_2/(\mathcal{C}, a)$.
  - $a \in A(\mathcal{B})$ and $a \in A(\mathcal{C})$. Either t_1 or t_2 execute the action, then: $t/(\mathcal{A}, a) = t_1/(\mathcal{B}, a) \oplus t_2/(\mathcal{C}, a)$

  Therefore, as $\mathcal{A}$ may be obtained from several different sets $\mathcal{B}$ and $\mathcal{C}$, we have:

$$
\begin{aligned}
t/(\mathcal{A}, a) = & \bigoplus_{\substack{\mathcal{B},\mathcal{C},c(\mathcal{B}\cup\mathcal{C})=\mathcal{A} \\ a\in A(\mathcal{B})\wedge a\notin A(\mathcal{C})}} t_1/(\mathcal{B}, a) \oplus \bigoplus_{\substack{\mathcal{B},\mathcal{C},c(\mathcal{B}\cup\mathcal{C})=\mathcal{A} \\ a\notin A(\mathcal{B})\wedge a\in A(\mathcal{C})}} t_2/(\mathcal{C}, a) \oplus \\
& \bigoplus_{\substack{\mathcal{B},\mathcal{C},c(\mathcal{B}\cup\mathcal{C})=\mathcal{A} \\ a\in A(\mathcal{B})\wedge a\in A(\mathcal{C})}} t_1/(\mathcal{B}, a) \oplus t_2/(\mathcal{C}, a)
\end{aligned}
$$

- $t = t_1 + t_2$. We have that every state $\mathcal{A}$ is built as $\mathcal{A} = \mathcal{B} \uplus \mathcal{C}$, where $\mathcal{B} \in \mathit{Accept}(\mathcal{PA}(t_1))$ and $\mathcal{C} \in \mathit{Accept}(\mathcal{PA}(t_2))$, and we can repeat the same case distinction as in the internal choice, thus obtaining exactly the same expression for $t/(\mathcal{A}, a)$.
- $t \backslash C$. If $a \in C$ then $(t\backslash C)/(\mathcal{A}, a)$ is not defined. If $a \notin C$ then:

$$
(t\backslash C)/(\mathcal{A}, a) = \bigotimes_{\mathcal{A}=\mathcal{B}\backslash C} \left[\frac{\pi(t, \mathcal{B})}{\pi(t\backslash C, \mathcal{A})}\right] (t/(\mathcal{B}, a))\backslash C
$$

Once we have defined the probabilistic acceptance sets of a process, not only for its first action, but all over its possible executions, we are interested in all the possible sequences of actions it can perform. Let us now introduce some notation for that. As each probabilistic node in the tree representing t can be reached by following a sequence of pairs (acceptance set, action), due to the alternating nature of these trees, we may identify these nodes by $t(s)$, where

$s = \langle \mathcal{A}_1, a_1, \ldots, \mathcal{A}_k, a_k \rangle$ is the sequence leading us to the node. We can use a similar notation for syntactical terms. Then, $(P/(\mathcal{A}, a))\,/(\mathcal{B}, b)$ will be represented shortly by $P/(\langle \mathcal{A}, a, \mathcal{B}, b \rangle)$

Parallel composition. The definition of the function over fPAT corresponding to $|$ is quite difficult, and a considerable amount of notation is needed. We will introduce intuitively the concepts related to this operator. Its formal definition is included in the extended version of this paper [2], available via world wide web.

Let us consider the processes $P_1 = a.(c \otimes_{0.4} e) \otimes_{0.3} b$, $P_2 = \overline{a}.d$ and $P = P_1 | P_2$. In figure 1 we can see the fPAT of P_1, P_2 and the first level of $P_1 | P_2$.

