Testing Probabilistic Automata

Roberto Segala

Dipartimento di Scienze dell'Informazione
Università di Bologna - Italy

Abstract. We study testing preorders for probabilistic automata and we characterize them in terms of relations that are based on inclusion of trace and failure distributions, i.e., probability distributions over failures and traces that can arise in a probabilistic computation. The novelty of our approach to testing is the use of multiple success actions rather than a single action. This allows us to observe the relative probabilities of different traces within a computation.
We define and characterize two kinds of testing preorders: preorders sensitive to infinite traces, and preorders sensitive to finite traces only. The second kind of preorder is an extension to the probabilistic framework of the testing preorders of De Nicola and Hennessy. We show that under assumptions of finite branching and strong convergence the two kinds of preorders coincide.

1 Introduction

The probabilistic automaton model [13] was introduced as a model for randomized concurrent computation that can be used for the analysis of randomized distributed algorithms [9, 11, 1] and that at the same time has strong mathematical foundations. To achieve the second goal, probabilistic automata are an extension of ordinary labeled transition systems [10], where a transition (s, a, s') is replaced by a triplet $(s, a, \mathcal{P})$ with $\mathcal{P}$ a discrete probability distribution over states.

In [14] several simulation and bisimulation relations, both in their strong and weak form, are defined for probabilistic automata, and in [12] a trace based semantics for probabilistic automata is studied. The results in [14, 12] show that indeed several standard concepts for ordinary labeled transition systems extend to the probabilistic case, thus confirming the original objectives of the probabilistic model.

In this paper we extend the theory of testing [3, 5] to the probabilistic model. Our main objective is to show that it is possible to define a theory of testing for probabilistic automata, and to give alternative characterizations of the may and must relations in terms of some form of probabilistic traces and failures. In particular, the trace distributions of [12] are the alternative characterization of the probabilistic may preorder. We introduce two versions of the testing preorders: one version deals with finite and infinite traces; the other version deals with finite traces only, so that the idea behind the testing preorders of [3] is captured better.

Our main intuition is that we need more success actions rather than a single action w. In fact, since in a probabilistic framework we are interested in observing the relative probabilities of different actions within the same probabilistic computation, we need at least one success action for each action to be observed. We assume that a test has at most countably many distinct success actions. At the end of the paper we show sufficient conditions for the success actions of a test to be finite.

Given a test T for a probabilistic automaton M, i.e., a probabilistic automaton with at most countably many success actions, we denote by H a probabilistic execution of $M \| T$, the parallel composition of M and T. A probabilistic execution is essentially a cycle-free Markov chain obtained by choosing a transition, possibly probabilistically, from each state. We define $W(H)$ to be a vector of probabilities $(p_1, p_2, \ldots)$, where for each i, p_i is the probability of occurrence of action w_i in H. We assume implicitly to have an enumeration of the success actions.

The may preorder is defined as follows: $M_1 \sqsubseteq_{MAY} M_2$ iff for every test T and every probabilistic execution H_1 of $M_1 \| T$, there is a probabilistic execution H_2 of $M_2 \| T$ such that $W(H_1) \leq W(H_2)$, where the less than or equal to relation is defined componentwise. Our definition is similar to the definition of may testing proposed in [7]. In [7] there is only one success action, and two systems are related by comparing, for each test, the supremums of the probabilities of success. In our case we do not compare supremums because the closure of the set of vectors W does not have a unique maximum element in general. Another consequence of not considering supremums is that our may preorder is sensitive to the presence or absence of limit vectors, while the may preorder of [7] is not.

The must preorder is symmetric to the may preorder. Namely, $M_1 \sqsubseteq_{MUST} M_2$ iff for every test T and every probabilistic execution H_2 of $M_2 \| T$, there is a probabilistic execution H_1 of $M_1 \| T$ such that $W(H_1) \leq W(H_2)$. Also in this case there is a similarity with the definition in [7]. Our definition states that whenever M_2 passes T with a probability vector W, then it is the case that, resolving the nondeterminism appropriately, M_1 passes T with a probability vector not better than W. Therefore, if M_1 performs always better than some vector W, then M_2 performs always better than W as well.

We show that the may preorder can be characterized alternatively in terms of the trace distribution precongruence of [12], and that the reversed must preorder can be characterized alternatively in terms of the *failure distribution precongruence*, which is a probabilistic generalization of the failure preorder of [2, 6]. The failure distribution precongruence is first defined in this paper and is the coarsest precongruence contained in the failure distribution inclusion preorder. A failure distribution is a probabilistic generalization of the failures of [2, 6], consisting of a probability distribution over failures induced by a probabilistic execution. Of course, as in [2], our characterization of the must preorder holds only for *strongly convergent* systems, i.e., systems that cannot perform infinite internal computation from any point. To deal with the more general case we would need a probabilistic extension of the *divergences* of [2], which we defer to further work.

The characterizations above of our may and must testing preorders are in terms of mathematical objects that include infinite traces. However, especially for decidability issues, it is typical to consider finite traces only. It would be desirable as well to consider only finite vectors of success actions rather than infinite vectors. Therefore, the second part of the paper concentrates on reducing the power of the tests so that the infinite behavior of a probabilistic automaton is not visible.

We define a new kind of trace and failure distributions where all the probability is concentrated on finite traces and failures only. In particular, for each trace and failure distribution there is a limit to the length of the trace and failures that may occur with a non-zero probability (*finite* trace and failure distributions). The related precongruence relations can be characterized again in terms of testing, where for each test there is a maximum number of external actions that can be performed sequentially. These last preorders are based on objects that we can regard as finite, and in our opinion are the relations that express better the testing preorders of [3]. Indeed, if we restrict ourselves to the non probabilistic case, our last testing preorders become the preorders of [3].

The finite preorders enjoy other properties that are valid for ordinary nondeterministic systems. Namely, it is known that the finite behaviors of a system are sufficient to describe the infinite behaviors of the system whenever the system is *finite branching* (finitely many transitions enabled from each state) and strongly convergent. We show that the same property holds in the probabilistic framework. Finally, we impose additional restrictions that allow us to deal only with tests that have finitely many success actions.

The rest of the paper is organized as follows. Section 2 gives an overview of the probabilistic automata of [13], the basic model used in this paper; Section 3 defines the failure distribution precongruence; Section 4 defines our may and must preorders and provides alternative characterizations in terms of trace and failure distributions; Section 5 defines the finite preorders and provides alternative characterizations in terms of finite trace and failure distributions; Section 6 shows sufficient conditions for finite trace and failure distributions to characterize ordinary trace and failure distributions, and for tests to have finitely many success actions.

