A Testing Theory for Generally Distributed Stochastic Processes*

(Extended Abstract)

Natalia López and Manuel Núñez

Dpt. Sistemas Informáticos y Programación
Universidad Complutense de Madrid
{natalia,mn}@sip.ucm.es

Abstract. In this paper we present a testing theory for stochastic processes. This theory is developed to deal with processes which probability distributions are not restricted to be exponential. In order to define this testing semantics, we compute the probability with which a process passes a test before an amount of time has elapsed. Two processes will be equivalent if they return the same probabilities for any test T and any time t. The key idea consists in *joining* all the random variables associated with the computations that the composition of process and test may perform. The combination of the values that this random variable takes and the probabilities of executing the actions belonging to the computation will give us the desired probabilities. Finally, we relate our stochastic testing semantics with other notions of testing.

1 Introduction

Process algebras [Hoa85,Hen88,Mil89,BW90] have become an important theoretical formalism to analyze distributed and concurrent systems. The first proposals were not powerful enough to describe some features of real systems. Due to that fact, process algebras have been extended with information to describe quantitative and qualitative features. Therefore, several timed (e.g. [RR88,BB93, NS94]), probabilistic (e.g. [LS91,GSS95,NdFL95,NdF95,CDSY99]), and timed-probabilistic (e.g. [Han91,Low95,GLNP97]) extensions of process algebras have appeared. However, these extensions are not enough to describe faithfully some systems. There exist systems where the probability to perform an action varies as time passes. So, during the last years a new extension has appeared: *Stochastic process algebras* [GHR93,ABC$^+$94,Hil96,BG98,HS00,HHK01]. These process algebras provide information about the probability to execute actions before an amount of time elapses. These probabilities are given by probability distribution functions. Except some of them ([BBG98,DKB98,HS00,BG01]), the majority works exclusively with exponential distributions. This assumption decreases the expressiveness of the languages. However, it simplifies several of the problems

* Work partially supported by the CICYT project TIC2000-0701-C02-01.

that appear when considering general distributions. In particular, some quantities of interest, like reachability probabilities or steady-state probabilities, can be efficiently calculated by using well known methods on Markov chains. Nevertheless, the main weakness of non-exponential models, that is the analysis of properties, can be (partially) overcome by restricting the class of distributions. Phase-type distributions [Neu92] are a good candidate: They are closed under minimum, maximum, and convolution, and any other distribution over the interval $(0, \infty)$ can be approximated by arbitrarily accurate phase-type distributions. Moreover, the analysis of performance measures can be efficiently done in some general cases (see [BKLL95,EKN99] for the study of this kind of distributions in a stochastic process algebra).

In order to define the semantics of processes, the classical theory of testing [dNH84,Hen88] uses the idea of an *experimenter*. The notion of testing a process is specified by the interaction between the tested process and a set of tests. Usually, this interaction is modeled by the parallel composition of a process and a test. There have appeared testing semantics for probabilistic extensions (e.g. [Chr90,YL92,NdFL95,KN98,CDSY99]), timed extensions (e.g. [HR95,LdF97]), and probabilistic-timed extensions (e.g. [CLLS96,GLNP97]). Unfortunately, the testing theory has not been so extensively used in the field of stochastic process algebras. The definition of a testing semantics (fulfilling good properties) for this kind of languages is rather difficult, because sequences of stochastic transitions must be somehow abstracted and, in general, this is not an easy task. As far as we know, [BC00] represents the only proposal of a testing theory for stochastic process algebras. However, their study is restricted to processes which probability distribution functions are always exponential.

In this paper we will define a testing semantics for a stochastic process algebra where probability distributions are not restricted to be exponential. In our setting, processes may perform both standard actions (visible actions and internal τ actions) and stochastic actions which represent (random) delays. Our language will contain a probabilistic choice operator. In particular, this will imply that the selection among alternative stochastic transitions will be made by using a *preselection policy*. That is, according to the corresponding probabilities, one of the possible stochastic actions is chosen. Then, the process will be delayed an amount of time depending on the probability distribution associated with the chosen action.

Regarding tests, we will suppose that they cannot perform internal transitions. We impose this restriction because probabilistic testing produces very strange results if tests have the ability to perform internal transitions. For example, if we consider a CCS like probabilistic process algebras with a unique (probabilistic) choice operator[1] and we allow internal actions in tests, we will usually get that the processes $\tau\,;a\,;\texttt{STOP}$ and $a\,;\texttt{STOP}$ are not probabilistically testing equivalent. In [CDSY99] a more detailed discussion on probabilistic test-

[1] This is equivalent to consider (probabilistic) transitions systems where transitions are labeled by actions and probabilities.

ing semantics with and without internal actions in tests is presented. So, in order to keep a reasonable equivalence, we will not include internal actions in tests.

As usual, we will define the interaction between processes and tests as a parallel composition synchronizing in all the visible actions. This composition will produce a (multi)set of computations. In order to define the passing of tests, we will extract the appropriate information from these computations. This information will be a probability (indicating the probability of executing this computation) together with a random variable generated from all the random variables appearing in the sequence. So, our definition of passing tests takes into account not only the probabilities associated with computations, but also the *time* that these computations need to finish. For example, $0.3 = pass_{\leq 2}(P, T)$ indicates that the process P passes the test T with probability 0.3 before 2 units of time have passed. A similar mechanism is used in the testing semantics presented in [GLNP97]. Another alternative is given in [BC00] where the average time of the computation is used. Given the fact that we do not restrict the kind of probability distributions, in our setting, this technique would equate processes that present different stochastic behaviors.