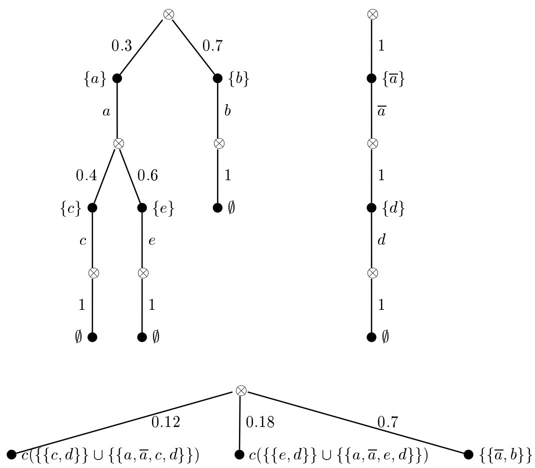

Fig. 1. fPAT of P_1, P_2 and first level of $P_1 | P_2$

In the first level, P may perform the actions that P_1 or P_2 may perform at their first level. Besides, in case of communication between P_1 and P_2, some actions of the following level in P_1 and P_2 are promoted to the first one in P. We have to consider all the possibilities, i.e., every action may or may not communicate with its homologous barred action, and this introduced some kind of nondeterminism.

In this example, actions a and $\overline{a}$ may communicate, so second level action, like c, e and d may be performed at the first level. Let us consider that a and $\overline{a}$ communicate, and the probabilistic choice after action a in P_1 is resolved following the branch labelled with probability 0.4. In this case, with probability $0.3 \times 0.4 = 0.12$, the semantic process P reaches a nondeterministic acceptance set $\mathcal{A}$ such that $A(\mathcal{A}) = \{a, \overline{a}, c, d\}$. In case of communication, only actions c and d may be performed. Nondeterminism is introduced in $\mathcal{A}$ considering the convex closure of the union of sets $\{c, d\}$ and $\{a, \overline{a}, c, d\}$.

Definition 13 *We define the sequences of P, denoted by $Seq(P)$, as the set of all the sequences of pairs (acceptance set,action) that we may obtain from* $[\![P]\!]_{fPAT}$.

Definition 14 *We say that two processes are equivalent according to the denotational semantics, which will be denoted by* $[\![P]\!]_{fPAT} = [\![Q]\!]_{fPAT}$, *iff:*

- $Seq(P) = Seq(Q)$, *and*
- *For every sequence s,* $\mathcal{PA}(P/(s)) = \mathcal{PA}(Q/(s))$.

We now state that this equivalence relation between probabilistic processes is an extension of that presented in [8] for nondeterministic processes.

Theorem 1 *Let $P, Q \in EPL_f$ be two processes.*

$$[\![P]\!]_{fAT} = [\![Q]\!]_{fAT} \text{ if and only if } [\![P]\!]_{fPAT} = [\![Q]\!]_{fPAT}$$

Also, we have that both the testing semantics and the denotational semantics identify the same processes, i.e., the denotational semantics is fully abstract with respect to the testing semantics.

Theorem 2 *Let $P, Q \in PNAL_f$ be two processes, then*

$$[\![P]\!]_{fPAT} = [\![Q]\!]_{fPAT} \text{ if and only if } P \equiv_{test} Q$$

6 Proof system

In this section we present a proof system, which is sound and complete with respect to the denotational semantics. This proof system is based on that of $\mathrm{EPL_f}$, by including the axioms related with the probabilistic choice operator. Nevertheless, some of the axioms of $\mathrm{EPL_f}$ are not preserved in the general case. For instance, the internal and the external choice now fail to be

idempotent, and distributivity of the internal over the external choice (and vice versa) is not maintained. However, these axioms can be used when we are working with a special kind of processes (probabilistically stable processes) and they are very useful when proving completeness.

Table 5, contains the set of axioms related to the following operators: **0**, prefix and the three choices. Axioms in Table 5 show that probabilistic choice is commutative (P1), associative in a probabilistic way (P2), and idempotent (P3). These axioms allow us to generalize probabilistic choice to an arbitrary number of arguments. Thus we can write $\bigotimes_i [p_i] P_i$ to denote a probabilistic choice among n processes, each one with probability $p_i > 0$. Other rules say that probabilistic choice is propagated to the following level when both processes are prefixed by the same action (P4), and that the probabilistic choice distributes over the internal and the external choice (PI1, PE1).

Let us observe that the axioms maintained from $\mathrm{EPL_f}$ are: the internal and the external choice are commutative and associative (I1,I2,E1,E2), the external choice has a zero (E3), and, finally, the internal and the external choice between processes prefixed with the same action are propagated to the following level (I3,IE1). The rest of axioms (those with the symbol $\downarrow$), do not hold in the general case, but if we restrict P, Q and R to a kind of processes, which we call *probabilistically stable processes*, they become true. These axioms are: idempotency of the internal and the external choice (I4$\downarrow$, E4$\downarrow$) , and distributivity of the internal choice over the external choice and vice versa (IE3$\downarrow$, IE4$\downarrow$).