2 Preliminaries

In this section we give an overview of the probabilistic automata of [13], which appear also in [12].

2.1 Probability Spaces

A *probability space* is a triple $(\Omega, \mathcal{F}, P)$ where Ω is a set, $\mathcal{F}$ is a collection of subsets of Ω that is closed under complement and countable union and such that $\Omega \in \mathcal{F}$, and P is a function from $\mathcal{F}$ to $[0,1]$ such that $P[\Omega] = 1$ and such that for any collection $\{C_i\}_i$ of at most countably many pairwise disjoint elements of $\mathcal{F}$, $P[\cup_i C_i] = \sum_i P[C_i]$. A probability space $(\Omega, \mathcal{F}, P)$ is *discrete* if $\mathcal{F} = 2^\Omega$ and for each $C \subseteq \Omega$, $P[C] = \sum_{x \in C} P[\{x\}]$. Given a set X, denote by $Probs(X)$ the set of discrete probability distributions whose sample space is a subset of X and such that the probability of each element is not 0.

The Dirac distribution over an element x, denoted by $\mathcal{D}(x)$, is the probability space with a unique element x.

Throughout the paper we denote a probability space $(\Omega, \mathcal{F}, P)$ by $\mathcal{P}$. As a notational convention, if $\mathcal{P}$ is decorated with indices and primes, then the same indices and primes carry to its elements. Thus, $\mathcal{P}'_i$ denotes $(\Omega'_i, \mathcal{F}'_i, P'_i)$.

The *product* $\mathcal{P}_1 \otimes \mathcal{P}_2$ of two discrete probability spaces $\mathcal{P}_1$, $\mathcal{P}_2$ is the discrete probability space $(\Omega_1 \times \Omega_2, 2^{\Omega_1 \times \Omega_2}, P)$, where $P[(x_1, x_2)] = P_1[x_1]P_2[x_2]$ for each $(x_1, x_2) \in \Omega_1 \times \Omega_2$.

A function $f : \Omega \to \Omega'$ is said to be a *measurable* function from $(\Omega, \mathcal{F})$ to $(\Omega', \mathcal{F}')$ if for each set C of $\mathcal{F}'$ the inverse image of C, denoted by $f^{-1}(C)$, is an element of $\mathcal{F}$. Let P be a probability measure on $(\Omega, \mathcal{F})$, and let P' be defined on $\mathcal{F}'$ as follows: for each element C of $\mathcal{F}'$, $P'(C) = P(f^{-1}(C))$. Then P' is a probability measure on $(\Omega', \mathcal{F}')$. The measure P' is called the measure *induced* by f, and is denoted by $f(P)$. If $\mathcal{P}$ is a discrete probability space and f is a function defined on Ω, then f can be *extended* to $\mathcal{P}$ by defining $f(\mathcal{P})$ to be the discrete probability space $(f(\Omega), 2^{f(\Omega)}, f(P))$.

2.2 Probabilistic Automata

A labeled transition system, also called an *automaton*, is a state machine with labeled transitions. Each transition leaves from a state and leads to the occurrence of a label, also called an *action*, and to a state. A probabilistic automaton is like an ordinary automaton except that each transition leads to an action and to a probability distribution over states.

Resolving the nondeterminism in an automaton leads to a linear chain of states interleaved with actions, called an *execution* or a *computation*; resolving the nondeterminism in a probabilistic automaton leads to a Markov chain structure since each transition leads probabilistically to more than one state. Such a structure is called a *probabilistic execution*. A probabilistic execution can be visualized as a probabilistic automaton that enables at most one transition from each state (a *fully probabilistic automaton*). Due to the complex structure of a probabilistic execution, it is convenient to view it as a special case of a probabilistic automaton; in this way the analysis of a probabilistic execution is simplified.

However, nondeterminism could be resolved also using randomization: a scheduler for n processes running in parallel could choose the next process to schedule by rolling an n-side dice; similarly, if some actions model the input of an external environment, the environment could provide the input at random or could provide no input with some non-zero probability. Thus, in a probabilistic execution the transition that leaves from a state may lead to a probability distribution over both actions and states and also over deadlock (no input). This new kind of transition is not part of our informal definition of a probabilistic automaton; yet, it is still convenient to view a probabilistic execution as a probabilistic automaton.

Thus, our definition of a probabilistic automaton allows for a transition to lead to probability distributions over actions and states and over a symbol δ that models deadlock; however, except for the handling of probabilistic executions, we concentrate on *simple probabilistic automata*, which allow only probabilistic choices over states within a transition.

Definition 1. A *probabilistic automaton* M consists of four components:

1. a set $states(M)$ of states,
2. a nonempty set $start(M) \subseteq states(M)$ of start states,
3. an action signature $sig(M) = (ext(M), int(M))$ where $ext(M)$ and $int(M)$ are disjoint sets of *external* and *internal* actions, respectively,

4. a transition relation $trans(M) \subseteq states(M) \times Probs((acts(M) \times states(M)) \cup \{\delta\})$, where $acts(M)$ denotes the set $ext(M) \cup int(M)$ of actions, and δ is a special symbol not in $acts(M)$.

A probabilistic automaton M is *simple* if for each transition $(s, \mathcal{P})$ of $trans(M)$ there is an action a such that $\Omega \subseteq \{a\} \times states(M)$. In such a case a transition can be represented alternatively as $(s, a, \mathcal{P}')$, where $\mathcal{P}' \in Probs(states(M))$, and is called a *simple transition*.

A probabilistic automaton is *fully probabilistic* if it has a unique start state and from each state there is at most one transition enabled. □

Observe that an ordinary automaton is a special case of a probabilistic automaton where each transition leads to a Dirac distribution. We defer the reader to [13] for a comparison with other models.

2.3 Executions and Probabilistic Executions

We now move to the notion of an execution, which is the result of resolving both the nondeterministic and the probabilistic choices in a probabilistic automaton; it corresponds to the notion of an execution for ordinary automata. We introduce also a notion of an extended execution, which we use later to study the probabilistic behavior of a probabilistic automaton.

Definition 2. An *execution* α of a probabilistic automaton M is a (finite or infinite) sequence of alternating states and actions starting with a start state and, if the execution is finite, ending in a state, $\alpha = s_0 a_1 s_1 a_2 s_2 \cdots$, where for each i there exists a probability space $\mathcal{P}$ such that $(s_i, \mathcal{P}) \in trans(M)$ and $(a_{i+1}, s_{i+1}) \in \Omega$. Denote by $fstate(\alpha)$ the first state of α, and, if α is finite, denote by $lstate(\alpha)$ the last state of α. Denote by $exec^*(M)$ and $exec(M)$ the sets of finite and all executions of M, respectively.

