Compositionality for Probabilistic Automata

Nancy Lynch[1⋆], Roberto Segala[2⋆⋆], and Frits Vaandrager[3⋆⋆⋆]

[1] MIT Laboratory for Computer Science
Cambridge, MA 02139, USA
lynch@theory.lcs.mit.edu
[2] Dipartimento di Informatica, Università di Verona
Strada Le Grazie 15, 37134 Verona, Italy
roberto.segala@univr.it
[3] Nijmegen Institute for Computing and Information Sciences
University of Nijmegen
P.O. Box 9010, 6500 GL Nijmegen, The Netherlands
fvaan@cs.kun.nl

Abstract. We establish that on the domain of probabilistic automata, the trace distribution preorder coincides with the simulation preorder.

1 Introduction

Probabilistic automata [9, 10, 12] constitute a mathematical framework for modeling and analyzing probabilistic systems, specifically, systems of asynchronously interacting components that may make nondeterministic and probabilistic choices. They have been applied successfully to distributed algorithms [3, 7, 1] and practical communication protocols [13].

An important part of a system modeling framework is a notion of *external behavior* of system components. Such a notion can be used to define implementation and equivalence relationships between components. For example, the external behavior of a nondeterministic automaton can be defined as its set of *traces*—the sequences of external actions that arise during its executions [5]. Implementation and equivalence of nondeterministic automata can be defined in terms of inclusion and equality of sets of traces. By analogy, Segala [9] has proposed defining the external behavior of a probabilistic automaton as its set of *trace distributions*, and defining implementation and equivalence in terms of inclusion and equality of sets of trace distributions. Stoelinga and Vaandrager have proposed a simple testing scenario for probabilistic automata, and have proved that the equivalence notion induced by their scenario coincides with Segala's trace distribution equivalence [14].

⋆ Supported by AFOSR contract #F49620-00-1-0097, NSF grant #CCR-0121277, and DARPA/AFOSR MURI #F49620-02-1-0325.
⋆⋆ Supported by MURST projects MEFISTO and CoVer.
⋆⋆⋆ Supported by PROGRESS project TES4999: Verification of Hard and Softly Timed Systems (HaaST) and DFG/NWO bilateral cooperation project 600.050.011.01 Validation of Stochastic Systems (VOSS).

However, a problem with these notions is that trace distribution inclusion and equivalence are not compositional. To address this problem, Segala [9] defined more refined notions of implementation and equivalence. In particular, he defined the *trace distribution precongruence*, $\leq_{DC}$, as the coarsest precongruence included in the trace distribution inclusion relation. This yields compositionality by construction, but does not provide insight into the nature of the $\leq_{DC}$ relation. Segala also provided a characterization of $\leq_{DC}$ in terms of the set of trace distributions observable in a certain *principal context*—a rudimentary probabilistic automaton that makes very limited nondeterministic and probabilistic choices. However, this indirect characterization still does not provide much insight into the structure of $\leq_{DC}$, for example, it does not explain its branching structure.

In this paper, we provide an explicit characterization of the trace distribution precongruence, $\leq_{DC}$, for probabilistic automata, that completely explains its branching structure. Namely, we show that $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$ if and only if there exists a *weak probabilistic (forward) simulation relation* from $\mathcal{P}_1$ to $\mathcal{P}_2$. Moreover, we provide a similar characterization of $\leq_{DC}$ for nondeterministic automata in terms of the existence of a weak (non-probabilistic) simulation relation. It was previously known that simulation relations are sound for $\leq_{DC}$ [9], for both nondeterministic and probabilistic automata; we show the surprising fact that they are also *complete*. That is, we show that, for both nondeterministic and probabilistic automata, probabilistic contexts can observe all the distinctions that can be expressed using simulation relations.

Sections 2 and 3 contain basic definitions and results for nondeterministic and probabilistic automata, respectively, and for the preorders we consider. These sections contain no new material, but recall definitions and theorems from the literature. For a more leisurely introduction see [5, 12]. Sections 4 and 5 contain our characterization results for nondeterministic and probabilistic automata. Section 6 contains our conclusions.

A full version of this paper, including all proofs, appears in [4].

2 Definitions for Nondeterministic Automata

A *(nondeterministic) automaton* is a tuple $\mathcal{A} = (Q, \bar{q}, E, H, D)$, where Q is a set of *states*, $\bar{q} \in Q$ is a *start state*, E is a set of *external actions*, H is a set of *internal (hidden) actions* with $E \cap H = \emptyset$, and $D \subseteq Q \times (E \cup H) \times Q$ is a *transition relation*. We denote $E \cup H$ by A and we refer to it as the set of *actions*. We denote a transition (q, a, q') of D by $q \xrightarrow{a} q'$. We write $q \rightarrow q'$ if $q \xrightarrow{a} q'$ for some a, and we write $q \rightarrow$ if $q \rightarrow q'$ for some q'. We assume finite branching: for each state q the number of pairs (a, q') such that $q \xrightarrow{a} q'$ is finite. We denote the elements of an automaton $\mathcal{A}$ by $Q_{\mathcal{A}}, \bar{q}_{\mathcal{A}}, E_{\mathcal{A}}, H_{\mathcal{A}}, D_{\mathcal{A}}, A_{\mathcal{A}}, \xrightarrow{a}_{\mathcal{A}}$. Often we use the name $\mathcal{A}$ for a generic automaton; then we usually omit the subscripts, writing simply Q, $\bar{q}$, E, H, D, A, and $\xrightarrow{a}$. We extend this convention to allow indices and primes as well; thus, the set of states of automaton $\mathcal{A}'_i$ is denoted by Q'_i.

An *execution fragment* of an automaton $\mathcal{A}$ is a finite or infinite sequence $\alpha = q_0 a_1 q_1 a_2 q_2 \cdots$ of alternating states and actions, starting with a state and,

if the sequence is finite, ending in a state, where each $(q_i, a_{i+1}, q_{i+1}) \in D$. State q_0, the first state of α, is denoted by $fstate(\alpha)$. If α is a finite sequence, then the last state of α is denoted by $lstate(\alpha)$. An *execution* is an execution fragment whose first state is the start state $\bar{q}$. We let $frags(\mathcal{A})$ denote the set of execution fragments of $\mathcal{A}$ and $frags^*(\mathcal{A})$ the set of finite execution fragments. Similarly, we let $execs(\mathcal{A})$ denote the set of executions of $\mathcal{A}$ and $execs^*(\mathcal{A})$ the set of finite executions.