In Fig. 1 we will use a graphical representation to introduce some stochastic processes. Greek letters like ξ, φ, and ψ (possibly decorated with an index) denote random variables, Latin letters represent visible actions, and τ represents an internal action. Transitions will be labeled by one of these actions together with a probability (we omit this value if it is equal to 1). Consider the processes P_1 and P_2. In this case, the key point consists in computing the probability to perform a before a certain amount of time has passed. If the random variable $\xi_1 + \xi_2$, that is the addition of the random variables ξ_1 and ξ_2, and the random variable ξ_3 are identically distributed, then we would like to equate P_1 and P_2. Let us note that $+$ denotes the addition of random variables, that is, the convolution of them. Consider now P_3, P_4 and P_5 in Fig. 1. These processes will be testing equivalent. Regardless of the temporal point where the (probabilistic) choice is taken, these processes follow the same *temporal* pattern (this would not be the case if a bisimulation semantics is considered). For example, P_3 will be firstly delayed according to ψ_1. Afterwards, it will be delayed either according to ψ_2 (with probability p) or according to ψ_3 (with probability $1 - p$). If we add the corresponding delays, we have that, with probability p, the action a (resp. b) will be performed after a delay determined by the addition of ψ_1 and ψ_2 (resp. the addition of ψ_1 and ψ_3). Intuitively, this is the very same situation for P_4 and P_5. Finally, P_6 and P_7 will also be testing equivalent. The reason is that both probabilistic choices produce the same result. This point motivates our presentation. In a stochastic process algebra where delays are separated from usual actions (as it is our case), stochastic *actions* must be considered somehow as *internal* actions carrying some additional information. Let us note that, in this example, if we replace φ_1 and φ_2 by *usual* actions b and c then the new processes are no longer testing equivalent.

The rest of the paper is structured as follows. In Sect. 2 we present our language and its operational semantics. In Sect. 3 we present our testing semantics

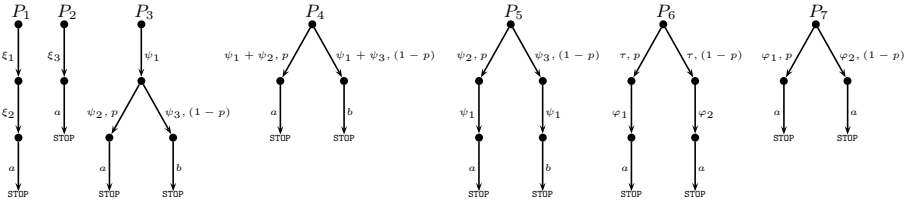

Fig. 1. Examples of stochastic processes.

by defining the set of tests, the interaction between processes and tests, and the corresponding notion of passing a test. Besides, a set of essential tests is given. In Sect. 4 we relate our semantics with other models of testing. Specifically, we will consider classical testing, pure probabilistic testing, and an adaptation of [BC00] to our framework. We will show that our testing semantics is a conservative extension of the first two. Finally, in Sect. 5 we present our conclusions and some lines for future work.

An extended version of this paper can be found in [LN01]. There, we present an alternative characterization of our testing equivalence. This characterization is based on a stochastic extension of the notion of *probabilistic acceptance sets* presented in [NdFL95].

2 Description of the Language

In this section we define our model of stochastic processes. First, we introduce some concepts on random variables. We will consider that the sample space (that is, the domain of random variables) is the set of real numbers $\mathbb{R}$ and that random variables take positive values only in $\mathbb{R}^+$, that is, given a random variable ξ we have $F_\xi(t) = 0$ for any $t < 0$. The reason for this restriction is that random variables will always be associated with time distributions.

Definition 1. Let ξ be a random variable. Its *probability distribution function*, denoted by F_ξ, is the function $F_\xi : \mathbb{R} \to [0,1]$ such as $F_\xi(x) = \mathrm{P}(\xi \leq x)$, where $\mathrm{P}(\xi \leq x)$ is the probability that ξ assumes values less than or equal to x.

We consider a *distinguished* random variable. By *unit* we denote a random variable such that $F_{unit}(x) = 1$ for any $x \geq 0$, that is, *unit* is distributed as the Dirac distribution in 0. Let us note that if we consider the addition of random variables, for any random variable ξ, we have that $\xi + unit = \xi$.

We suppose a fixed set of visible actions Act ($a, a', \ldots$ to range over Act). We assume the existence of a special action $\tau \notin \mathrm{Act}$, which represents internal behaviour. We denote by Act_τ the set $\mathrm{Act} \cup \{\tau\}$ ($\alpha, \alpha', \ldots$ to range over Act_τ). We denote by $\mathcal{V}$ the set of random variables ($\xi, \psi, \ldots$ to range over $\mathcal{V}$); $\gamma, \gamma', \ldots$ will denote generic elements in $\mathrm{Act}_\tau \cup \mathcal{V}$. Finally, $Id_\mathcal{P}$ represents the set of process variables.

Definition 2. The set of processes, denoted by $\mathcal{P}$, is given by the following BNF-expression:

$$P ::= \texttt{STOP} \mid X \mid \sum_{i=1}^{n} [p_i]\gamma_i\,;P_i \mid \mathrm{rec}X.P$$

where $X \in Id_{\mathcal{P}}$, for any $1 \leq i \leq n$ we have $\gamma_i \in \mathrm{Act}_\tau \cup \mathcal{V}$, $0 < p_i \leq 1$, and $\sum p_i = 1$.

In the definition of processes, we will usually omit trailing occurrences of STOP. Besides, we will use some syntactic sugar for the case of unary and binary choices.[2] For example, $\gamma_1\,;P_1$ stands for $\sum_{i=1}^{1}[1]\gamma_i\,;P_i$, while $\gamma_1\,;P_1 \boxplus_p \gamma_2\,;P_2$ stands for $\sum_{i=1}^{2}[p_i]\gamma_i\,;P_i$, where $p_1 = p$ and $p_2 = 1-p$.