An extended execution of M is either an execution of M, or a sequence $\alpha = s_0 a_1 s_1 \cdots a_n s_n \delta$ such that $s_0 a_1 s_1 \cdots a_n s_n$ is an execution of M.

An extended execution α_1 of M is a *prefix* of an extended execution α_2 of M, written $\alpha_1 \leq \alpha_2$, if either $\alpha_1 = \alpha_2$ or α_2 is obtained by extending α_1, i.e., $\alpha_1 = s_0 a_1 s_1 \cdots a_n s_n$ and $\alpha_2 = s_0 a_1 s_1 \cdots a_n s_n a_{n+1} s_{n+1} \cdots$. □

As we said already, an execution is the result of resolving both the nondeterministic and the probabilistic choices in a probabilistic automaton. The result of the resolution of nondeterministic choices only is a fully probabilistic automaton, called a *probabilistic execution*, which is the entity that replaces the executions of ordinary automata. Informally, since in ordinary automata there is no probability left once the nondeterminism is resolved, the executions and probabilistic executions of an ordinary automaton describe the same objects. Before giving the formal definition of a probabilistic execution, we introduce *combined transitions*, which allow us to express the ability to resolve the nondeterminism using probability. Informally, a combined transition leaving from a state s is obtained by choosing a transition that leaves from s probabilistically, and then behaving according to the transition chosen. Among the choices it is possible not to schedule any transition. This possibility is expressed by the term $(1 - \sum_i p_i)$ in the probability of δ in the definition below.

Definition 3. Given a probabilistic automaton M, a finite or countable set $\{\mathcal{P}_i\}_i$ of probability distributions of $Probs((acts(M) \times states(M)) \cup \{\delta\})$, and a weight $p_i > 0$ for each i such that $\sum_i p_i \leq 1$, the combination $\sum_i p_i \mathcal{P}_i$ of the distributions $\{\mathcal{P}_i\}_i$ is the probability space $\mathcal{P}$ such that

- $\Omega = \begin{cases} \cup_i \Omega_i & \text{if } \sum_i p_i = 1 \\ \cup_i \Omega_i \cup \{\delta\} & \text{if } \sum_i p_i < 1 \end{cases}$
- $\mathcal{F} = 2^{\Omega}$
- for each $(a, s) \in \Omega$, $P[(a, s)] = \sum_{i \mid (a,s) \in \Omega_i} p_i P_i[(a, s)]$
- if $\delta \in \Omega$, then $P[\delta] = (1 - \sum_i p_i) + \sum_{i \mid \delta \in \Omega_i} p_i P_i[\delta]$.

A pair $(s, \mathcal{P})$ is a *combined transition* of M if there exists a finite or countable family of transitions $\{(s, \mathcal{P}_i)\}_i$ and a set of positive weights $\{p_i\}_i$ with $\sum_i p_i \leq 1$, such that $\mathcal{P} = \sum_i p_i \mathcal{P}_i$. Denote $(s, \mathcal{P})$ by $\sum_i p_i (s, \mathcal{P}_i)$. □

We are now ready to define a probabilistic execution. Informally, a probabilistic execution can be seen as the result of unfolding the transition relation of a probabilistic automaton and then choosing probabilistically a transition from each state.

Definition 4. Let α be a finite execution of a probabilistic automaton M. Define a function $\alpha\frown$ that applied to a pair (a, s) returns $(a, \alpha a s)$, and applied to δ returns δ. Recall from the last paragraph of Section 2.1 that the function $\alpha\frown$ can be extended to discrete probability spaces.

A *probabilistic execution* of a probabilistic automaton M, is a fully probabilistic automaton, denoted by H, such that

1. $states(H) \subseteq exec^*(M)$. Let q range over states of probabilistic executions.
2. $start(H)$ contains a start state of M.
3. for each transition $tr = (q, \mathcal{P})$ of H there is a combined transition $tr' = (lstate(q), \mathcal{P}')$ of M, called the *corresponding combined transition*, such that $\mathcal{P} = q \frown \mathcal{P}'$.
4. each state of H is reachable and enables one transition, where a state q of H is reachable if there is an execution of H whose last state is q.

Denote by $prexec(M)$ the set of probabilistic executions of M. Also, denote by q_0^H the start state of a generic probabilistic execution H, and for each transition $(q, \mathcal{P})$ of H, denote $\mathcal{P}$ by $\mathcal{P}_q^H$.

A *probabilistic execution* is *complete* iff for each state q, $\delta \in \Omega_q^H$ implies that no transition is enabled in M from $lstate(q)$. Denote by $cprexec(M)$ the set of complete probabilistic executions of M.

Example 1. Figure 1 contains an example of a probabilistic execution. The object denoted by H is a probabilistic execution of the probabilistic automaton denoted by M. For notational convenience, in the representation of a probabilistic execution H we do not write explicitly the full names of the states of H since the full names are derivable from the position of each state in the diagram; moreover, whenever a state q enables the transition $(q, \mathcal{D}(\delta))$ we do not draw any arc leaving from the state of the diagram that represents q. □

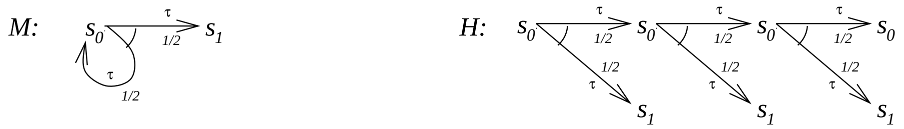

Fig. 1. Probabilistic executions.

There is a strong correspondence between the extended executions of a probabilistic automaton and the extended executions of one of its probabilistic executions. We express this correspondence by means of an operator $\downarrow$ that takes an extended execution α of H and gives back the corresponding extended execution $\alpha\!\downarrow$ of M, and an operator $\uparrow q_0^H$ that takes an extended execution α of M and gives back the corresponding extended execution $\alpha\!\uparrow\! q_0^H$ of H if it exists.