Execution fragment α is a *prefix* of execution fragment α', denoted by $\alpha \leq \alpha'$, if sequence α is a prefix of sequence α'. Finite execution fragment $\alpha_1 = q_0 a_1 q_1 \cdots a_k q_k$ and execution fragment α_2 can be concatenated if $fstate(\alpha_2) = q_k$. In this case the *concatenation* of α_1 and α_2, $\alpha_1 \frown \alpha_2$, is the execution fragment $q_0 a_1 q_1 \cdots a_k \alpha_2$. Given an execution fragment α and a finite prefix α', $\alpha \triangleright \alpha'$ (read α after α') is defined to be the unique execution fragment α'' such that $\alpha = \alpha' \frown \alpha''$.

The *trace* of an execution fragment α of an automaton $\mathcal{A}$, written $trace_{\mathcal{A}}(\alpha)$, or just $trace(\alpha)$ when $\mathcal{A}$ is clear from context, is the sequence obtained by restricting α to the set of external actions of A. For a set S of executions of $\mathcal{A}$, $traces_{\mathcal{A}}(S)$, or just $traces(S)$ when $\mathcal{A}$ is clear from context, is the set of traces of the executions in S. We say that β is a trace of $\mathcal{A}$ if there is an execution α of $\mathcal{A}$ with $trace(\alpha) = \beta$. Let $traces(\mathcal{A})$ denote the set of traces of $\mathcal{A}$. We define the *trace preorder* relation on automata as follows: $\mathcal{A}_1 \leq_T \mathcal{A}_2$ iff $E_1 = E_2$ and $traces(\mathcal{A}_1) \subseteq traces(\mathcal{A}_2)$. We use $\equiv_T$ to denote the kernel of $\leq_T$.

If $a \in A$, then $q \stackrel{a}{\Longrightarrow} q'$ iff there exists an execution fragment α such that $fstate(\alpha) = q$, $lstate(\alpha) = q'$, and $trace(\alpha) = trace(a)$. (Here and elsewhere, we abuse notation slightly by extending the *trace* function to arbitrary sequences.) We call $q \stackrel{a}{\Longrightarrow} q'$ a *weak transition*. We let tr range over either transitions or weak transitions. For a transition $tr = (q, a, q')$, we denote q by $source(tr)$ and q' by $target(tr)$.

Composition: Automata $\mathcal{A}_1$ and $\mathcal{A}_2$ are *compatible* if $H_1 \cap A_2 = A_1 \cap H_2 = \emptyset$. The *composition* of compatible automata $\mathcal{A}_1$ and $\mathcal{A}_2$, denoted by $\mathcal{A}_1 \| \mathcal{A}_2$, is the automaton $\mathcal{A} \stackrel{\Delta}{=} (Q_1 \times Q_2, (\bar{q}_1, \bar{q}_2), E_1 \cup E_2, H_1 \cup H_2, D)$ where D is the set of triples (q, a, q') such that, for $i \in \{1, 2\}$:

$$a \in A_i \Rightarrow (\pi_i(q), a, \pi_i(q')) \in D_i \text{ and } a \notin A_i \Rightarrow \pi_i(q) = \pi_i(q').$$

Let α be an execution fragment of $\mathcal{A}_1 \| \mathcal{A}_2$, $i \in \{1, 2\}$. Then $\pi_i(\alpha)$, the i^{th} projection of α, is the sequence obtained from α by projecting each state onto its i^{th} component, and removing each action not in A_i together with its following state. Sometimes we denote this projection by $\alpha \lceil \mathcal{A}_i$.

Proposition 1. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be automata, with $\mathcal{A}_1 \leq_T \mathcal{A}_2$. Then, for each automaton $\mathcal{C}$ compatible with both $\mathcal{A}_1$ and $\mathcal{A}_2$, $\mathcal{A}_1 \| \mathcal{C} \leq_T \mathcal{A}_2 \| \mathcal{C}$.*

Simulation relations: We define two kinds of simulation relations: a forward simulation, which provides a step-by-step correspondence, and a weak forward simulation, which is insensitive to the occurrence of internal steps.

Namely, relation $R \subseteq Q_1 \times Q_2$ is a *forward simulation* (resp., *weak forward simulation*) from $\mathcal{A}_1$ to $\mathcal{A}_2$ iff $E_1 = E_2$ and both of the following hold:

1. $\bar{q}_1 \; R \; \bar{q}_2$.
2. If $q_1 \; R \; q_2$ and $q_1 \xrightarrow{a} q_1'$, then there exists q_2' such that $q_2 \xrightarrow{a} q_2'$ (resp., $q_2 \stackrel{a}{\Longrightarrow} q_2'$) and $q_1' \; R \; q_2'$.

We write $\mathcal{A}_1 \leq_F \mathcal{A}_2$ (resp., $\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$) when there is a forward simulation (resp., a weak forward simulation) from $\mathcal{A}_1$ to $\mathcal{A}_2$.

Proposition 2. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be automata. Then:*

1. *If $\mathcal{A}_1 \leq_F \mathcal{A}_2$ then $\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$.*
2. *If $H_1 = H_2 = \emptyset$, then $\mathcal{A}_1 \leq_F \mathcal{A}_2$ iff $\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$.*
3. *If $\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$ then $\mathcal{A}_1 \leq_T \mathcal{A}_2$.*

Proof. Standard; for instance, see [6].

Tree-structured automata: An automaton is *tree-structured* if each state is reached via a unique execution. The *unfolding* of automaton $\mathcal{A}$, denoted by $\mathit{Unfold}(\mathcal{A})$, is the tree-structured automaton $\mathcal{B}$ obtained from $\mathcal{A}$ by unfolding its transition graph into a tree. Formally, $Q_{\mathcal{B}} = \mathit{execs}^*(\mathcal{A})$, $\bar{q}_{\mathcal{B}} = \bar{q}_{\mathcal{A}}$, $E_{\mathcal{B}} = E_{\mathcal{A}}$, $H_{\mathcal{B}} = H_{\mathcal{A}}$, and $D_{\mathcal{B}} = \{(\alpha, a, \alpha a q) \mid (\mathit{lstate}(\alpha), a, q) \in D_{\mathcal{A}}\}$.

Proposition 3. $\mathcal{A} \equiv_F \mathit{Unfold}(\mathcal{A})$.