In the previous definition, STOP denotes the process that cannot execute any action. We have included an n-ary probabilistic choice operator. Let us remark that choices are not resolved in a pure probabilistic way. For example, consider the process $a\,;\texttt{STOP} \boxplus_p b\,;\texttt{STOP}$, and suppose that the environment offers only the action a. In this case, a will be executed with probability 1, regardless the value of p. This point will be clear when we define the testing semantics. For example, the processes $a;\texttt{STOP}\boxplus_p b;\texttt{STOP}$ and $\tau;a;\texttt{STOP}\boxplus_p \tau;b;\texttt{STOP}$ are not testing equivalent. Regarding the terms appearing in a choice, $\alpha\,;P$ (with $\alpha \in \mathrm{Act}_\tau$) denotes a process that performs α and after that behaves as P. Besides, a subterm $\xi\,;P$ (with $\xi \in \mathcal{V}$) indicates that the process P is delayed by a random amount of time according to ξ. Specifically, P will start its execution with a probability p before t units of time have been consumed, where $p = P(\xi \leq t)$. Finally, $\mathrm{rec}X.P$ denotes recursion in the usual way.

We will suppose that all the random variables appearing in the definition of a process are independent. This restriction avoids *side effects*. In particular, this implies that the same random variable cannot appear twice in the definition of a process. Note that this restriction does not imply that we cannot have identically distributed random variables (as long as they have different names). Anyway, for the sake of convenience, we will use sometimes in graphical representations the same random variable in different transitions. For example, two transitions labeled by the same random variable ξ is a shorthand to indicate that these two transitions are labeled by independent random variables ψ_1 and ψ_2 that are identically distributed.

We would like to finish the presentation of our syntax by commenting on two points. First, we have chosen an n-ary probabilistic choice only because the operational semantics is easier to define. As we will comment, we would like to keep *urgency*, that is, if a process may execute a τ action then delays are forbidden. This implies that the probabilities previously associated with those stochastic transitions must be redistributed among the remaining transitions. In our current setting, we only need a simple normalization function; if we use

[2] We use $\boxplus$ to denote the binary choice operator because $+$ denotes addition of random variables.

a binary choice, we need a much more complicated function (it has six cases) that needs two additional predicates (for checking stability and deadlock of the components of the choice). Given the fact that we do not lose expressiveness, we have preferred to keep the operational semantics as simple as possible. We would also like to comment on the absence of a parallel operator. The definition of the (operational) semantics of a parallel operator is straightforward if probability distributions are exponential (because of the *memoryless* property), but this is not the case if distributions are not restricted. There are already several proposals satisfactorily dealing with a parallel operator. Among them, [BG01] presents a language being very close to ours. In their model, there is no probabilistic relation between usual actions but stochastic actions are related by *weights*. Briefly, they deal with the interleaving of stochastic actions by splitting them into two events: Start and termination. This mechanism also works in our setting. Nevertheless, in this paper we have preferred to concentrate on the definition and study of an appropriate testing semantics, which can be adapted to other (possibly more expressive) non-Markovian frameworks, rather than in the definition of a more expressive process algebra. Indeed, dealing with a parallel operator means that our ideas on stochastic testing are more difficult to transmit.

In order to define the operational semantics of processes, transitions are labeled either by an action belonging to Act_τ or by a random variable belonging to $\mathcal{V}$. These transitions have an additional label: A probability. So, a derivation $P \xrightarrow{\gamma}_p P'$ expresses that there exists a transition from P to P' labeled by the action $\gamma \in \mathrm{Act}_\tau \cup \mathcal{V}$, and this transition is performed with probability p. As in most probabilistic models, we need to take into account the different occurrences of the same probabilistic transition.

Example 1. Consider the process $P = \sum_{i=1}^{n} [\frac{1}{n}]a \,;\, P'$. If we do not take care, we have that P has only the transition $P \xrightarrow{a}_{\frac{1}{n}} P'$. So, this process would not be equivalent to $Q = a \,;\, P'$.

There are several standard methods in the literature of probabilistic processes to deal with this problem. For example, in [GSS95] every transition of a term has a unique index, in [YL92] equal transitions are joined (by adding probabilities), and in [NdFL95] multisets of transitions are considered, that is, if a transition can be derived in several ways, each derivation generates a different instance. This last approach will be taken in this paper. For instance, in the previous example we have that the transition $P \xrightarrow{a}_{\frac{1}{n}} P'$ has multiplicity equal to n.

In the definition of the operational semantics (see Fig. 2), we use the auxiliary function $\mathcal{N}_1(P)$. This function computes the *total probability* of a process P to perform actions. This is a normalization function that takes care of keeping the previously commented *urgency* property. So, $\mathcal{N}_1(P)$ returns 1 if P cannot immediately perform τ's; otherwise, $\mathcal{N}_1(P)$ is equal to the addition of the probabilities associated with the actions belonging to Act_τ. The first rule says that if $\gamma \in \mathrm{Act}_\tau$ is one of the *first* actions of a choice, this action may be performed; the probability associated with γ will be *normalized*. The second rule is used for

$$\frac{\gamma_i \in \mathrm{Act}_\tau}{\sum_{i=1}^{n} [p_i]\gamma_i ; P_i \xrightarrow{\gamma_i}_{\frac{p_i}{\mathcal{N}_1(\sum_{i=1}^{n}[p_i]\gamma_i)}} P_i} \qquad \frac{\gamma_i \in \mathcal{V} \ \wedge \ \mathcal{N}_1(\sum_{i=1}^{n}[p_i]\gamma_i)=1}{\sum_{i=1}^{n} [p_i]\gamma_i ; P_i \xrightarrow{\gamma_i}_{p_i} P_i} \qquad \frac{P[\mathrm{rec}X.P/X] \xrightarrow{\gamma}_p P'}{\mathrm{rec}X.P \xrightarrow{\gamma}_p P'}$$

$$\mathcal{N}_1(\sum_{i=1}^{n}[p_i]\gamma_i) = \begin{cases} 1 & \text{if } \tau \notin \{\gamma_i \mid 1 \le i \le n\} \\ \sum \{\!\{\, p_i \mid \gamma_i \in \mathrm{Act}_\tau \,\}\!\} & \text{otherwise} \end{cases}$$

Fig. 2. Operational Semantics.

stochastic actions. The side condition assures that no stochastic transition will be allowed if the process may immediately perform τ. Regarding this side condition, let us note that $\mathcal{N}_1$ takes the value 1 only in two cases: Either there does not exist j such that $\tau = \gamma_j$ or for any i we have $\gamma_i \in \mathrm{Act}_\tau$. In the latter case, the first condition of the second rule does not hold. The third rule is standard for CCS-like languages.