2.4 Events

We now define a probability space $(\Omega_H, \mathcal{F}_H, P_H)$ for a probabilistic execution H, so that it is possible to analyze the probabilistic behavior of a probabilistic automaton once the nondeterminism is resolved. The sample space Ω_H is the set of extended executions of M that represent complete extended executions of H, where an extended execution α of H is complete iff it is either infinite or $\alpha = \alpha'\delta$ and $\delta \in \Omega^H_{lstate(\alpha')}$. For each finite extended execution α of M, let C_α, the *cone* with prefix α, be the set $\{\alpha' \in \Omega_H \mid \alpha \leq \alpha'\}$, and let $\mathcal{C}_H$ be the class of cones for H. The probability $\mu_H(C_\alpha)$ of the cone C_α is the product of the probabilities associated with each edge that generates α in H. Formally, if $\alpha = q_0^H a_1 s_1 \cdots s_{n-1} a_n s_n$, then $\mu_H(C_\alpha) \triangleq P^H_{q_0}[(a_1, q_1)] \cdots P^H_{q_{n-1}}[(a_n, q_n)]$, where each q_i is defined to be $q_0^H a_1 s_1 \cdots a_i s_i$, and if $\alpha = q_0^H a_1 s_1 \cdots s_{n-1} a_n s_n \delta$, then $\mu_H(C_\alpha) \triangleq P^H_{q_0}[(a_1, q_1)] \cdots P^H_{q_{n-1}}[(a_n, q_n)] P^H_{q_n}[\delta]$, where each q_i is defined to be $q_0^H a_1 s_1 \cdots a_i s_i$. In [13] it is shown that there is a unique measure $\bar{\mu}_H$ that extends μ_H to the σ-field $\sigma(\mathcal{C}_H)$ generated by $\mathcal{C}_H$, i.e., the smallest σ-field that contains $\mathcal{C}_H$. Then, $\mathcal{F}_H$ is $\sigma(\mathcal{C}_H)$ and P_H is $\bar{\mu}_H$. With this definition it is possible to show that any union of cones is measurable.

2.5 Convergence

An execution α of a probabilistic automaton M is *divergent* if it is infinite and contains finitely many external actions. An execution α is *convergent* if it is not divergent. A probabilistic automaton M is *strongly convergent* if there are no divergent executions of M. A probabilistic automaton M is *probabilistically convergent* if there are no probabilistic executions of M whose divergent executions have a non-zero probability.

2.6 Parallel Composition

The parallel composition operator for probabilistic automata is defined in the CSP style [6], i.e., by synchronizing two probabilistic automata on their common actions.

As outlined in [4], it is not clear how to define a parallel composition operator for general probabilistic automata that extends the CSP synchronization style; thus, we define it only for simple probabilistic automata, and we concentrate on simple probabilistic automata for the rest of this paper. We use general probabilistic automata only for the analysis of probabilistic executions. The reader is referred to [13] for more details.

Definition 5. Two simple probabilistic automata M_1 and M_2 are said to be *compatible* if $int(M_1) \cap acts(M_2) = \emptyset$, and $int(M_2) \cap acts(M_1) = \emptyset$.

The *parallel composition* $M_1 \| M_2$ of two compatible simple probabilistic automata M_1 and M_2 is the simple probabilistic automaton M such that $states(M) = states(M_1) \times states(M_2)$, $start(M) = start(M_1) \times start(M_2)$, $ext(M) = ext(M_1) \cup ext(M_2)$, $int(M) = int(M_1) \cup int(M_2)$, and the transition relation satisfies the following: $((s_1, s_2), a, \mathcal{P}) \in trans(M)$ iff $\mathcal{P} = \mathcal{P}_1 \otimes \mathcal{P}_2$, such that

1. if $a \in acts(M_1)$ then $(s_1, a, \mathcal{P}_1) \in trans(M_1)$, else $\mathcal{P}_1 = \mathcal{D}(s_1)$, and
2. if $a \in acts(M_2)$ then $(s_2, a, \mathcal{P}_2) \in trans(M_2)$, else $\mathcal{P}_2 = \mathcal{D}(s_2)$. □

2.7 Trace Distribution Precongruence

Definition 6. Let H be a probabilistic execution of a probabilistic automaton M. For each extended execution α of M, let $trace(\alpha)$ denote the ordered sequence of external actions of M that appear in α.

Let f be a function from Ω_H to $\Omega = ext(M)^* \cup ext(M)^\omega$ that assigns to each execution of Ω_H its trace. The *trace distribution* of H, denoted by $tdistr(H)$, is the probability space $(\Omega, \mathcal{F}, P)$ where $\mathcal{F}$ is the σ-field generated by the cones C_β, where β is an element of $ext(M)^*$, and $P = f(P_H)$. The fact that f is measurable follows from standard arguments. Denote a generic trace distribution by $\mathcal{D}$. A trace distribution of a probabilistic automaton M is the trace distribution of one of the probabilistic executions of M. □

Given two probabilistic executions H_1 and H_2, it is possible to check whether $tdistr(H_1) = tdistr(H_2)$ just by verifying that $P_{tdistr(H_1)}[C_\beta] = P_{tdistr(H_2)}[C_\beta]$ for each finite sequence of actions β. This follows from standard measure theory arguments.

It is easy to see that trace distributions extend the traces of ordinary automata: the trace distribution of a linear probabilistic execution is a distribution that associates probability 1 to a unique trace. It is easy as well to see that the notion of prefix for traces extends to the probabilistic framework. A trace distribution $\mathcal{D}$ is a *prefix* of a trace distribution $\mathcal{D}'$, denoted by $\mathcal{D} \leq \mathcal{D}'$, iff for each finite trace β, $P_{\mathcal{D}}[C_\beta] \leq P_{\mathcal{D}'}[C_\beta]$. Thus, two trace distributions are equal iff each one is a prefix of the other.

Definition 7. Let M_1, M_2 be two probabilistic automata with the same external actions. The *trace distribution preorder* is defined as follows.

$$M_1 \sqsubseteq_D M_2 \text{ iff } tdistrs(M_1) \subseteq tdistrs(M_2). \qquad \square$$

The trace distribution preorder is a direct extension of the trace preorder of ordinary automata; however, it is not a precongruence, as it is shown in [12]. This leads to the following definition.

Definition 8. Let M_1, M_2 be two probabilistic automata with the same external actions. The *trace distribution precongruence*, denoted by $\sqsubseteq_T$, is the coarsest precongruence that is contained in the trace distribution preorder. $\square$

In [13] the trace distribution precongruence is characterized by means of a *principal context*, denoted by C_P and represented in Figure 2, in the following sense.

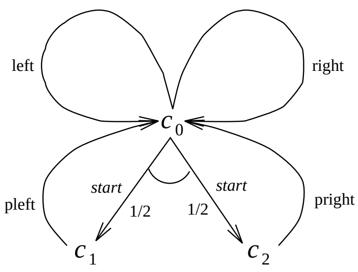

Fig. 2. The principal context.