Proof. See [6]. It is easy to check that the relation R, where $\alpha \; R \; q$ iff $\mathit{lstate}(\alpha) = q$, is a forward simulation from $\mathit{Unfold}(\mathcal{A})$ to $\mathcal{A}$ and that the inverse relation of R is a forward simulation from $\mathcal{A}$ to $\mathit{Unfold}(\mathcal{A})$.

Proposition 4. $\mathcal{A} \equiv_T \mathit{Unfold}(\mathcal{A})$.

Proof. By Proposition 3 and Proposition 2, Parts 1 and 3.

3 Definitions for Probabilistic Automata

A *discrete probability measure* over a set X is a measure μ on $(X, 2^X)$ such that $\mu(X) = 1$. A *discrete sub-probability measure* over X is a measure μ on $(X, 2^X)$ such that $\mu(X) \leq 1$. We denote the set of discrete probability measures and discrete sub-probability measures over X by $\mathit{Disc}(X)$ and $\mathit{SubDisc}(X)$, respectively. We denote the support of a discrete measure μ, i.e., the set of elements that have non-zero measure, by $\mathit{supp}(\mu)$. We let $\delta(q)$ denote the *Dirac measure* for q, the discrete probability measure that assigns probability 1 to $\{q\}$. Finally, if X is finite, then $\mathcal{U}(X)$ denotes the *uniform distribution* over X, the measure that assigns probability $1/|X|$ to each element of X.

A *probabilistic automaton (PA)* is a tuple $\mathcal{P} = (Q, \bar{q}, E, H, D)$, where all components are exactly as for nondeterministic automata, except that D, the *transition relation*, is a subset of $Q \times (E \cup H) \times \mathit{Disc}(Q)$. We define A as before.

We denote transition (q, a, μ) by $q \xrightarrow{a} \mu$. We assume finite branching: for each state q the number of pairs (a, μ) such that $q \xrightarrow{a} \mu$ is finite. Given a transition $tr = (q, a, \mu)$ we denote q by $source(tr)$ and μ by $target(tr)$.

Thus, a probabilistic automaton differs from a nondeterministic automaton in that a transition leads to a probability measure over states rather than to a single state. A nondeterministic automaton is a special case of a probabilistic automaton, where the last component of each transition is a Dirac measure. Conversely, we can associate a nondeterministic automaton with each probabilistic automaton by replacing transition relation D by the relation D' given by

$$(q, a, q') \in D' \Leftrightarrow \exists \mu : (q, a, \mu) \in D \wedge \mu(q') > 0.$$

Using this correspondence, notions such as execution fragments and traces carry over from nondeterministic automata to probabilistic automata.

A *scheduler* for a PA $\mathcal{P}$ is a function $\sigma : frags^*(\mathcal{P}) \rightarrow SubDisc(D)$ such that $tr \in supp(\sigma(\alpha))$ implies $source(tr) = lstate(\alpha)$. A scheduler σ is said to be *deterministic* if for each finite execution fragment α, either $\sigma(\alpha)(D) = 0$ or else $\sigma(\alpha) = \delta(tr)$ for some $tr \in D$.

A scheduler σ and a state q_0 induce a measure μ on the σ-field generated by cones of execution fragments as follows. If $\alpha = q_0 a_1 q_1 \cdots a_k q_k$ is a finite execution fragment, then the *cone* of α is defined by $C_\alpha = \{\alpha' \in frags(\mathcal{P}) \mid \alpha \leq \alpha'\}$, and the measure of C_α is defined by

$$\mu(C_\alpha) = \prod_{i \in \{0,k-1\}} \left(\sum_{(q_i, a_{i+1}, \mu') \in D} \sigma(q_0 a_1 \cdots a_i q_i)((q_i, a_{i+1}, \mu'))\mu'(q_{i+1}) \right).$$

Standard measure theoretical arguments ensure that μ is well defined. We call the measure μ a *probabilistic execution fragment* of $\mathcal{P}$ and we say that μ is *generated* by σ and q_0. We call state q_0 the *first state* of μ and denote it by $fstate(\mu)$. If $fstate(\mu)$ is the start state $\bar{q}$, then μ is called a *probabilistic execution*.

The trace function is a measurable function from the σ-field generated by cones of execution fragments to the σ-field generated by cones of traces. Given a probabilistic execution fragment μ, we define the *trace distribution* of μ, $tdist(\mu)$, to be the image measure of μ under *trace*. We denote the set of trace distributions of probabilistic executions of a PA $\mathcal{P}$ by $tdists(\mathcal{P})$. We define the *trace distribution preorder* relation on probabilistic automata by: $\mathcal{P}_1 \leq_D \mathcal{P}_2$ iff $E_1 = E_2$ and $tdists(\mathcal{P}_1) \subseteq tdists(\mathcal{P}_2)$.

Combined transitions: Let $\{q \xrightarrow{a} \mu_i\}_{i \in I}$ be a collection of transitions of $\mathcal{P}$, and let $\{p_i\}_{i \in I}$ be a collection of probabilities such that $\sum_{i \in I} p_i = 1$. Then the triple $(q, a, \sum_{i \in I} p_i \mu_i)$ is called a *combined transition* of $\mathcal{P}$.

Consider a probabilistic execution fragment μ that assigns probability 1 to the set of all finite execution fragments with trace a. Let μ' be the measure defined by $\mu'(q) = \mu(\{\alpha \mid lstate(\alpha) = q\})$. Then $fstate(\mu) \stackrel{a}{\Longrightarrow} \mu'$ is a *weak combined transition* of $\mathcal{P}$. If μ can be generated by a deterministic scheduler, then $fstate(\mu) \stackrel{a}{\Longrightarrow} \mu'$ is a *weak transition*.

Proposition 5. *Let $\{tr_i\}_{i\in I}$ be a collection of weak combined transitions of a PA $\mathcal{P}$, all starting in the same state q, and all labeled by the same action a, and let $\{p_i\}_{i\in I}$ be probabilities such that $\sum_{i\in I} p_i = 1$. Then $\sum_{i\in I} p_i tr_i$ is a weak combined transition of $\mathcal{P}$ labeled by a.*

Proof. See [9] or [11].