We use the following conventions: $P \xrightarrow{\gamma}$ stands for there exist $P' \in \mathcal{P}$ and $p \in (0,1]$ such that $P \xrightarrow{\gamma}_p P'$; we write $P \not\xrightarrow{\gamma}$ if there do not exist such P' and p. We write $P \Longrightarrow_p P'$ if there exist $P_1, \cdots, P_{n-1} \in \mathcal{P}$ and $p_1, \cdots, p_n \in (0,1]$ such that $P \xrightarrow{\tau}_{p_1} P_1 \xrightarrow{\tau}_{p_2} \cdots P_{n-1} \xrightarrow{\tau}_{p_n} P'$ and $p = \prod p_i$ (if $n = 0$, we have $P \Longrightarrow_1 P$). Besides, for any $\gamma \in \mathrm{Act} \cup \mathcal{V}$ and $p \in (0,1]$, $P \overset{\gamma}{\Longrightarrow}_p P'$ denotes that there exist two processes $P_1, P_2 \in \mathcal{P}$ and $p_1, p_2, p_3 \in (0,1]$ such that $P \Longrightarrow_{p_1} P_1 \xrightarrow{\gamma}_{p_2} P_2 \Longrightarrow_{p_3} P'$ and $p = p_1 \cdot p_2 \cdot p_3$. Finally, given a set $A \subseteq \mathrm{Act}_\tau \cup \mathcal{V}$ we write $P \xrightarrow{A}_p$ if we have that $p = \sum \{\!\{\, p' \mid \exists \gamma \in A, P' \in \mathcal{P} : \ P \xrightarrow{\gamma}_p P' \,\}\!\}$; otherwise, we write $P \not\xrightarrow{A}_p$.

3 Stochastic Testing Semantics

In this section we present our stochastic testing semantics. As usual, it is based on the interaction between tested processes and tests. First, we define our set of tests (we consider a set of test identifiers $Id_{\mathcal{T}}$).

Definition 3. The *set of tests*, denoted by $\mathcal{T}$, is given by the following BNF-expression:

$$T ::= \mathtt{STOP} \mid \sum_{i=1}^{n} [p_i]\alpha_i \,; T_i \mid \mathrm{rec}X.T$$

where $X \in Id_{\mathcal{T}}$, for any $1 \le i \le n$ we have $\alpha_i \in \mathrm{Act} \cup \{\omega\}$, $0 < p_i \le 1$, and $\sum p_i = 1$.

The same syntactic sugar that we gave for processes will be also used for tests. We have added a new action ω indicating successful termination of the testing procedure. As we commented in the introduction, we do not allow τ actions in tests. So, a test may perform only either visible actions, belonging to

Act, or the special action ω. We will explain the meaning of our tests by following the black box analogy described in [Mil81], where processes are considered as black boxes with buttons. The test $a\,;T$ corresponds to press the a-button and if it goes down then we continue the experiment with the test T. Regarding *non-probabilistic* tests, a test as $a\,;T \boxplus b\,;T'$ can be explained as pressing two buttons simultaneously. In our model, we do consider probabilistic tests. The test $a\,;T \boxplus_p b\,;T'$ is explained as pressing two buttons at the same time but with *different* strengths.

The operational behaviour of tests is the same as that for processes (considering ω as a *usual* action). Let us remark that the function $\mathcal{N}_1$ (used in Fig. 2 as normalization factor) will always take the value 1. The interaction between a process and a test is modeled by the parallel composition of the tested process P and the test T, denoted by $P \parallel T$. The rules describing how processes and tests interact are given in Fig. 3. We have to make a trade-off between the classical testing framework and the probabilistic framework. In the former, if a test may perform the ω action then the testing procedure may finish. In particular, a test as $a\,;T_1 \boxplus_p \omega\,;T_2$ would behave exactly as the test $\omega\,;\texttt{STOP}$. If we consider the probabilistic framework given in [NdFL95], the testing procedure finishes (with probability 1) only if the tested process is stable. So, we will consider that synchronizations in visible actions can be performed only if the test cannot perform an ω action (this is expressed in the first rule in Fig. 3). If the process may perform either τ or stochastic actions then they are performed (second and third rules, respectively). Finally, if the test can perform ω then the interaction of process and test does so. In order to avoid useless computations, we *cut* the testing procedure by evolving into STOP. This transition is performed with a probability equal to 1 minus the measure of *instability* of the tested process. The side condition assures that a 0 probability transition is not generated. As usually, we have a normalization function. The function $\mathcal{N}_2(P \parallel T)$ computes the sum of the probabilities associated with transitions whose labels belong to $\mathrm{Act}_\tau \cup \mathcal{V}$ such that $P \parallel T$ may perform them.

In the following definition we introduce the notion of *successful* computation. We will also define some concepts on successful computations which will be used when defining the notion of passing a test.

Definition 4. Let P be a process and T be a test. A *computation* C is a sequence of transitions $C = P \parallel T \xrightarrow{\gamma_1} {}_{p_1}P_1 \parallel T_1 \xrightarrow{\gamma_2} {}_{p_2}P_2 \parallel T_2 \xrightarrow{\gamma_3} {}_{p_3} \cdots \xrightarrow{\gamma_n} {}_{p_n}P_n \parallel T_n \cdots$. We say that $P \parallel T$ is the *initial state* of C or C is a computation from $P \parallel T$.

A computation C is *successful* if $P_n \parallel T_n \xrightarrow{\omega} {}_p\texttt{STOP}$ for some $n \geq 0$ and $p > 0$. In this case, we say that $\mathit{length}(C) = n$. We denote by $\mathit{Success}(P \parallel T)$ the multiset of successful computations from $P \parallel T$.