Proposition 9. *$M_1 \sqsubseteq_T M_2$ iff $M_1 \| C_P \sqsubseteq_D M_2 \| C_P$.* $\square$

3 Failure Distribution Precongruence

In this section we extend to the probabilistic framework the failures of [2]. For each state s of a probabilistic automaton M define $wenabled(s)$ to be the set of actions a that can be performed from s up to internal transitions. A *failure* of a probabilistic automaton M is a pair (β, F), where β is a finite trace and F is a set of actions, such that there exists a finite execution α of M with $\beta = trace(\alpha)$ and $F \cap wenabled(lstate(\alpha)) = \emptyset$. Its meaning is that M can produce the external trace β and reach a state from which no action from F can be performed up to internal transitions.

Definition 10. Let H be a probabilistic execution of a probabilistic automaton M. Build a new object H' from H as follows. In every transition $(q, \mathcal{P})$ of H with $\delta \in \Omega$, replace δ with a collection of pairs (q, F), where F is a set of actions of M and $F \cap wenabled(lstate(q)) = \emptyset$, and assign to each pair (q, F) an arbitrary probability p_F so that $\sum p_F = P[\delta]$. In other words, H' is obtained from H by replacing deadlocks with failure sets.

Let f be a function from $\Omega_{H'}$ to $\Omega = ext(M)^* \cup (ext(M)^* \times 2^{acts(M)}) \cup ext(M)^\omega$ that assigns to each execution of $\Omega_{H'}$ its trace or failure as follows:

$$f(\alpha) = \begin{cases} trace(\alpha) & \text{if } trace(\alpha) \in ext(M)^\infty \\ (trace(\alpha), F) & \text{if } lstate(\alpha) = (q, F) \text{ for some } q \\ (trace(\alpha), \emptyset) & \text{if } \alpha \text{ is infinite and } trace(\alpha) \text{ is finite.} \end{cases}$$

A *failure distribution* of H, denoted by $fdistr(H)$, is the probability space $(\Omega, \mathcal{F}, P)$ where $\mathcal{F}$ is the σ-field generated by the cones C_β and $C_{(\beta,F)}$, where β is an element of $ext(M)^*$ and F is a set of actions of M, and $P = f(P_H)$. The fact that f is measurable follows from standard arguments. Denote a generic failure distribution by $\mathcal{F}$. A failure distribution of a probabilistic automaton M is a failure distribution of one of the probabilistic executions of M. Denote by $fdistrs(M)$ the set of failure distributions of M. □

Similar to the case for trace distributions, a failure distribution preorder can be defined as inclusion of failure distributions. Then, the *failure distribution precongruence*, denoted by $\sqsubseteq_F$, is the coarsest precongruence contained in the failure distribution preorder. The principal context characterization is valid also for failure distributions.

Proposition 11. *$M_1 \sqsubseteq_F M_2$ iff $fdistrs(M_1 \| C_P) \subseteq fdistrs(M_2 \| C_P)$.* □

Observe that a failure distribution as defined so far is not a real extension of the failures of [2] since in a failure distribution the infinite traces may have a non zero probability. Later in the paper we define finite failure distributions, which can be regarded as real extensions of the failures of [2].

4 The Probabilistic May and Must Preorders

Let M be a probabilistic automaton. A *test* for M is a probabilistic automaton T, compatible with M, whose actions include the external actions of M and a collection of at most countably many *success* actions $w_1, w_2, \ldots$. Given a probabilistic execution H of $M \| T$, let $W(H)$ denote the vector $(p_1, p_2, \ldots)$, where each p_i denotes the probability that action w_i occurs in H, i.e., the probability in $\mathcal{P}_H$ of the executions that contain an occurrence of action w_i. Given two probabilistic automata M_1, M_2, define two *testing* preorders as follows.

$$M_1 \sqsubseteq_{MAY} M_2 \text{ iff } \forall_T \forall_{H_1 \in cprexecs(M_1\|T)} \exists_{H_2 \in cprexecs(M_2\|T)} W(H_1) \leq W(H_2),$$
$$M_1 \sqsubseteq_{MUST} M_2 \text{ iff } \forall_T \forall_{H_2 \in cprexecs(M_2\|T)} \exists_{H_1 \in cprexecs(M_1\|T)} W(H_1) \leq W(H_2).$$

The definitions of the may and must preorders differ only in the use of the existential and universal quantifications. They are inspired by the testing preorders of [3] and by the probabilistic preorders of [7, 8]. The may preorder states that for each test T, if there is a possibility that M_1 passes T with a probability vector W, then also M_1 has a possibility to pass T with a probability vector at least W. The must preorder states that for each test T, if there is a possibility that M_2 passes T with a probability vector W, then M_2 has a possibility to pass T with a probability vector at most W.

An alternative definition of the testing preorders can be given as follows. A test is a pair $(T, \bar{W})$, where T is a probabilistic automaton compatible with M whose actions include the external actions of M and a collection of at most countably many *success* actions $w_1, w_2, \ldots$, and $\bar{W}$ is a vector of probabilities. Then,

$$M \text{ MAY } (T, \bar{W}) \text{ iff } \exists_{H \in cprexec(M\|T)} \bar{W} \leq W(H),$$
$$M \text{ MUST } (T, \bar{W}) \text{ iff } \nexists_{H \in cprexec(M\|T)} W(H) \leq \bar{W},$$

and the testing preorders are defined as

$$M_1 \sqsubseteq_{MAY} M_2 \text{ iff } \forall_{(T,\bar{W})} M_1 \text{ MAY } (T,\bar{W}) \Rightarrow M_2 \text{ MAY } (T,\bar{W}),$$
$$M_1 \sqsubseteq_{MUST} M_2 \text{ iff } \forall_{(T,\bar{W})} M_1 \text{ MUST } (T,\bar{W}) \Rightarrow M_2 \text{ MUST } (T,\bar{W}).$$

A reader may be tempted to replace the non-existential quantification with a universal quantification in the definition of M MUST $(T,\bar{W})$ and to reverse the inequality. That is, M MUST $(T,\bar{W})$ iff $\forall_{H\in cprexec(M\|T)} \bar{W} \leq W(H)$. Such an alternative definition would not work because the closure of the set of vectors W of a probabilistic automaton does not have a unique minimum element in general.

The testing preorders defined so far are equivalent to the trace distribution precongruence and, under the restriction of strong convergence, to the failure distribution precongruence.

Theorem 12. *$M_1 \sqsubseteq_{MAY} M_2$ iff $M_1 \sqsubseteq_T M_2$.*

Proof sketch. $\sqsubseteq_T$ implies $\sqsubseteq_{MAY}$ directly since the vector W is derived from the trace distributions of a probabilistic automaton.