Composition: Two PAs, $\mathcal{P}_1$ and $\mathcal{P}_2$, are *compatible* if $H_1 \cap A_2 = A_1 \cap H_2 = \emptyset$. The *composition* of two compatible PAs $\mathcal{P}_1, \mathcal{P}_2$, denoted by $\mathcal{P}_1 \| \mathcal{P}_2$, is the PA $\mathcal{P} = (Q_1 \times Q_2, (\bar{q}_1, \bar{q}_2), E_1 \cup E_2, H_1 \cup H_2, D)$ where D is the set of triples $(q, a, \mu_1 \times \mu_2)$ such that, for $i \in \{1, 2\}$:

$$a \in A_i \Rightarrow (\pi_i(q), a, \mu_i) \in D_i \text{ and } a \notin A_i \Rightarrow \mu_i = \delta(\pi_i(q)).$$

The trace distribution preorder is not preserved by composition [10, 11]. Thus, we define the *trace distribution precongruence*, $\leq_{DC}$, to be the coarsest precongruence included in the trace distribution preorder $\leq_D$. This relation has a simple characterization:

Proposition 6. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be PAs. Then $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$ iff for every PA $\mathcal{C}$ that is compatible with both $\mathcal{P}_1$ and $\mathcal{P}_2$, $\mathcal{P}_1 \| \mathcal{C} \leq_D \mathcal{P}_2 \| \mathcal{C}$.*

Simulation relations: The definitions of forward simulation and weak forward simulation in Section 2 can be extended naturally to PAs [10]. However, Segala has shown [8] that the resulting simulations are not complete for $\leq_{DC}$, and has defined new candidate simulations. These new simulations relate states to probability distributions on states.

In order to define formally the new simulations we need three new concepts. First we show how to lift a relation between sets to a relation between distributions over sets [2]. Let $R \subseteq X \times Y$. The *lifting* of R is a relation $R' \subseteq Disc(X) \times Disc(Y)$ such that $\mu_X \; R' \; \mu_Y$ iff there is a function $w : X \times Y \to [0, 1]$ that satisfies:

1. If $w(x, y) > 0$ then $x \; R \; y$.
2. For each $x \in X$, $\sum_{y\in Y} w(x, y) = \mu_X(x)$.
3. For each $y \in Y$, $\sum_{x\in X} w(x, y) = \mu_Y(y)$.

We abuse notation and denote the lifting of a relation R by R as well.

Next we define a flattening operation that converts a measure μ contained in $Disc(Disc(X))$ into a measure $flatten(\mu)$ in $Disc(X)$. Namely, we define

$$flatten(\mu) = \sum_{\rho\in supp(\mu)} \mu(\rho)\rho \; .$$

Finally, we lift the notion of a transition to a *hyper-transition* [11] that begins and ends with a probability distributions over states. Thus, let $\mathcal{P}$ be a PA and let $\mu \in Disc(Q)$. For each $q \in supp(\mu)$, let $q \xrightarrow{a} \mu_q$ be a combined transition of $\mathcal{P}$. Let μ' be $\sum_{q\in supp(\mu)} \mu(q)\mu_q$. Then $\mu \xrightarrow{a} \mu'$ is called a *hyper-transition* of $\mathcal{P}$.

Also, for each $q \in supp(\mu)$, let $q \stackrel{a}{\Longrightarrow} \mu_q$ be a weak combined transition of $\mathcal{P}$. Let μ' be $\sum_{q \in supp(\mu)} \mu(q)\mu_q$. Then $\mu \stackrel{a}{\Longrightarrow} \mu'$ is called a *weak hyper-transition* of $\mathcal{P}$.

We now define simulations for probabilistic automata. A relation $R \subseteq Q_1 \times Disc(Q_2)$ is a *probabilistic forward simulation* (resp., *weak probabilistic forward simulation*) from PA $\mathcal{P}_1$ to PA $\mathcal{P}_2$ iff $E_1 = E_2$ and both of the following hold:

1. $\bar{q}_1 \; R \; \delta(\bar{q}_2)$.
2. For each pair q_1, μ_2 such that $q_1 \; R \; \mu_2$ and each transition $q_1 \stackrel{a}{\rightarrow} \mu_1'$ there exists a distribution $\mu_2' \in Disc(Disc(Q_2))$ such that $\mu_1' \; R \; \mu_2'$ and such that $\mu_2 \stackrel{a}{\rightarrow} flatten(\mu_2')$ (resp., $\mu_2 \stackrel{a}{\Longrightarrow} flatten(\mu_2')$) is a hyper-transition (resp., a weak hyper-transition) of D_2.

We write $\mathcal{P}_1 \leq_{PF} \mathcal{P}_2$ (resp., $\mathcal{P}_1 \leq_{wPF} \mathcal{P}_2$) whenever there is a probabilistic forward simulation (resp., a weak probabilistic forward simulation) from $\mathcal{P}_1$ to $\mathcal{P}_2$. Note that a forward simulation between nondeterministic automata is a probabilistic forward simulation between the two automata viewed as PAs:

Proposition 7. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be nondeterministic automata. Then:*

1. *$\mathcal{A}_1 \leq_F \mathcal{A}_2$ implies $\mathcal{A}_1 \leq_{PF} \mathcal{A}_2$, and*
2. *$\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$ implies $\mathcal{A}_1 \leq_{wPF} \mathcal{A}_2$.*

Proposition 8. *Let $\mathcal{P}_1$ and $\mathcal{P}_2$ be PAs. Then:*

1. *If $\mathcal{P}_1 \leq_{PF} \mathcal{P}_2$ then $\mathcal{P}_1 \leq_{wPF} \mathcal{P}_2$.*
2. *If $H_1 = H_2 = \emptyset$ then $\mathcal{P}_1 \leq_{PF} \mathcal{P}_2$ iff $\mathcal{P}_1 \leq_{wPF} \mathcal{P}_2$.*
3. *If $\mathcal{P}_1 \leq_{wPF} \mathcal{P}_2$ then $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$.*

Proof. See [9].

Tree-structured probabilistic automata: The *unfolding* of a probabilistic automaton $\mathcal{P}$, denoted by $Unfold(\mathcal{P})$, is the tree-structured probabilistic automaton $\mathcal{Q}$ obtained from $\mathcal{P}$ by unfolding its transition graph into a tree. Formally, $Q_{\mathcal{Q}} = execs^*(\mathcal{P})$, $\bar{q}_{\mathcal{Q}} = \bar{q}_{\mathcal{P}}$, $E_{\mathcal{Q}} = E_{\mathcal{P}}$, $H_{\mathcal{Q}} = H_{\mathcal{P}}$, and $D_{\mathcal{Q}} = \{(\alpha, a, \mu) \mid \exists_{\mu'}(lstate(\alpha), a, \mu') \in D_{\mathcal{P}}, \forall_q \mu'(q) = \mu(\alpha a q)\}$.

