Compositionality for Probabilistic Automata

Nancy Lynch	extsuperscript{1},*, Roberto Segala	extsuperscript{2,**}, and Frits Vaandrager	extsuperscript{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 mod-
eling and analyzing probabilistic systems, specifically, systems consisting of
asynchronously interacting components that may make nondeterministic and
probabilistic choices. They have been applied successfully to distributed algo-
rithms [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 \( q \xrightarrow{a} q' \) if \( q \xrightarrow{a} q' \) for some \( a \), and we write \( q \xrightarrow{a} q' \) if \( q \xrightarrow{a} q' \) 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 \( Q_A, \bar{q}_A, E_A, H_A, D_A, A_A, \xrightarrow{a}_A \). Often we use the name \( \mathcal{A} \) for a generic automaton; then we usually omit the subscripts, writing simply \( Q, \bar{q}, E, H, D, A, \xrightarrow{a} \). We extend this convention to allow indices and primes as well: thus, the set of states of automaton \( \mathcal{A}' \) 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 \(\alpha\), is denoted by \(fstate(\alpha)\). If \(\alpha\) is a finite sequence, then the last state of \(\alpha\) is denoted by \(lstate(\alpha)\). An execution is an execution fragment whose first state is the start state \(q\). We let \(frags(A)\) denote the set of execution fragments of \(A\) and \(frags^*(A)\) the set of finite execution fragments. Similarly, we let \(execs(A)\) denote the set of executions of \(A\) and \(execs^*(A)\) the set of finite executions.

Execution fragment \(\alpha\) is a prefix of execution fragment \(\alpha'\), denoted by \(\alpha \leq \alpha'\), if sequence \(\alpha\) is a prefix of sequence \(\alpha'\). Finite execution fragment \(\alpha_1 = q_0 a_1 q_1 \cdots a_k q_k\) and execution fragment \(\alpha_2\) can be concatenated if \(fstate(\alpha_2) = q_k\). In this case the concatenation of \(\alpha_1\) and \(\alpha_2\), \(\alpha_1 \alpha_2\), is the execution fragment \(q_0 a_1 q_1 \cdots a_k a_2\). Given an execution fragment \(\alpha\) and a finite prefix \(\alpha'\), \(\alpha \triangleright \alpha'\) (read \(\alpha\) after \(\alpha'\)) is defined to be the unique execution fragment \(\alpha''\) such that \(\alpha = \alpha' \triangleright \alpha''\).

The trace of an execution fragment \(\alpha\) of an automaton \(A\), written \(trace_A(\alpha)\), or just \(trace(\alpha)\) when \(A\) is clear from context, is the sequence obtained by restricting \(\alpha\) to the set of external actions of \(A\). For a set \(S\) of executions of \(A\), \(traces_A(S)\), or just \(traces(S)\) when \(A\) is clear from context, is the set of traces of the executions in \(S\). We say that \(\beta\) is a trace of \(\alpha\) if there is an execution \(\alpha\) of \(A\) with \(trace(\alpha) = \beta\). Let \(traces(\alpha)\) denote the set of traces of \(\alpha\). We define the trace preorder relation on automatas as follows: \(A_1 \leq_T A_2\) if \(E_1 = E_2\) and \(traces(A_1) \subseteq traces(A_2)\). We use \(=\gamma\) to denote the kernel of \(\leq_T\).

If \(a \in A\), then \(q \rightarrow q'\) iff there exists an execution fragment \(\alpha\) 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 \(q \rightarrow q'\) 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 \(A_1\) and \(A_2\) are compatible if \(H_1 \cap A_2 = A_1 \cap H_2 = \emptyset\). The composition of compatible automata \(A_1\) and \(A_2\), denoted by \(A_1||A_2\), is the automaton \(A \triangleq (Q_1 \times Q_2, (q_0, q_{0'}), E_1 \cup E_2, H_1 \cup H_2, D)\) 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 \(\alpha\) be an execution fragment of \(A_1||A_2\), \(i \in \{1, 2\}\). Then \(\pi_i(\alpha)\), the \(i\)th projection of \(\alpha\), is the sequence obtained from \(\alpha\) 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 \(\alpha_i\).

**Proposition 1.** Let \(A_1\) and \(A_2\) be automata, with \(A_1 \leq_T A_2\). Then, for each automaton \(C\) compatible with both \(A_1\) and \(A_2\), \(A_1||C \leq_T A_2||C\).

Simulation Relations: Below 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 \( A_1 \) to \( A_2 \) iff \( E_1 = E_2 \) and both of the following hold:

1. \( \overline{q}_1 R \overline{q}_2 \).
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., \( q_2 \xrightarrow{\delta} q'_2 \)) and \( q'_1 R q'_2 \).

We write \( A_1 \leq_F A_2 \) (resp., \( A_1 \leq_w F A_2 \)) when there is a forward simulation (resp., a weak forward simulation) from \( A_1 \) to \( A_2 \).

**Proposition 2.** Let \( A_1 \) and \( A_2 \) be automata. Then:

1. If \( A_1 \leq_F A_2 \) then \( A_1 \leq_w F A_2 \).
2. If \( H_1 = H_2 = \emptyset \), then \( A_1 \leq_F A_2 \) iff \( A_1 \leq_w F A_2 \).
3. If \( A_1 \leq_w F A_2 \) then \( A_1 \leq_T A_2 \).