Conversely, let $M_1 \sqsubseteq_{MAY} M_2$, and let $\mathcal{D}$ be a trace distribution of $M_1\|C_P$. Build a test T as follows. The states of T are the symbol δ and all pairs of the form (β, b) where $b \in \{0,1\}$ and β is a finite trace such that $P_{\mathcal{D}}[C_\beta] > 0$. For each state $(\beta, 0)$ there is a transition $(\beta,0) \xrightarrow{w_\beta} (\beta,1)$; for each state $(\beta a, 0)$ there is a transition $(\beta,1) \xrightarrow{a} (\beta a, 0)$; for each finite trace β such that $P_{\mathcal{D}}[\{\beta\}] > 0$ there is a transition $(\beta,1) \xrightarrow{w_{\beta\delta}} \delta$.

For simplicity we assume that there is a probabilistic execution of $M_1\|C_P$ whose set of divergent executions has probability 0 and whose trace distribution is $\mathcal{D}$. The handling of divergences is more complex since we need to detect them by scheduling some additional actions of C_P. Consider now a probabilistic execution H_1 of $M_1\|C\|T$ such that in $W(H_1)$ the probability of occurrence of each action w_β is $P_{\mathcal{D}}[C_\beta]$, and the probability of each action $w_{\beta\delta}$ is $P_{\mathcal{D}}[\{\beta\}]$. Such a probabilistic execution can be built because $\mathcal{D}$ is a trace distribution of $M_1\|C_P$ and because of our simplifying assumption.

From $M_1 \sqsubseteq_{MAY} M_2$, there is a probabilistic execution H_2 of $M_2\|C_P\|T$ such that $W(H_1) \leq W(H_2)$ (we use the test $C_P\|T$ here). Next, we observe that also in $W(H_2)$ the probability of each action w_β is $P_{\mathcal{D}}[C_\beta]$, and the probability of each $w_{\beta\delta}$ is $P_{\mathcal{D}}[\{\beta\}]$, since otherwise we would be able to find events of $\mathcal{F}_{H_2}$ with a probability greater than 1. Therefore, from the construction of T, $\mathcal{D}$ is a trace distribution of $M_2\|C_P$. □

Theorem 13. *If M_1 and M_2 are probabilistically convergent, then $M_1 \sqsubseteq_{MUST} M_2$ iff $M_2 \sqsubseteq_F M_1$.*

Proof sketch. Let $M_1 \sqsubseteq_{MUST} M_2$, and let $\mathcal{F}$ be a failure distribution of $M_2\|C_P$. Build a test T as follows. The states of T are the symbol δ, all pairs of the form (β, b) where $b \in \{0,1\}$ and β is a finite trace such that $P_{\mathcal{F}}[C_\beta] > 0$, all pairs of the form (β, F) and (β, F'), where $P_{\mathcal{F}}[\{(\beta,F)\}] > 0$, and all pairs of the form $(\beta, \emptyset)$ and $(\beta, \emptyset')$, where $P_{\mathcal{F}}[C_\beta] > 0$. In reality some of the pairs $(\beta, \emptyset')$ are not needed; we just include them to keep the construction simple.

For each state $(\beta, 0)$ there is a transition $(\beta, 0) \xrightarrow{w_\beta} (\beta, 1)$; for each state $(\beta a, 0)$ there is a transition $(\beta, 1) \xrightarrow{a} (\beta a, 0)$; for each state (β, F) there is a transition $(\beta, 1) \xrightarrow{w_{\beta F}} (\beta, F)$ and a transition $(\beta, F') \xrightarrow{w'_{\beta F}} \delta$; furthermore, for each $a \in F$, there is a transition $(\beta, F) \xrightarrow{a} (\beta, F')$; finally, for each state (β, F') and each prefix β' of β there is a transition $(\beta', 1) \xrightarrow{w_{\beta F}} (\beta, F')$.

Consider now a complete probabilistic execution H_2 of $M_2 \| C_P \| T$ such that in $W(H_2)$ the probability of each action w_β is $P_{\mathcal{F}}[C_\beta]$, the probability of each $w_{\beta F}$ is $P_{\mathcal{D}}[\{(\beta, F)\}]$, and the probability of each $w'_{\beta F}$ is 0. Such a probabilistic execution can be built because $\mathcal{F}$ is a failure distribution of $M_2 \| C_P$ and M_2 is probabilistically convergent.

From $M_1 \sqsubseteq_{MUST} M_2$, there is a complete probabilistic execution H_1 of $M_1 \| C_P \| T$ such that $W(H_1) \leq W(H_2)$ (we use the test $C_P \| T$). Observe that also in H_1 each action $w'_{\beta F}$ occurs with probability 0, which means that none of the transitions labeled with $w_{\beta F}$ are performed from states different from $(\beta, 1)$ (otherwise $w'_{\beta F}$ would be the only action enabled). Furthermore, from the completeness of H_1, since actions of the form $w_{\beta F}$ continue to be enabled unless one of them occurs, and since M_1 is probabilistically convergent, the probability in H_2 of actions of the form $w_{\beta F}$ plus infinite traces is 1. Thus, we use the probabilistic convergence of M_1 and M_2 to derive $W(H_1) = W(H_2)$, and consequently that $\mathcal{F}$ is a failure distribution of $M_1 \| C_P$.

Conversely, suppose that $M_2 \sqsubseteq_F M_1$, and let H_2 be a complete probabilistic execution of $M_2 \| T$. From $M_2 \sqsubseteq_F M_1$ there is a probabilistic execution H_1 of $M_1 \| T$ such that $W(H_1) = W(H_2)$ and such that all failures are with the set $ext(M_1 \| T)$. Therefore, H_1 is complete. □

From the proof sketches of Theorems 12 and 13 we understand the structure of the *canonical* tests for our preorders, i.e., the minimal tests that are sufficient to characterize the testing preorders. A canonical test for the may preorder is the principal context in parallel with a nondeterministic automaton that has a tree-like structure. In the test all the external actions are interleaved with success actions, and each finite trace β to be observed is marked by a special success action $w_{\beta\delta}$ that leads to a deadlock state. Essentially a test has the structure of a trace distribution, where each success action is used to detect the probability of a specific cone. In this sense our tests are similar to the tests of [3]. The success actions that are interleaved with the external actions are the main difference with respect to [3]; these actions are needed because our testing preorders are sensitive to infinite traces; In the following section we study preorders that are not sensitive to infinite traces, thus removing the interleaved success actions.