Proposition 9. *$\mathcal{P} \equiv_{PF} Unfold(\mathcal{P})$.*

Proof. It is easy to check that the relation R where $\alpha \; R \; \delta(q)$ iff $lstate(\alpha) = q$ is a probabilistic forward simulation from $Unfold(\mathcal{P})$ to $\mathcal{P}$ and that the "inverse" of R is a probabilistic forward simulation from $\mathcal{P}$ to $Unfold(\mathcal{P})$.

Proposition 10. *$\mathcal{P} \equiv_{DC} Unfold(\mathcal{P})$.*

Proof. By Proposition 9, and Proposition 8, Parts 1 and 3.

4 Characterizations of $\leq_{DC}$: Nondeterministic Automata

In this section, we prove our characterization theorems for $\leq_{DC}$ for nondeterministic automata: Theorem 1 characterizes $\leq_{DC}$ in terms of $\leq_F$, for automata without internal actions, and Theorem 2 characterizes $\leq_{DC}$ in terms of $\leq_{wF}$, for arbitrary nondeterministic automata. In each case, we prove the result first for tree-structured automata and then extend it to the non-tree-structured case via unfolding. The interesting direction for these results is the completeness direction, showing that $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$ implies the existence of a simulation relation from $\mathcal{A}_1$ to $\mathcal{A}_2$.

Our proofs of completeness for nondeterministic automata use the simple characterization in Proposition 6, applied to a special context for $\mathcal{A}_1$ that we call the dual probabilistic automaton of $\mathcal{A}_1$. Informally speaking, the *dual probabilistic automaton* of a nondeterministic automaton $\mathcal{A}$ is a probabilistic automaton $\mathcal{C}$ whose traces contain information about states and transitions of $\mathcal{A}$. $\mathcal{C}$'s states and start state are the same as those of $\mathcal{A}$. For every state q of $\mathcal{A}$, $\mathcal{C}$ has a self-loop transition labeled by q. Also, if Tr is the (nonempty) set of transitions from q in $\mathcal{A}$, then from state q, $\mathcal{C}$ has a uniform transition labeled by ch to $\{target(tr) \mid tr \in Tr\}$.

Definition 1. *The* dual probabilistic automaton *of an automaton $\mathcal{A}$ is a PA $\mathcal{C}$ such that*

- $Q_{\mathcal{C}} = Q_{\mathcal{A}}$, $\bar{q}_{\mathcal{C}} = \bar{q}_{\mathcal{A}}$,
- $E_{\mathcal{C}} = Q_{\mathcal{A}} \cup \{ch\}$, $H_{\mathcal{C}} = \emptyset$,
- $D_{\mathcal{C}} = \{(q, ch, \mathcal{U}(\{q' \mid q \rightarrow_{\mathcal{A}} q'\})) \mid q \rightarrow_{\mathcal{A}}\} \cup \{(q, q, q) \mid q \in Q_{\mathcal{A}}\}$.

Since $\mathcal{C}$ and $\mathcal{A}$ share no actions, $\mathcal{C}$ cannot ensure that its traces faithfully emulate the behavior of $\mathcal{A}$. However, an appropriate scheduler can synchronize the two automata and ensure such an emulation.

4.1 Automata Without Internal Actions

We first consider tree-structured automata.

Proposition 11. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be tree-structured nondeterministic automata without internal actions, such that $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$. Then $\mathcal{A}_1 \leq_F \mathcal{A}_2$.*

Proof. Assume that $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$. Let $\mathcal{C}$ be the dual probabilistic automaton of $\mathcal{A}_1$. Without loss of generality, we assume that the set of actions of $\mathcal{C}$ is disjoint from those of $\mathcal{A}_1$ and $\mathcal{A}_2$. This implies that $\mathcal{C}$ is compatible with both $\mathcal{A}_1$ and $\mathcal{A}_2$.

Consider the scheduler σ_1 for $\mathcal{A}_1 \| \mathcal{C}$ that starts by scheduling the self-loop transition labelled by the start state of $\mathcal{C}$, leading to state $(\bar{q}_1, \bar{q}_1)$, which is of the form (q, q). Then σ_1 repeats the following as long as $q \rightarrow_1$:

1. Schedule the ch transition of $\mathcal{C}$, thus choosing a new state q' of $\mathcal{A}_1$.

2. Schedule (q, a, q') in $\mathcal{A}_1$, where a is uniquely determined by the selected state q' (recall that $\mathcal{A}_1$ is a tree).
3. Schedule the self-loop transition of $\mathcal{C}$ labeled by q', resulting in the state (q', q'), which is again of the form (q, q).

Scheduler σ_1 induces a trace distribution μ_T. Observe that μ_T satisfies the following three properties, for all finite traces β and for all states q:

$$\mu_T(C_{\bar{q}_1}) = 1 \tag{1}$$

$$q \rightarrow_1 \quad \Rightarrow \quad \mu_T(C_{\beta q c h}) = \mu_T(C_{\beta q}) \tag{2}$$

$$\mu_T(C_{\beta q c h}) > 0 \quad \Rightarrow \sum_{a, q' | q \xrightarrow{a}_1 q'} \mu_T(C_{\beta q c h a q'}) = \mu_T(C_{\beta q c h}) \tag{3}$$

Since $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$, Proposition 6 implies that μ_T is also a trace distribution of $\mathcal{A}_2 \| \mathcal{C}$. That is, there exists a probabilistic execution μ of $\mathcal{A}_2 \| \mathcal{C}$, induced by some scheduler σ_2, whose trace distribution is μ_T. Now we define a relation R: $q_1 \; R \; q_2$ if and only if there exists an execution α of $\mathcal{A}_2 \| \mathcal{C}$ such that:

1. $lstate(\alpha) = (q_2, q_1)$,
2. $\mu(C_\alpha) > 0$, and
3. $\sigma_2(\alpha)$ assigns a non-zero probability to a transition labeled by q_1.