P1	$P \otimes_p Q \equiv Q \otimes_{1-p} P$	**E1**	$P + Q \equiv Q + P$
P2	$(P \otimes_p Q) \otimes_q R \equiv P \otimes_{p \cdot q} (Q \otimes_{\frac{(1-p)\cdot q}{1-p\cdot q}} R)$	**E2**	$P + (Q + R) \equiv (P + Q) + R$
P3	$P \otimes_p P \equiv P$	**E3**	$P + \mathbf{0} \equiv P$
P4	$a.P \otimes_p a.Q \equiv a(P \otimes_p Q)$	**IE1**	$a.P + a.Q \equiv a.P \oplus a.Q$
PI1	$P \oplus (Q \otimes_p R) \equiv (P \oplus Q) \otimes_p (P \oplus R)$	**I4$\downarrow$**	$P \oplus P \equiv P$
PE1	$P + (Q \otimes_p R) \equiv (P + Q) \otimes_p (P + R)$	**E4$\downarrow$**	$P + P \equiv P$
I1	$P \oplus Q \equiv Q \oplus P$	**IE3$\downarrow$**	$P + (Q \oplus R) \equiv (P + Q) \oplus (P + R)$
I2	$P \oplus (Q \oplus R) \equiv (P \oplus Q) \oplus R$	**IE4$\downarrow$**	$P \oplus (Q + R) \equiv (P \oplus Q) + (P \oplus R)$
I3	$a.P \oplus a.Q \equiv a(P \oplus Q)$		

Table 5. Axioms for $\mathbf{0}$, prefix and choice operators

We may also apply, over probabilistically stable processes, the derived equations given in [8], which are useful to prove completeness. Finally, in Table 6 we can see the axioms for restriction and parallel composition, which, as usual, are derived operators. In rule PaE1, the function $comm(P, Q)$ (defined in [8]) is false if there is not any possibility of communication between P and Q.

Note that axioms PaI1 and PaE1, which are introduced in [8], hold only for processes in $\mathrm{EPL_f}$. Applying repeatedly P4, PI1, PE1, ReP1, and PaP1 (distribution axioms) every process in the form $P|Q$ may be transformed into $\bigotimes_{i,j}[p_i \cdot q_j]P_i|Q_j$, where $P_i, Q_j \in \mathrm{EPL_f}$.

ReE1	$(P + Q)\backslash C = P\backslash C + Q\backslash C$	**ReI1**	$(P \oplus Q)\backslash C = P\backslash C \oplus Q\backslash C$
Re1	$a.P\backslash C = \begin{cases} a.(P\backslash C) & \text{if } a \notin C \vee \overline{a} \notin C \\ \mathbf{0} & \text{otherwise} \end{cases}$	**Re01**	$\mathbf{0}\backslash C = \mathbf{0}$
ReP1	$(P \otimes_q Q)\backslash C = P\backslash C \otimes_q Q\backslash C$	**PaP1**	$P\|(Q \otimes_q R) = (P\|Q) \otimes_q (P\|R)$
Pa01	$\mathbf{0}\|P = P$	**PaI1**	$(P \oplus Q)\|R = P\|R \oplus Q\|R, \quad P, Q, R \in \mathrm{EPL_f}$
PaE1	if P, Q are $\sum\{a_i P_i, i \in I\}$, $\sum\{b_j Q_j, j \in J\}$ then $P\|Q = \begin{cases} ext(P,Q) & \text{if } comm(P,Q) \text{ is false} \\ (ext(P,Q) + int(P,Q)) \oplus int(P,Q) & \text{otherwise} \end{cases}$		where $int(P,Q) = \bigoplus\{P_i\|Q_j, a_i = \overline{b_j}\}$ $ext(P,Q) = \sum\{a_i(P_i\|Q), i \in I\} + \sum\{b_j(P\|Q_j), j \in J\}$, and $P, Q \in \mathrm{EPL_f}$

Table 6. Axioms for restriction and parallel composition

Theorem 3 *Axioms are sound with respect to the denotational semantics.*

As usual, to prove the completeness, we look for the adequate normal forms. Essentially, these normal forms represent the different ways a process has to complete its execution. This leads us to a generalized probabilistic choice at the top, followed by a generalized internal choice among a set of states, which is followed by a generalized prefixed external choice among the actions in this set, whose continuations are also in normal form.