The canonical tests for the must preorder are very similar to the canonical tests for the may preorder. The main difference is that the actions that test for finite traces are replaced by gadgets that test for failures. Each gadget is based on two actions: the first success action is used to show that we are testing for a specific failure set F; the second success action becomes enabled only if some action from F occurs after the occurrence of the first success action. Informally, we regard a test as successful if the second success action occurs with probability 0. To make sure that some failure is tested before stopping, the canonical tests have several transitions labeled with the first kind of success actions that lead to states where only success actions of the second kind are enabled. Our canonical tests are similar to the canonical tests of [3]; the main difference is that here, since we must check for several failures, we use success actions also to signal that we are testing a specific failure.

5 Finite Trace and Failure Distributions

We now impose a restriction to trace distributions and failure distributions, namely, that for each trace or failure distribution there is a constant k such that all the probability is concentrated on traces or failures of length at most k. We call the corresponding trace and failure distributions *finite*, and we denote the corresponding precongruence relations by $\sqsubseteq_{Tf}$ and $\sqsubseteq_{Ff}$, respectively. In this case for each test T there must be a number k such that no execution of T contains more than k external actions. We denote the corresponding testing preorders by $\sqsubseteq_{MAYf}$ and $\sqsubseteq_{MUSTf}$.

The principal context characterization changes in this case, since the method of [13] to reduce the branching structure of a distinguishing context does not work any more. The new principal context is represented in Figure 3.

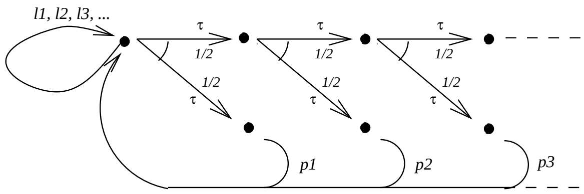

Fig. 3. The principal context for the finite preorders.

Theorem 14. *If M_1 and M_2 are probabilistically convergent, then $M_1 \sqsubseteq_{MAYf} M_2$ iff $M_1 \sqsubseteq_{Tf} M_2$.*

Proof sketch. $\sqsubseteq_{Tf}$ implies $\sqsubseteq_{MAYf}$ directly since the vector W is derived from the finite trace distributions of a probabilistic automaton in parallel with a finite test.

Conversely, let $M_1 \sqsubseteq_{MAYf} M_2$, and let $\mathcal{D}$ be a finite trace distribution of $M_1 \| C_{Pf}$. We build a test T as follows. The states of T are the symbol δ, and all finite traces β such that $P_{\mathcal{D}}[C_\beta] > 0$. For each state βa there is a transition $\beta \xrightarrow{a} \beta a$; for each state β such that $P_{\mathcal{D}}[\{\beta\}] > 0$ there is a transition $\beta \xrightarrow{w_{\beta\delta}} \delta$. The test T is guaranteed to be finite because $\mathcal{D}$ is a finite trace distribution. Then the conclusion is derived as in the proof of Theorem 12, where we do not need to deal with the interleaved success actions and where the probabilistic convergence of M_1 is used to rule out the case of a trace distribution that derives only from probabilistic executions that diverge with a non-zero probability. □

Theorem 15. *If M_1 and M_2 are probabilistically convergent, then $M_1 \sqsubseteq_{MUSTf} M_2$ iff $M_2 \sqsubseteq_{Ff} M_1$.*

Proof sketch. Let $M_1 \sqsubseteq_{MUSTf} M_2$, and let $\mathcal{F}$ be a finite failure distribution of $M_2 \| C_{Pf}$. We build a test T as follows. The states of T are the symbol δ, all finite traces β such that $P_{\mathcal{F}}[C_\beta] > 0$, all pairs of the form (β, F) and (β, F'), where $P_{\mathcal{F}}[\{(\beta, F)\}] > 0$, and all pairs of the form $(\beta, \emptyset)$ and $(\beta, \emptyset')$, where $P_{\mathcal{F}}[C_\beta] > 0$. For each state βa there is a transition $\beta \xrightarrow{a} \beta a$; for each state (β, F) there is a transition $\beta \xrightarrow{w_{\beta F}} (\beta, F)$ and a transition $(\beta, F') \xrightarrow{w_{\beta F'}} \delta$; furthermore, for each $a \in F$, there is a transition $(\beta, F) \xrightarrow{a} (\beta, F')$; finally, for each state (β, F') and each prefix β' of β there is a transition $\beta' \xrightarrow{w_{\beta F}} (\beta, F')$.

The test T is guaranteed to be finite because $\mathcal{F}$ is a finite failure distribution. Then the conclusion is derived as in the proof of Theorem 13, where we do not need to deal with the interleaved success actions.

Conversely, suppose that $M_2 \sqsubseteq_{Ff} M_1$, and let H_2 be a complete probabilistic execution of $M_2 \| T$. From $M_2 \sqsubseteq_{Ff} M_1$ there is a probabilistic execution H_1 of $M_1 \| T$ such that $W(H_1) = W(H_2)$ and such that all failures are with the set $ext(M_1 \| T)$. Therefore, H_1 is complete. □

The canonical tests for the finite testing preorders, beside the structure of the principal context, are essentially finite depth trees without interleaved success actions. These tests are very similar to the tests of [3]. A difference between our testing preorders and those of [3] is that our preorders are totally insensitive to infinite traces, while the must preorder of [3] is partially sensitive to infinite traces if a system is not finite branching.

6 Continuity and Finiteness of Tests

In this section we study sufficient conditions for finite trace and failure distributions to describe infinite failure and trace distributions as well. In the nondeterministic case strong convergence and finite branching are sufficient; here we show that the same holds in the probabilistic case. We describe also additional restrictions that allow us to consider only vectors with finitely many success actions, thus reducing our canonical tests to be really finite.

Before stating the first result, we need to say what we mean by finite branching. In a probabilistic automaton there are two kinds of nondeterminism: the *pure nondeterminism*, which is due to the choice between different transitions, and the *probabilistic nondeterminism*, which is due to the choice of a next state once a transition is chosen. By finite branching we mean that there are finitely many start states and from every state there are finitely many transitions enabled. We do not impose any restriction on the probability distributions associated with each transition.