We claim that R is a forward simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$. For the start condition, we must show that $\bar{q}_1 \; R \; \bar{q}_2$. Define execution α to be the trivial execution consisting of the start state $(\bar{q}_2, \bar{q}_1)$. Conditions 1 and 2 are clearly satisfied. For Condition 3, observe that, by Equation (1), $\mu_T(C_{\bar{q}_1}) = 1$. Therefore, since there are no internal actions in $\mathcal{A}_2$ or $\mathcal{C}$, the only action that can be scheduled initially by σ_2 is $\bar{q}_1$. Therefore, $\sigma_2(\alpha)$ assigns probability 1 to the unique transition whose label is $\bar{q}_1$, as needed.

For the step condition, assume $q_1 \; R \; q_2$, and let $q_1 \xrightarrow{a}_1 q_1'$. By definition of R, there exists a finite execution α of $\mathcal{A}_2 \| \mathcal{C}$, with last state (q_2, q_1), such that $\mu(C_\alpha) > 0$ and $\sigma_2(\alpha)$ assigns a non-zero probability to a transition labeled by q_1. Therefore, the sequence $\alpha' = \alpha q_1 (q_2, q_1)$ is an execution of $\mathcal{A}_2 \| \mathcal{C}$ such that $\mu(C_{\alpha'}) > 0$. Therefore, $\mu_T[C_{\beta q_1}] > 0$, where $\beta = trace(\alpha)$. Since q_1 enables at least one transition in $\mathcal{A}_1$, Equation (2) implies that $\mu_T(C_{\beta q_1 ch}) = \mu_T(C_{\beta q_1})$. Then since $\mathcal{A}_2$ and $\mathcal{C}$ have no internal actions, σ_2 must schedule action ch from α' with probability 1.

Since action ch leads to state q_1' of $\mathcal{C}$ with non-zero probability, which enables only actions q_1' and ch, by Equation (3), σ_2 schedules at least one transition labeled by a, followed by a transition labeled by q_1'. Observe that the transition labeled by a is a transition of $\mathcal{A}_2$. Let (q_2, a, q_2') be such a transition. Then, the sequence $\alpha'' = \alpha' ch (q_2, q_1') a (q_2', q_1')$ is an execution of $\mathcal{A}_2 \| \mathcal{C}$ such that $\mu(C_{\alpha''}) > 0$ and such that $\sigma_2(\alpha'')$ assigns a non-zero probability to a transition labeled by q_1'. This shows that $q_1' \; R \; q_2'$ and completes the proof since we have found a state q_2' such that $q_2 \xrightarrow{a}_2 q_2'$ and $q_1' \; R \; q_2'$.

Now we present our main result, for general (non-tree-structured) nondeterministic automata without internal actions.

Theorem 1. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be nondeterministic automata without internal actions. Then $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$ if and only if $\mathcal{A}_1 \leq_F \mathcal{A}_2$.*

Proof. First we prove soundness of forward simulations:

$$\begin{array}{ll} \mathcal{A}_1 \leq_F \mathcal{A}_2 & \Rightarrow \text{(Proposition 7, Part 1)} \\ \mathcal{A}_1 \leq_{PF} \mathcal{A}_2 & \Rightarrow \text{(Proposition 8, Part 1)} \\ \mathcal{A}_1 \leq_{wPF} \mathcal{A}_2 & \Rightarrow \text{(Proposition 8, Part 3)} \\ \mathcal{A}_1 \leq_{DC} \mathcal{A}_2 \ . & \end{array}$$

Completeness is established by:

$$\begin{array}{ll} \mathcal{A}_1 \leq_{DC} \mathcal{A}_2 & \Rightarrow \text{(Proposition 10)} \\ \mathit{Unfold}(\mathcal{A}_1) \leq_{DC} \mathcal{A}_1 \leq_{DC} \mathcal{A}_2 \leq_{DC} \mathit{Unfold}(\mathcal{A}_2) & \Rightarrow (\leq_{DC} \text{ is transitive}) \\ \mathit{Unfold}(\mathcal{A}_1) \leq_{DC} \mathit{Unfold}(\mathcal{A}_2) & \Rightarrow \text{(Proposition 11)} \\ \mathit{Unfold}(\mathcal{A}_1) \leq_F \mathit{Unfold}(\mathcal{A}_2) & \Rightarrow \text{(Proposition 3)} \\ \mathcal{A}_1 \leq_F \mathit{Unfold}(\mathcal{A}_1) \leq_F \mathit{Unfold}(\mathcal{A}_2) \leq_F \mathcal{A}_2 & \Rightarrow (\leq_F \text{ is transitive}) \\ \mathcal{A}_1 \leq_F \mathcal{A}_2 \ . & \end{array}$$

4.2 Automata With Internal Actions

Next we extend the results of Section 4.1 to automata that include internal actions. The proofs are analogous to those in Section 4.1, and use the same dual probabilistic automaton. The difference is that, in several places in the proof of Proposition 12, we need to reason about multi-step extensions of executions instead of single-step extensions. Again, we begin with tree-structured automata.

Proposition 12. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be tree-structured nondeterministic automata such that $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$. Then $\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$.*

Proof. Assume that $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$. Let $\mathcal{C}$ be the dual probabilistic automaton of $\mathcal{A}_1$, and define scheduler σ_1 exactly as in the proof of Proposition 11. Equations (1), (2) and (3) hold in this case as well. We redefine relation R: $q_1 \ R \ q_2$ iff there exists an execution α of $\mathcal{A}_2 \| \mathcal{C}$ such that:

1. $\mathit{lstate}(\alpha) = (q_2, q_1)$,
2. $\mu(C_\alpha) > 0$, and
3. there exists an execution fragment, α', of $\mathcal{A}_2 \| \mathcal{C}$, such that $\mathit{trace}(\alpha') = q_1$ and $\mu(C_{\alpha^\frown \alpha'}) > 0$.

We claim that R is a weak forward simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$. For the start condition, we show that $\bar{q}_1 \ R \ \bar{q}_2$. Define α to be the trivial execution consisting of the start state $(\bar{q}_2, \bar{q}_1)$; this clearly satisfies Conditions 1 and 2. For Condition 3, observe that, by Equation (1), $\mu_T(C_{\bar{q}_1}) = 1$. The inverse image under the trace mapping for $\mathcal{A}_2 \| \mathcal{C}$, of $C_{\bar{q}_1}$, is a union of cones of the form $C_{\alpha'}$, where α' is

an execution of $\mathcal{A}_2 \| \mathcal{C}$ with trace $\bar{q}_1$; therefore, there exists such an α' with $\mu(C_{\alpha'}) > 0$. Since the first state of α' is $(\bar{q}_2, \bar{q}_1)$, $\alpha \frown \alpha' = \alpha'$. Thus, $\mu(C_{\alpha \frown \alpha'}) > 0$, as needed.