Definition 15 *(Normal forms)*

- *Process **0** is in normal form.*
- *if $\mathcal{A}_i$ is a convex set and for every $a \in A(\mathcal{A}_i)$ there is a normal form $n(\mathcal{A}_i, a)$, then*

$$\bigotimes_i [q_i] \bigoplus_{A \in \mathcal{A}_i} \sum_{a \in A} a.n(\mathcal{A}_i, a)$$

  is a normal form.

These normal forms share some characteristics of classical models (CSP, EPL), as well as some characteristics of other probabilistic models [4]. Thus, the continuations after performing the same action in different probabilistic branches do not need to be identical (usual in probabilistic models), and the acceptance sets $\mathcal{A}_i$ must be convex as usual in classical models.

Lemma 1 *Every term in PNAL can be transformed, by using the given proof system, into another equivalent term in normal form.*

Theorem 4 *Let $P, Q \in PNAL$ be two processes, then $[\![P]\!]_{fPAT} = [\![Q]\!]_{fPAT}$ if and only if $P \equiv Q$*

Finally, by using theorems 2 and 4, we can relate the proof system with the testing semantics as follows:

Corollary 1 *Let $P, Q \in PNAL$ be two processes, then*

$$P \equiv_{test} Q \text{ if and only if } P \equiv Q$$

7 Conclusions and comparisons

In this paper we have presented an algebraic theory of probabilistic and nondeterministic processes which is a probabilistic extension of Hennessy's theory presented in [8]. An operational and a testing semantics have been introduced, as well as a denotational semantics and a proof system. The testing semantics, the denotational semantics and the proof system are shown to be fully abstract.

To our knowledge, this paper is the first that presents a general framework of probabilistic and nondeterministic processes based on Hennessy's model. Similar approaches have been presented, based on different classical process algebras. Thus, in [1] an operational semantics, a probabilistic bisimulation and a proof system are introduced for a probabilistic extension of ACP. In [14], a denotational semantics and a proof system are introduced for a probabilistic extension of CSP. In [13], only an operational and a denotational semantics are defined for another probabilistic version of CSP which does not include a hiding operator. Finally, in [21, 10] a probabilistic extension of CCS is presented, and an operational semantics, a testing semantics and a denotational semantics are defined.

In order to deal with probabilism and nondeterminism we have chosen an approach that considers that a probabilistic choice is resolved before an internal choice. Thus, the transition system we obtain is very similar to "bundle probabilistic transition systems" (BPTS) presented in [5], which are the converse of the simple model of [17, 16]. In our case (as in BPTS), a set of nondeterministic alternatives is chosen with a certain probability, while in [17, 16], an action is chosen nondeterministically and then, for each action, a distribution of probabilities over the successor states is given (reactive probabilistic choice).

A similar operational semantics is defined in [21], but in this case some of the premises of the rules include negative conditions over transitions, which is problematic (see [6]). In our model we have avoided this problem by introducing the "probabilistic stable" predicate, which is defined syntactically.

The decision of resolving probabilistic choices first leads us to a model in which some of the properties of classical process algebras are not maintained. For instance, the internal choice now fails to be idempotent, as the following example shows. Let P be the process $a.\mathbf{0} \otimes_{0.5} b.\mathbf{0}$, we then have:

$$\begin{aligned}\mathcal{PA}(P) =& \{(0.5, \{\{a\}\}), (0.5, \{\{b\}\})\}\\ \mathcal{PA}(P \oplus P) =& \{(0.25, \{\{a\}\}), (0.25, \{\{b\}\}), (0.5, \{\{a\}, \{b\}, \{a, b\}\})\}\end{aligned}$$

This is, however, a known consequence of introducing probabilities and nondeterminism by taking the approach of first resolving the probabilistic choices. We encounter the same problem in some of the aforementioned papers [1, 14, 21], and also in [20], where a probabilistic extension of CCS is presented by introducing intervals of probabilities that allow us to specify processes with uncertain information.