Theorem 16. *Let M be a probabilistically convergent, finitely branching probabilistic automaton, and let $\mathcal{D}_1, \mathcal{D}_2, \ldots$ be a chain of finite trace distributions of M. Let $\mathcal{D}$ be $\lim_i \mathcal{D}_i$. Then $\mathcal{D}$ is a trace distribution of M.*

Proof sketch. Consider a collection of probabilistic executions $H_1, H_2, \ldots$, with the same start state s_0, such that, for each i, $tdistr(H_i) = \mathcal{D}_{f(i)}$, where f is a monotonic increasing function from naturals to naturals. Such collection of probabilistic executions exists since M has finitely many start states. Consider the set of probability distributions over states that describe what is reached after the first transition in the H_i's. From the finite branching structure of M, the set of distributions over states reachable from s_0 in one transition is compact under the distance $d(\mathcal{P}_1, \mathcal{P}_2) = \sum_{s \in \Omega_1 \cup \Omega_2} |P_1[s] - P_2[s]|$. Thus, the distributions reached in the H_i's have at least one accumulation point.

We build the transition relation of a probabilistic execution H inductively as follows. For the first transition of H choose one of the accumulation points described above. For the i^{th} level transitions, assume that the transitions at previous levels are defined. For each $\epsilon > 0$, let $\mathcal{H}_\epsilon$ be the set of probabilistic executions H' that at the first $i-1$ levels are within distance ϵ of the probabilities of the states in H. Each $\mathcal{H}_\epsilon$ is not empty. For each $k \geq 0$, let H'_k be an element of $\mathcal{H}_{1/k}$. Then, from the finite branching structure of M, the first i-level distributions of the H'_is have an accumulation point that at the first $i-1$ levels coincides with the distributions of the states of H. Choose such point for the i-level distribution of H.

Let $\mathcal{D}_H$ denote $tdistr(H)$. We need to show that $\mathcal{D}_H = \mathcal{D}$. To do this we show that for each finite trace β and each $\epsilon > 0$ there is an i such that for each $j \geq i$, $|P_H[C_\beta] - P_{H_j}[C_\beta]| \leq \epsilon$.

First we observe that, from probabilistic convergence, for each fixed length l and for each $\epsilon > 0$ there is a constant $k_{l,\epsilon}$ such that in all probabilistic executions of M the set of executions that at length at least $k_{l,\epsilon}$ have less than l external actions has probability at most ϵ. Thus, we derive that there is a depth k such that in every probabilistic execution of M the probability of β truncated at depth k is within $\epsilon/3$ of its real value. From the construction of H, there are infinitely many H_i's such that the probability of β truncated at level k in H and H_i differ at most by $\epsilon/3$. Thus, there is a minimum number l such that, for $i \geq l$, the probability of β in H and H_i differ at most by ϵ. ◻

Theorem 17. *Let M be a probabilistically convergent, finitely branching probabilistic automaton, and let $\mathcal{F}_1, \mathcal{F}_2, \ldots$ be a chain of finite failure distributions of M. Let $\mathcal{F}$ be $\lim_i \mathcal{F}_i$. Then $\mathcal{F}$ is a failure distribution of M.*

Proof sketch. Same idea as in Theorem 16 where we extend probabilistic executions with failure sets as well. ◻

With the results above we derive that in the strongly convergent and finite branching case, finite trace and failure distributions give the same information as general trace and failure distributions. Thus, the general and finite testing preorders coincide.

We conclude this section by stating sufficient conditions for the tests to have only finitely many success actions. Since the success actions are used to study the relative probabilities of different traces, we need to make sure that every finite trace distribution contains finitely many traces with a non-zero probability. This is guaranteed by strong convergence (not by probabilistic convergence) and by the requirement that every transition leads to probability distributions over finitely many states.

7 Concluding Remarks

We have defined a theory of testing for probabilistic automata and we have characterized our testing preorders in terms of trace and failure distributions, which are a probabilistic generalization of the traces and failures of ordinary, nondeterministic, automata. The main idea behind the new theory of testing is that a test is equipped with at most countably many success actions rather than a single action as in [3].

We have defined finite versions of the testing preorders that generalize the testing preorders of [3], and we have characterized the new preorders in terms of finite trace and failure distributions. We have shown that for finite branching and probabilistically convergent systems the finite and general testing preorders coincide. Finally we have given sufficient conditions for distinguishing processes by means of tests with finitely many success actions.

Although we have shown that there is a natural extension of the theory of testing to probabilistic automata and that the characterization in terms of traces and failures extend as well, the underlying theory is not as simple as in the non probabilistic case. In particular we do not know an explicit definition of parallel composition of the trace and failure distribution sets. Studying such definitions would be helpful in understanding better the theory.

References

1. S. Aggarwal. Time optimal self-stabilizing spanning tree algorithms. Technical Report MIT/LCS/TR-632, MIT Laboratory for Computer Science, 1994. Master's thesis.
2. S.D. Brookes, C.A.R. Hoare, and A.W. Roscoe. A theory of communicating sequential processes. *Journal of the ACM*, 31(3):560–599, 1984.
3. R. De Nicola and M. Hennessy. Testing equivalences for processes. *Theoretical Computer Science*, 34:83–133, 1984.
4. H. Hansson. *Time and Probability in Formal Design of Distributed Systems*, volume 1 of *Real-Time Safety Critical Systems*. Elsevier, 1994.
5. M. Hennessy. *Algebraic Theory of Processes*. MIT Press, Cambridge, MA, 1988.
6. C.A.R. Hoare. *Communicating Sequential Processes*. Prentice-Hall International, Englewood Cliffs, 1985.
7. B. Jonsson, C. Ho-Stuart, and W. Yi. Testing and refinement for nondeterministic and probabilistic processes. In *Proceedings of the Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*, LNCS 863, pages 418–430, 1994.
8. B. Jonsson and W. Yi. Compositional testing preorders for probabilistic processes. In *Proceedings 10^{th} Annual Symposium on Logic in Computer Science,* San Diego, California. IEEE Computer Society Press, 1995.
9. N.A. Lynch, I. Saias, and R. Segala. Proving time bounds for randomized distributed algorithms. In *Proceedings of the 13^{th} Annual ACM Symposium on Principles of Distributed Computing,* Los Angeles, CA, pages 314–323, 1994.
10. G.D. Plotkin. A structural approach to operational semantics. Technical Report DAIMI FN-19, Computer science Department, Aarhus University, 1981.
11. A. Pogosyants and R. Segala. Formal verification of timed properties of randomized distributed algorithms. In *Proceedings of the 14^{th} Annual ACM Symposium on Principles of Distributed Computing,* Ottawa, Ontario, Canada, pages 174–183, August 1995.
12. R. Segala. A compositional trace-based semantics for probabilistic automata. In *Proceedings of CONCUR 95,* Philadelphia, PA, USA, LNCS 962, pages 234–248, 1995.
13. R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, MIT, Dept. of Electrical Engineering and Computer Science, 1995.
14. R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.