Let $C \in \mathit{Success}(P \parallel T)$. We define the *random variable associated with* C, denoted by $\mathit{random}(C)$, as:

$$\mathit{random}(C) = \begin{cases} \mathit{unit} & \text{if } C = P \parallel T \xrightarrow{\omega} {}_p\texttt{STOP} \\ \mathit{random}(C') & \text{if } C = P \parallel T \xrightarrow{\gamma} {}_pC' \ \wedge\ \gamma \in \mathrm{Act}_\tau \\ \gamma + \mathit{random}(C') & \text{if } C = P \parallel T \xrightarrow{\gamma} {}_pC' \ \wedge\ \gamma \in \mathcal{V} \end{cases}$$

$$\frac{P \xrightarrow{a}_p P',\ T \xrightarrow{a}_q T',\ T \not\xrightarrow{\omega}}{P \| T \xrightarrow[\mathcal{N}_2(P\|T)}^{a}{}_{p\cdot q} P' \| T'} \qquad \frac{P \xrightarrow{\tau}_p P'}{P \| T \xrightarrow[\mathcal{N}_2(P\|T)}^{\tau}{}_{p} P' \| T}$$

$$\frac{P \xrightarrow{\xi}_p P'}{P \| T \xrightarrow[\mathcal{N}_2(P\|T)}^{\xi}{}_{p} P' \| T} \qquad \frac{T \xrightarrow{\omega},\ P \not\xrightarrow{\mathcal{V}\cup\{\tau\}}_1}{P \| T \xrightarrow{\omega}_{1-instab(P)} \texttt{STOP}}$$

$$instab(P) = \sum \{\!|\ p \mid \exists \gamma \in \mathcal{V} \cup \{\tau\}, P' \in \mathcal{P} :\ P \xrightarrow{\gamma}_p P' \ |\!\}$$

$$\mathcal{N}_2(P \parallel T) = \begin{cases} 1 & \text{if } T \xrightarrow{\omega} \\ \sum \{\!|\ p \cdot q \mid \exists a, P', T' : P \xrightarrow{a}_p P' \wedge T \xrightarrow{a}_q T' \ |\!\} + instab(P) & \text{if } T \not\xrightarrow{\omega} \end{cases}$$

Fig. 3. Interaction between processes and tests.

Let $C \in Success(P \parallel T)$. We define the *probability* of C, denoted by $Prob(C)$, as

$$Prob(C) = \begin{cases} p & \text{if } C = P \parallel T \xrightarrow{\omega}_p \\ p \cdot Prob(C') & \text{if } C = P \parallel T \xrightarrow{\gamma}_p C' \end{cases}$$

First, let us note that we will have a multiset of computations. The random variable $random(C)$ is used to compute the time that C needs to be executed, that is, for any time t we have that $\mathrm{P}(random(C) \leq t)$ gives the probability of executing C before a time t has passed. Finally, $Prob(C)$ cumulates the probabilities of all the decisions taken to obtain that particular computation.

Example 2. Let us consider the computations depicted in Fig. 4, where the symbol $\circledcirc$ represents the successful termination of the testing procedure. Consider $P_1 = (\xi_1 \,;\, a) \boxplus_{\frac{1}{3}} (a \,;\, b)$ and $T_1 = (a \,;\, \omega) \boxplus_{\frac{1}{4}} (a \,;\, b)$. The first graph in Fig. 4 describes the multiset of computations from $P_1 \parallel T_1$.

Consider the recursive process $P_2 = \mathrm{rec}X.(\xi_2 ; P_2 \boxplus_{\frac{1}{2}} a)$ and the tests $T_2 = a;\omega$ and $T_3 = \omega$. In this case, the multisets of computations from $P_2 \parallel T_2$ and $P_2 \parallel T_3$ are both infinite (See the second and third graphs in Fig. 4).

In our model, the notion of passing tests has an additional value as parameter: The time that the process needs to pass the test. A process P passes a test T before a certain amount of time t with a probability p if p is equal to the addition of all the probabilities associated with successful computations from $P \parallel T$ that take a time less than or equal to t to be finished.

Definition 5. Let P be a process, T be a test, $p \in (0, 1]$, and $t \in \mathbb{R}^+$. We say that P *passes* T *before time* t *with probability* p, denoted by $pass_{\leq t}(P, T) = p$, if

$$\sum_{C \,\in\, Success(P \,\parallel\, T)} \mathrm{P}(random(C) \leq t) \cdot Prob(C) = p$$

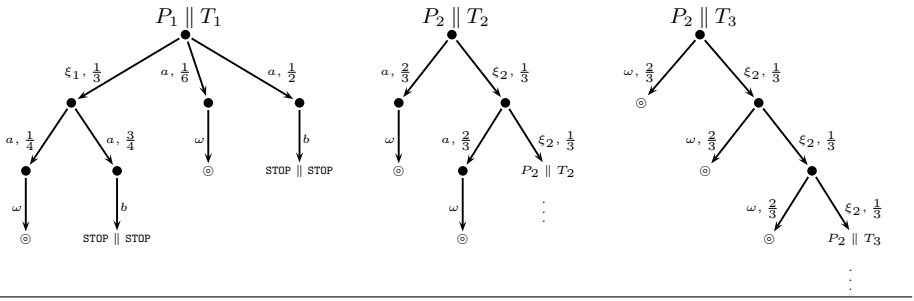

Fig. 4. Examples of computations.

The following result gives an alternative definition of the previous notion. The proof follows straightforward from the fact that successful computations have finite length.