For the step condition, assume $q_1 \ R \ q_2$, and let $q_1 \xrightarrow{a}_1 q_1'$. By definition of R, there exists a finite execution α of $\mathcal{A}_2 \| \mathcal{C}$, with last state (q_2, q_1), such that $\mu(C_\alpha) > 0$ and there exists an execution fragment, α', of $\mathcal{A}_2 \| \mathcal{C}$, such that $trace(\alpha') = q_1$ and $\mu(C_{\alpha \frown \alpha'}) > 0$. Let $\beta = trace(\alpha)$; then $trace(\alpha \frown \alpha') = \beta q_1$, and so $\mu_T(C_{\beta q_1}) > 0$. Since q_1 enables at least one transition in $\mathcal{A}_1$, Equation (2) implies that $\mu_T(C_{\beta q_1 ch}) = \mu_T(C_{\beta q_1})$. Thus there exists an execution fragment α'' of $\mathcal{A}_2 \| \mathcal{C}$ with trace ch such that $\mu(C_{\alpha \frown \alpha' \frown \alpha''}) > 0$. Furthermore, since the transition of $\mathcal{C}$ labeled by ch leads to state q_1' with non-zero probability, we can assume that the last state of α'' is of the form (q'', q_1') for some state q''.

Since $\mu_T(C_{\beta q_1 ch}) > 0$, Equation (3) applies. Furthermore, since from the last state of α'' the only external actions of $\mathcal{C}$ that are enabled are ch and q_1', there exists an execution fragment α''' with trace aq_1' (a is uniquely determined by q_1' since $\mathcal{A}_1$ is tree-structured), such that $\mu(C_{\alpha \frown \alpha' \frown \alpha'' \frown \alpha'''}) > 0$.

Now we split α''' into $\alpha_1''' \frown \alpha_2'''$, where $trace(\alpha_1''') = a$. Then the last state of α_1''' is of the form (q''', q_1'). We claim that $q_1' \ R \ q'''$. Indeed, the execution $\alpha \frown \alpha' \frown \alpha'' \frown \alpha_1'''$ ends with state (q''', q_1') (Condition 1) and satisfies $\mu(C_{\alpha \frown \alpha' \frown \alpha'' \frown \alpha_1'''}) > 0$ (Condition 2). Furthermore, α_2''' is an execution fragment that satisfies Condition 3.

It remains to show that $q_2 \stackrel{a}{\Longrightarrow} q'''$. For this, it suffices to observe that the execution fragment $(\alpha' \frown \alpha'' \frown \alpha_1''') \lceil \mathcal{A}_2$ has trace a, first state q_2, and last state q'''.

Theorem 2. *Let $\mathcal{A}_1$, $\mathcal{A}_2$ be nondeterministic automata. Then $\mathcal{A}_1 \leq_{DC} \mathcal{A}_2$ if and only if $\mathcal{A}_1 \leq_{wF} \mathcal{A}_2$.*

Proof. Analogous to the proof of Theorem 1.

5 Characterizations of $\leq_{DC}$: Probabilistic Automata

Finally, we present our characterization theorems for $\leq_{DC}$ for probabilistic automata: Theorem 3 characterizes $\leq_{DC}$ in terms of $\leq_{PF}$, for PAs without internal actions, and Theorem 4 characterizes $\leq_{DC}$ in terms of $\leq_{wPF}$, for arbitrary PAs. Again, we give the results first for tree-structured automata and extend them by unfolding.

Our proofs of completeness for PAs are analogous to those for nondeterministic automata. We define a new kind of *dual probabilistic automaton* $\mathcal{C}$ for a PA $\mathcal{P}$, which is slightly different from the one for nondeterministic automata. The main differences are that the new $\mathcal{C}$ keeps track, in its state, of transitions as well as states of the given PA $\mathcal{P}$, and that the new $\mathcal{C}$ has separate transitions representing nondeterministic and probabilistic choices within $\mathcal{P}$. Specifically, the states of $\mathcal{C}$ include a distinguished start state, all the states of $\mathcal{P}$, and all the transitions of $\mathcal{P}$. $\mathcal{C}$ has a special transition from its own start state $\bar{q}_\mathcal{C}$ to the start state of $\mathcal{P}$, $\bar{q}_\mathcal{P}$, labeled by $\bar{q}_\mathcal{P}$. Also, from every state q of $\mathcal{P}$, $\mathcal{C}$ has

a uniform transition labeled by ch to the set of transitions of $\mathcal{P}$ that start in state q. Finally, for every transition tr of $\mathcal{P}$, and every state q in the support of $target(tr)$, $\mathcal{C}$ has a transition labeled by q from tr to q.

Definition 2. *The* dual probabilistic automaton *of a PA $\mathcal{P}$ is a PA $\mathcal{C}$ such that*

- $Q_{\mathcal{C}} = \{\bar{q}_{\mathcal{C}}\} \cup Q_{\mathcal{P}} \cup D_{\mathcal{P}}$,
- $E_{\mathcal{C}} = Q_{\mathcal{P}} \cup \{ch\}$, $H_{\mathcal{C}} = \emptyset$,
- $D_{\mathcal{C}} = \{(\bar{q}_{\mathcal{C}}, \bar{q}_{\mathcal{P}}, \bar{q}_{\mathcal{P}})\} \cup$
  $\{(q, ch, \mathcal{U}(\{tr \in D_{\mathcal{P}} \mid source(tr) = q\})) \mid q \in Q_{\mathcal{P}}\} \cup$
  $\{(tr, q, q) \mid tr \in D_{\mathcal{P}}, q \in supp(target(tr))\}$.