The normal forms are defined in a very natural way. They consist in a generalized probabilistic choice at the top, followed by a generalized internal choice between a set of states, which is followed by a generalized prefixed external choice between the actions in this set, whose continuations are also in normal form. But, as we could see in Section 6, distribution axioms allow us to define an alternative normal form that consists in a probabilistic choice followed by a non–probabilistic process in $\mathrm{EPL_f}$'s normal form, i.e., each probabilistic choice must be resolved before any internal choice. The same conclusion is obtained in [14] for a probabilistic extension of CSP.

Finally, the proof system we have obtained is very similar to Andova's proof system [1]. One noteworthy feature is that, in her case, the proof system is sound and complete with respect to probabilistic bisimulation, while in our case it is sound and complete with respect to the testing semantics.

References

1. S. Andova. Process algebra with probabilistic choice. In *Proceedings of ARTS'99, LNCS 1601*, 1999.
2. D. Cazorla, F. Cuartero, V. Valero, and F.L. Pelayo. A process algebra for probabilistic and nondeterministic processes. Technical Report DIAB-00-01-11, UCLM, 2000. http://www.info-ab.uclm.es/sec-ab/Tecrep/diab000111.ps.
3. F. Cuartero, D. de Frutos, and V. Valero. PCSP: A denotational model for probabilistic processes. In *Proceedings of Third AMAST Workshop on Real-Time Systems*, 1996.
4. F. Cuartero, D. de Frutos, and V. Valero. A sound and complete proof system for probabilistic processes. In *ARTS'96, LNCS 1231*, 1997.
5. P.R. D'Argenio, H. Hermanns, and J.-P. Katoen. On generative parallel composition. *Electronic Notes in Theoretical Computer Science*, 22:25, 1999.
6. J.F. Groote. Transition system specifications with negative premises. *Theoretical Computer Science*, 118:263–299, 1993.
7. H. Hansson and B. Jonsson. A calculus for communicating systems with time and probabilities. In *Proceedings Real-Time Systems Symposium*, Orlando, Florida, 1990.
8. M. Hennessy. *Algebraic Theory of Processes.* MIT Press, 1988.
9. C. Jones and G.D. Plotkin. A probabilistic powerdomain of evaluations. In *Proceedings of 4th Annual Symposium on Logic in Computer Science*, 1989.
10. B. Jonsson and W. Yi. Compositional testing preorders for probabilistic processes. In *10th IEEE Symposium on Logic in Computer Science*, pages 431–443, San Diego, Ca., 1995.
11. B. Jonsson and W. Yi. Fully abstract characterization of probabilistic may testing. In *5th Int. AMAST Workshop on Real-Time and Probabilistic Systems, ARTS'99*, 1999.
12. K.G. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.
13. G. Lowe. Representing nondeterministic and probabilistic behaviour in reactive processes. Technical Report PRG-TR-11-93, Oxford university computing laboratory, 1993.
14. C. Morgan, A. McIver, K. Seidel, and J.W. Sanders. Refinement-oriented probability for CSP. *Formal Aspects of Computing*, 8(6):617–647, 1996.
15. M. Núñez, D. de Frutos, and L. Llana. Acceptance trees for probabilistic processes. In *CONCUR'95, LNCS 962*, pages 249–263, 1995.
16. R. Segala. A compositional trace-based semantics for probabilistic automata. In *Proceedings of CONCUR'95, LNCS 962*, 1995.
17. R. Segala and N. Lynch. Probabilistic simulations for probabilistic processes. In *Proceedings of CONCUR'94, LNCS 836*, 1994.
18. R. van Glabbeek, S.A. Smolka, and B.U. Steffen. Reactive, generative ans stratified models of probabilistic processes. *Information and Computation*, 121(1):59–80, 1995.
19. M.Y. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proceedings of 26th FOCS*, pages 327–338, 1985.
20. W. Yi. Algebraic reasoning for real-time probabilistic processes with uncertainty information. In *Formal Techniques in Real-Time and Fault-Tolerant Systems, LNCS 863*, pages 680–693, Kiel, Germany, 1994.
21. W. Yi and K.G. Larsen. Testing probabilistic and nondeterministic processes. In *Protocol Specification, Testing and Verification XII*, pages 47–61, 1992.