Lemma 1. Let P be a process, T be a test, and $t \in \mathbb{R}^+$. We have that

$$pass_{\leq t}(P,T) = \lim_{n \to \infty} \sum_{\substack{C \in Success(P \parallel T) \\ length(C) < n}} \mathrm{P}(random(C) \leq t) \cdot Prob(C)$$

Definition 6. *(Testing equivalence)* Let P, Q be processes, and $\mathcal{T}' \subseteq \mathcal{T}$ a family of tests. We say that P is *stochastically testing equivalent* to Q with respect to $\mathcal{T}'$, denoted by $P \sim_{\mathcal{T}'} Q$, if for any $T \in \mathcal{T}'$ and any $t \in \mathbb{R}^+$ we have $pass_{\leq t}(P,T) = pass_{\leq t}(Q,T)$. If we consider the whole family of tests $\mathcal{T}$, we write $P \sim_{stoc} Q$ instead of $P \sim_{\mathcal{T}} Q$, and we say that P and Q are stochastically testing equivalent.

In the following example we present some processes that are not stochastically testing equivalent and some tests are given to distinguish them.

Example 3. Let us consider the following processes: $Q_1 = \tau;a \boxplus_p \tau;b$, $Q_2 = a \boxplus_p b$, and $Q_3 = \tau\,;\, a \boxplus_p b$. As it is the case for probabilistic models, they are not equivalent in our semantics. Considering $T = a\,;\omega$, we have that for any $t \in \mathbb{R}^+$, $pass_{\leq t}(Q_1,T) = p$, meanwhile $pass_{\leq t}(Q_2,T) = pass_{\leq t}(Q_3,T) = 1$. Moreover, the test $T' = b;\omega$ shows that Q_2 and Q_3 are not stochastically testing equivalent: For any $t \in \mathbb{R}^+$ we have $pass_{\leq t}(Q_2,T') = 1$ but $pass_{\leq t}(Q_3,T') = 1 - p$.

Consider the processes $R_1 = \xi;a$ and $R_2 = \psi;a$. If ξ and ψ are not identically distributed, then there exists a time $t_1 \in \mathbb{R}^+$ such that $\mathrm{P}(\xi \leq t_1) \neq \mathrm{P}(\psi \leq t_1)$. So, $pass_{\leq t_1}(P,\omega) \neq pass_{\leq t_1}(Q,\omega)$.

Consider $R_3 = \xi\,;\, a$ and $R_4 = a\,;\,\xi$, and suppose that ξ is not distributed as *unit*. This implies that there exists $t_1 \in \mathbb{R}^+$ such that $\mathrm{P}(\xi \leq t_1) = p < 1$. Then these two processes can be distinguished by the test ω, because we have that $pass_{\leq t_1}(R_3,\omega) = p$ and $pass_{\leq t_1}(R_4,\omega) = 1$.

Consider the processes and tests of Example 2. Once we have computed the corresponding set of computations the probability of passing the tests can be

computed. We only need to add the probabilities associated with successful computations. So, we have that for any $t \in \mathbb{R}^+$, $pass_{\leq t}(P_1, T_1) = \frac{1}{12} \cdot \mathrm{P}(\xi_1 \leq t) + \frac{1}{6}$ and $pass_{\leq t}(P_2, T_2) = pass_{\leq t}(P_2, T_3) = \sum_{i=0}^{\infty} (\frac{1}{3})^i \cdot \frac{2}{3} \cdot \mathrm{P}(i \cdot \xi_2 \leq t)$, where $n \cdot \xi$ stands for the addition of ξ with itself n times.

We will finish this section by showing that the set of tests can be reduced. Specifically, we will give a family of test which has the same discriminatory power as the whole family of tests $\mathcal{T}$. First, we can restrict ourselves to finite tests (i.e. non-recursive tests). The proof is made by using an appropriate extension of the technique given in [GN99], where a similar result is given for a probabilistic process algebra.

Lemma 2. Let $\mathcal{T}_f \subseteq \mathcal{T}$ be the set of tests without occurrences of the recursion operator, and P, Q be processes. Then $P \sim_{\mathcal{T}_f} Q$ iff $P \sim_{stoc} Q$.

We will show that the set of tests can be restricted even more. In the following definition, a set of *essential* tests is given.

Definition 7. The set of *essential tests*, denoted by $\mathcal{T}_e$, is given by the following BNF-expression:

$$T ::= \sum_{i=1}^{n} [p_i](a_i \,;\, T_i) \mid \omega \qquad \text{where } T_i = \begin{cases} T & \text{if } i = n \\ \mathtt{STOP} & \text{otherwise} \end{cases}$$

where $\{a_1, \ldots, a_n\} \subseteq \mathrm{Act}$, for any $1 \leq i \leq n$ we have $0 < p_i \leq 1$, and $\sum p_i = 1$.

An essential test is either the test ω or a generalized probabilistic choice among a set of visible actions. In the latter case, all the *continuations* except one are equal to STOP. Let us remark that similar sets of essential tests appear in [NdFL95,CDSY99]. The proof of this result follows the same pattern as the given in [Núñ96].

Theorem 1. Let P, Q be processes. Then $P \sim_{\mathcal{T}_e} Q$ iff $P \sim_{stoc} Q$.

4 Relation with Other Notions of Testing

In this section we compare our stochastic testing semantics with other testing models. Specifically, we will consider the *may* and *must* notions of testing, a probabilistic testing semantics similar to that of [CDSY99], and the (Markovian) testing semantics of [BC00]. First, we will define a subset of the whole set of processes $\mathcal{P}$.

Definition 8. The set of *probabilistic processes*, denoted by $\mathcal{P}_P$, is given by the BNF-expression

$$P ::= \mathtt{STOP} \mid X \mid \sum_{i=1}^{n} [p_i]\alpha_i \,;\, P_i \mid \mathrm{rec}X.P$$

where $X \in Id_{\mathcal{P}}$, for any $1 \leq i \leq n$ we have $\alpha_i \in \mathrm{Act}_\tau$, $0 < p_i \leq 1$, and $\sum p_i = 1$.

Next, we will study the notions of *may* and *must* testing [dNH84,Hen88]. We can define equivalent notions for our language $\mathcal{P}_P$.