Proposition 13. *Let $\mathcal{P}_1$, $\mathcal{P}_2$ be tree-structured probabilistic automata without internal actions, such that $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$. Then $\mathcal{P}_1 \leq_{PF} \mathcal{P}_2$.*

Proof. (Sketch:) Assume that $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$. Let $\mathcal{C}$ be the dual probabilistic automaton of $\mathcal{P}_1$. Consider the scheduler σ_1 for $\mathcal{P}_1 \| \mathcal{C}$ that starts by scheduling the transition of $\mathcal{C}$ from the start state of $\mathcal{C}$ to the start state of $\mathcal{P}_1$, leading to state $(\bar{q}_1, \bar{q}_1)$, which is of the form (q, q). Then σ_1 repeats the following as long as $q \rightarrow_1$:

1. Schedule the ch transition of $\mathcal{C}$, thus choosing a transition tr of $\mathcal{P}_1$.
2. Schedule transition tr of $\mathcal{P}_1$, leading $\mathcal{P}_1$ to a new state q'.
3. Schedule the transition of $\mathcal{C}$ labeled by the state q', resulting in the state (q', q'), which is again of the form (q, q).

Scheduler σ_1 induces a trace distribution μ_T. Since $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$, Proposition 6 implies that μ_T is also a trace distribution of $\mathcal{P}_2 \| \mathcal{C}$. That is, there exists a probabilistic execution μ of $\mathcal{P}_2 \| \mathcal{C}$, induced by some scheduler σ_2, whose trace distribution is μ_T.

For each state q_1 in Q_1, let Θ_{q_1} be the set of finite executions of $\mathcal{A}_2 \| \mathcal{C}$ whose last transition is labeled by q_1. For each state q_2 of $\mathcal{P}_2$, let Θ_{q_1,q_2} be the set of finite executions in Θ_{q_1} whose last state is the pair (q_2, q_1). Now define relation R: $q_1 \; R \; \mu_2$ iff for each state q_2 of Q_2,

$$\mu_2(q_2) = \frac{\sum_{\alpha \in \Theta_{q_1,q_2}} \mu(C_\alpha)}{\sum_{\alpha \in \Theta_{q_1}} \mu(C_\alpha)} \; . \tag{4}$$

We claim that R is a probabilistic forward simulation from $\mathcal{P}_1$ to $\mathcal{P}_2$. The proof of this claim appears in [4].

Theorem 3. *Let $\mathcal{P}_1$, $\mathcal{P}_2$ be probabilistic automata without internal actions. Then $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$ if and only if $\mathcal{P}_1 \leq_{PF} \mathcal{P}_2$.*

Proposition 14. *Let $\mathcal{P}_1$, $\mathcal{P}_2$ be tree-structured probabilistic automata such that $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$. Then $\mathcal{P}_1 \leq_{wPF} \mathcal{P}_2$.*

Proof. (Sketch:) We use the same dual automaton $\mathcal{C}$. Define scheduler σ_1 and relation R exactly as in the proof of Proposition 13. Now R is a weak probabilistic forward simulation, as shown in [4].

Theorem 4. *Let $\mathcal{P}_1$, $\mathcal{P}_2$ be probabilistic automata. Then $\mathcal{P}_1 \leq_{DC} \mathcal{P}_2$ if and only if $\mathcal{P}_1 \leq_{wPF} \mathcal{P}_2$.*

6 Concluding Remarks

We have characterized the trace distribution precongruence for nondeterministic and probabilistic automata, with and without internal actions, in terms of four kinds of simulation relations, $\leq_F$, $\leq_{wF}$, $\leq_{PF}$, and $\leq_{wPF}$. In particular, this shows that probabilistic contexts are capable of observing all the distinctions that can be expressed using these simulation relations. Some technical improvements are possible. For example, our finite branching restriction can be relaxed to countable branching, simply by replacing uniform distributions in the dual automata by other distributions such as exponential distributions.

For future work, it would be interesting to try to restrict the class of schedulers used for defining the trace distribution precongruence, so that fewer distinctions are observable by probabilistic contexts. It remains to define such restrictions and to provide explicit chacterizations of the resulting new notions of $\leq_{DC}$, for instance in terms of button pushing scenarios.

References

1. S. Aggarwal. Time optimal self-stabilizing spanning tree algorithms. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, May 1994. Available as Technical Report MIT/LCS/TR-632.
2. K.G. Larsen B. Jonsson. Specification and refinement of probabilistic processes. In *Proceedings* 6^{th} *Annual Symposium on Logic in Computer Science,* Amsterdam, pages 266–277. IEEE Press, 1991.
3. N.A. Lynch, I. Saias, and R. Segala. Proving time bounds for randomized distributed algorithms. In *Proceedings of the 13th Annual ACM Symposium on the Principles of Distributed Computing*, pages 314–323, Los Angeles, CA, August 1994.
4. N.A. Lynch, R. Segala, and F.W. Vaandrager. Compositionality for probabilistic automata. Technical Report MIT-LCS-TR-907, MIT, Laboratory for Computer Science, 2003.
5. N.A. Lynch and M.R. Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2(3):219–246, September 1989.
6. N.A. Lynch and F.W. Vaandrager. Forward and backward simulations, I: Untimed systems. *Information and Computation*, 121(2):214–233, September 1995.
7. A. Pogosyants, R. Segala, and N.A. Lynch. Verification of the randomized consensus algorithm of Aspnes and Herlihy: a case study. *Distributed Computing*, 13(3):155–186, 2000.
8. R. Segala. Compositional trace–based semantics for probabilistic automata. In *Proc. CONCUR'95*, volume 962 of *Lecture Notes in Computer Science*, pages 234–248, 1995.
9. R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, June 1995. Available as Technical Report MIT/LCS/TR-676.
10. R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995.

11. M.I.A. Stoelinga. *Alea jacta est: Verification of Probabilistic, Real-Time and Parametric Systems.* PhD thesis, University of Nijmegen, April 2002.
12. M.I.A. Stoelinga. An introduction to probabilistic automata. *Bulletin of the European Association for Theoretical Computer Science*, 78:176–198, October 2002.
13. M.I.A. Stoelinga and F.W. Vaandrager. Root contention in IEEE 1394. In J.-P. Katoen, editor, *Proceedings 5th International AMAST Workshop on Formal Methods for Real-Time and Probabilistic Systems,* Bamberg, Germany, volume 1601 of *Lecture Notes in Computer Science*, pages 53–74. Springer-Verlag, 1999.
14. M.I.A. Stoelinga and F.W. Vaandrager. A testing scenario for probabilistic automata. In J.C.M. Baeten, J.K. Lenstra, J. Parrow, and G.J. Woeginger, editors, *Proceedings* 30^{th} *ICALP*, volume 2719 of *Lecture Notes in Computer Science*, pages 407–418. Springer-Verlag, 2003.