Definition 9. Let $P \in \mathcal{P}_P$ and $T \in \mathcal{T}$. We write P *may* T if $Success(P \,\|\, T) \neq \emptyset$ and P *must* T if for any *maximal* (i.e. that it cannot be extended) computation C we have that $C \in Success(P \,\|\, T)$. Let $P, Q \in \mathcal{P}_P$. We write $P \sim_{may} Q$ if for any $T \in \mathcal{T}$ we have P *may* T iff Q *may* T. We write $P \sim_{must} Q$ if for any $T \in \mathcal{T}$ we have P *must* T iff Q *must* T.

Note that in the previous definitions we do not use the probabilistic information contained in either the processes or the tests. We can recover these notions of testing in our framework as the following result states (the proof is straightforward).

Lemma 3. Let $P \in \mathcal{P}_P$ and $T \in \mathcal{T}$. We have P *may* T iff $\exists t \in \mathbb{R}^+$ such that $pass_{\leq t}(P, T) > 0$. If P is divergence free, then P *must* T iff $\exists t \in \mathbb{R}^+$ such that $pass_{\leq t}(P, T) = 1$.

Note that, in the previous lemma, the values of t are irrelevant. The following result (whose proof is trivial) states that our testing semantics is a strict refinement of the classical notions for divergence free processes.

Corollary 1. Let $P, Q \in \mathcal{P}_P$. We have that $P \sim_{stoc} Q$ implies $P \sim_{may} Q$. Moreover, if P and Q are divergence free, we also have that $P \sim_{stoc} Q$ implies $P \sim_{must} Q$.

Let us remark that the previous result does not hold for must testing if we consider divergent processes. For example, the (non-probabilistic) processes $\mathrm{rec}X.(a \boxplus \tau \,;\, X)$ and $a \,;\, \mathtt{STOP}$ are not must testing equivalent. However, for any $p \in (0, 1)$ the probabilistic processes $\mathrm{rec}X.(a \boxplus_p \tau \,;\, X)$ and $a \,;\, \mathtt{STOP}$ are stochastically testing equivalent. So, for a process presenting divergent behavior, passing a test with probability 1 is not equivalent to pass it in the must semantics. A more extended discussion on this can be found in [NR99].

The probabilistic testing theory defined in [CDSY99] computes the probability of passing a test as the sum of the probabilities associated with all the successful computations. First, they study a testing semantics where tests are τ-free. Then they study the general case. We will define a (pure) probabilistic testing semantics following the lines of [CDSY99].

Definition 10. Let $P \in \mathcal{P}_P$, $T \in \mathcal{T}$, and $p \in (0, 1]$. We write P *pass*$_p$ T iff $\sum_{C \in Success(P \| T)} Prob(C) = p$.

Two processes $P, Q \in \mathcal{P}_P$ are *probabilistically testing equivalent*, denoted by $P \sim_P Q$, if for any test T we have P *pass*$_p$ T iff Q *pass*$_p$ T.

This notion of testing can be easily included in our framework as the following result states. The proof is trivial just taking into account that if $P \in \mathcal{P}_P$ and T is a test, then for any successful computation C from $P \parallel T$ we have $random(C) = unit$.

Lemma 4. Let $P \in \mathcal{P}_P$, $T \in \mathcal{T}$, and $p \in (0,1]$. We have P $pass_p$ T iff $\forall t \in \mathbb{R}^+$ $pass_{\leq t}(P,T) = p$. Moreover, for any $P, Q \in \mathcal{P}_P$ we have $P \sim_P Q$ iff $P \sim_{stoc} Q$.

In [BC00] a Markovian testing theory is presented. Given the fact that our language is very different from theirs, we cannot automatically compare both testing semantics. In the following, we will adapt their notion of testing to our framework. The testing theory presented in [BC00] does not compute additions of random variables. Instead, they consider the *average time* that computations need to be performed. In our case, we can consider the *expected value* of the random variable associated with the execution of a computation. Due to our assumption of independence of random variables, this expected value is equal to the addition of the expected values of the random variables performed along the computation.

Definition 11. Let $P \in \mathcal{P}$ and $T \in \mathcal{T}$. For any $t \in \mathbb{R}^+$ the probability with which P *passes* T in *average time* before time t has passed, denoted by $pass_{rate \leq t}(P,T)$, is defined as

$$pass_{rate \leq t}(P,T) = \sum_{C \in Success(P \parallel T)} Prob(C) \cdot \mathrm{P}(\mathrm{E}[random(\mathrm{C})] \leq \mathrm{t})$$

Two processes $P, Q \in \mathcal{P}$ are *testing equivalent in average time*, denoted by $P \sim_{av} Q$, if for any test T and any $t \in \mathbb{R}^+$, $pass_{rate \leq t}(P,T) = pass_{rate \leq t}(Q,T)$.

The following result trivially follows from the corresponding notions of testing. It is also trivial to see that the reverse implication does not hold. Consider two processes $P_1 = \xi_1 \,;\, \mathtt{STOP}$ and $P_2 = \xi_2 \,;\, \mathtt{STOP}$ such that $\mathrm{E}[\xi_1] = \mathrm{E}[\xi_2]$, but ξ_1 and ξ_2 not identically distributed. We have $P_1 \sim_{av} P_2$ while $P_1 \not\sim_{stoc} P_2$.

Lemma 5. Let $P, Q \in \mathcal{P}_P$. We have $P \sim_{stoc} Q$ implies $P \sim_{av} Q$.

5 Conclusions and Future Work

In this paper we have studied a testing semantics for a class of stochastic processes with general distributions. We have given a set of essential tests. We have also compare our framework with other notions of testing. Regarding future work, we are interested in the study of an axiomatization of our testing semantics. We would also like to present our semantic framework for a more expressive language (containing a parallel operator). We have already used the model defined in [BG01] but the presentation of our testing framework is rather involved.

Acknowledgments. We would like to thank the anonymous referees of this paper for the careful reading and the useful comments.

References

[ABC+94] M. Ajmone Marsan, A. Bianco, L. Ciminiera, R. Sisto, and A. Valenzano. A LOTOS extension for the performance analysis of distributed systems. *IEEE/ACM Transactions on Networking*, 2(2):151–165, 1994.

[BB93] J.C.M. Baeten and J.A. Bergstra. Real time process algebra. *Formal Aspects of Computing*, 3:142–188, 1993.

[BBG98] M. Bravetti, M. Bernardo, and R. Gorrieri. Towards performance evaluation with general distributions in process algebras. In *CONCUR'98, LNCS 1466*, pages 405–422. Springer, 1998.

[BC00] M. Bernardo and W.R. Cleaveland. A theory of testing for markovian processes. In *CONCUR'2000, LNCS 1877*, pages 305–319. Springer, 2000.

[BG98] M. Bernardo and R. Gorrieri. A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science*, 202:1–54, 1998.

[BG01] M. Bravetti and R. Gorrieri. The theory of interactive generalized semi-markov processes. To appear in Theoretical Computer Science, 2001.

[BKLL95] E. Brinksma, J.-P. Katoen, R. Langerak, and D. Latella. A stochastic causality-based process algebra. *The Computer Journal*, 38(7):553–565, 1995.

[BW90] J.C.M. Baeten and W.P. Weijland. *Process Algebra*. Cambridge Tracts in Computer Science 18. Cambridge University Press, 1990.

[CDSY99] R. Cleaveland, Z. Dayar, S.A. Smolka, and S. Yuen. Testing preorders for probabilistic processes. *Information and Computation*, 154(2):93–148, 1999.

[Chr90] I. Christoff. Testing equivalences and fully abstract models for probabilistic processes. In *CONCUR'90, LNCS 458*, pages 126–140. Springer, 1990.

[CLLS96] R. Cleaveland, I. Lee, P. Lewis, and S.A. Smolka. A theory of testing for soft real-time processes. In *8th International Conference on Software Engineering and Knowledge Engineering*, 1996.

[DKB98] P.R. D'Argenio, J.-P. Katoen, and E. Brinksma. An algebraic approach to the specification of stochastic systems. In *Programming Concepts and Methods*, pages 126–147. Chapman & Hall, 1998.

[dNH84] R. de Nicola and M.C.B. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.

[EKN99] A. El-Rayes, M. Kwiatkowska, and G. Norman. Solving infinite stochastic process algebra models through matrix-geometric methods. In *7th International Workshop on Process Algebra and Performance Modelling*, pages 41–62, 1999.

[GHR93] N. Götz, U. Herzog, and M. Rettelbach. Multiprocessor and distributed system design: The integration of functional specification and performance analysis using stochastic process algebras. In *16th Int. Symp. on Computer Performance Modelling, Measurement and Evaluation (PERFORMANCE'93), LNCS 729*, pages 121–146. Springer, 1993.

[GLNP97] C. Gregorio, L. Llana, M. Núñez, and P. Palao. Testing semantics for a probabilistic-timed process algebra. In *4th International AMAST Workshop on Real-Time Systems, Concurrent, and Distributed Software, LNCS 1231*, pages 353–367. Springer, 1997.

[GN99] C. Gregorio and M. Núñez. Denotational semantics for probabilistic refusal testing. In *PROBMIV'98, Electronic Notes in Theoretical Computer Science 22*. Elsevier, 1999.

[GSS95] 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.

[Han91] H. Hansson. *Time and Probability in Formal Design of Distributed Systems*. PhD thesis, Department of Computer Systems. Uppsala University, 1991.

[Hen88] M. Hennessy. *Algebraic Theory of Processes*. MIT Press, 1988.

[HHK01] H. Hermanns, U. Herzog, and J.-P. Katoen. Process algebra for performance evaluation. To appear in Theoretical Computer Science, 2001.

[Hil96] J. Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, 1996.

[Hoa85] C.A.R. Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985.

[HR95] M. Hennessy and T. Regan. A process algebra for timed systems. *Information and Computation*, 117(2):221–239, 1995.

[HS00] P.G. Harrison and B. Strulo. SPADES – a process algebra for discrete event simulation. *Journal of Logic Computation*, 10(1):3–42, 2000.

[KN98] 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.

[LdF97] L. Llana and D. de Frutos. Denotational semantics for timed testing. In *4th AMAST Workshop on Real-Time Systems, Concurrent and Distributed Software, LNCS 1231*, pages 368–382, 1997.

[LN01] N. López and M. Núñez. A testing theory for generally distributed stochastic processes. Available at: `http://dalila.sip.ucm.es/~natalia/papers/stoctesting.ps.gz`, 2001.

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

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

[Mil81] R. Milner. A modal characterization of observable machine-behaviour. In *6th CAAP, LNCS 112*, pages 25–34. Springer, 1981.

[Mil89] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[NdF95] M. Núñez and D. de Frutos. Testing semantics for probabilistic LOTOS. In *Formal Description Techniques VIII*, pages 365–380. Chapman & Hall, 1995.

[NdFL95] M. Núñez, D. de Frutos, and L. Llana. Acceptance trees for probabilistic processes. In *CONCUR'95, LNCS 962*, pages 249–263. Springer, 1995.

[Neu92] M. Neuts. Two further closure properties of Ph-distributions. *Asia-Pacific Journal of Operational Research*, 9(1):77–85, 1992.

[NR99] 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.

[NS94] X. Nicollin and J. Sifakis. The algebra of timed process, ATP: Theory and application. *Information and Computation*, 114(1):131–178, 1994.

[Núñ96] M. Núñez. *Semánticas de Pruebas para Álgebras de Procesos Probabilísticos*. PhD thesis, Universidad Complutense de Madrid, 1996.

[RR88] G.M. Reed and A.W. Roscoe. A timed model for communicating sequential processes. *Theoretical Computer Science*, 58:249–261, 1988.

[YL92] W. Yi and K.G. Larsen. Testing probabilistic and nondeterministic processes. In *Protocol Specification, Testing and Verification XII*, pages 47–61. North Holland, 1992.