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

**Tree-Structured Automata:** An automaton is tree-structured if each state can be reached via a unique execution. The unfolding of automaton \( A \), denoted by \( \text{Unfold}(A) \), is the tree-structured automaton \( B \) obtained from \( A \) by unfolding its transition graph into a tree. Formally, \( Q_B = \exp*(A) \), \( \overline{q}_B = \overline{q}_A \), \( E_B = E_A \), \( H_B = H_A \), and \( D_B = \{ (\text{Istate}(\alpha), a, q) \mid (\text{Istate}(\alpha), a, q) \in D_A \} \).

**Proposition 3.** \( A \equiv_F \text{Unfold}(A) \).

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

**Proposition 4.** \( A \equiv_T \text{Unfold}(A) \).

**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 \( \mu \) on \( (X, 2^X) \) such that \( \mu(X) = 1 \). A discrete sub-probability measure over \( X \) is a measure \( \mu \) 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 \( \text{Disc}(X) \) and \( \text{SubDisc}(X) \), respectively. We denote the support of a discrete measure \( \mu \), i.e., the set of elements that have non-zero measure, by \( \text{supp}(\mu) \). 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 \( Q \times (E \cup H) \times \text{Disc}(Q) \). We define \( A \) as before.
We denote transition \((q, a, \mu)\) by \(q \xrightarrow{a} \mu\). We assume finite branching: for each state \(q\) the number of pairs \((\alpha, \mu)\) such that \(q \xrightarrow{\alpha} \mu\) is finite. Given a transition \(tr = (q, a, \mu)\) we denote \(q\) by \(\text{source}(tr)\) and \(\mu\) by \(\text{target}(tr)\).

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

\[
(q, a, q') \in D' \iff \exists \mu : (q, a, \mu) \in D \land \mu(q') > 0.
\]

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

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

A scheduler \(\sigma\) and a state \(q_0\) induce a measure \(\mu\) on the \(\sigma\)-field generated by cones of execution fragments as follows. If \(\alpha = q_0a_1q_1 \cdots a_kq_k\) is a finite execution fragment, then the cone of \(\alpha\) is defined by \(C_{\alpha} = \{\alpha' \in \text{frags}(\mathcal{P}) \mid \alpha \leq \alpha'\}\), and the measure of \(C_{\alpha}\) is defined by

\[
\mu(C_{\alpha}) = \prod_{i \in \{0, k-1\}} \sum_{(q_i, a_{i+1}, \mu') \in D} \sigma(q_0a_1 \cdots a_i q_i)((q_i, a_{i+1}, \mu')) \mu'(q_{i+1}).
\]

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

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

**Combined Transitions:** Let \(\{q_0 \xrightarrow{\alpha_i} \mu_i\}_{i \in I}\) 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 \((q_0, a, \sum_{i \in I} p_i \mu_i)\) is called a combined transition of \(\mathcal{P}\).

Consider a probabilistic execution fragment \(\mu\) that assigns probability 1 to the set of all finite execution fragments with trace \(a\). Let \(\mu'\) be the measure defined by \(\mu'(q) = \mu(\{\alpha \mid \text{lstate}(\alpha) = q\})\). Then \(\text{fstate}(\mu) \xrightarrow{\mu'} \mu'\) is a weak combined transition of \(\mathcal{P}\). If \(\mu\) can be generated by a deterministic scheduler, then \(\text{fstate}(\mu) \xrightarrow{\mu'} \mu'\) 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 \( \sum_{i \in I} p_i \cdot tr_i \) is a weak combined transition of \( \mathcal{P} \) labeled by \( a \).


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 \( \mathcal{P}_1 \parallel \mathcal{P}_2 \), is the PA \( \mathcal{P} = (Q_1 \times Q_2, (q_1, q_2), E_1 \cup E_2, H_1 \cup H_2, D) \) where \( D \) is the set of triples \( (q, a, \mu_1 \times \mu_2) \) such that, for \( i \in \{1, 2\} \):

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

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 \( C \) that is compatible with both \( \mathcal{P}_1 \) and \( \mathcal{P}_2 \), \( \mathcal{P}_1 \parallel C \leq_D \mathcal{P}_2 \parallel C \).

Simulation Relations: The definitions of forward simulation and weak forward simulation in Sect. 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 \( \mu_X \cdot R' \cdot \mu_Y \iff \) 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 \( \mu \) 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 \( \mu' \) 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 \( q \in \text{supp}(\mu) \), let \( q \xrightarrow{\mu} \mu_q \) be a weak combined transition of \( \mathcal{P} \). Let \( \mu' \) be \( \sum_{q \in \text{supp}(\mu)} \mu(q) \mu_q \). Then \( \mu \xrightarrow{\mu} \mu' \) is called a weak hyper-transition of \( \mathcal{P} \).

We now define simulations for probabilistic automata. A relation \( R \subseteq Q_1 \times \text{Disc}(Q_2) \) 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. \( \bar{q}_1 \xrightarrow{R} \bar{q}_2 \).
2. For each pair \( q_1, \mu_2 \) such that \( q_1 \xrightarrow{R} \mu_2 \) and each transition \( q_1 \xrightarrow{\mu_1} \mu_1' \) there exists a distribution \( \mu'_2 \in \text{Disc}(\text{Disc}(Q_2)) \) such that \( \mu_1' \xrightarrow{R} \mu'_2 \) and such that \( \mu_2 \xrightarrow{\text{flatten}} \text{flatten}(\mu'_2) \) (resp., \( \mu_2 \xrightarrow{\text{flatten}} \text{flatten}(\mu'_2) \)) is a hyper-transition (resp., a weak hyper-transition) of \( D_2 \).

We write \( \mathcal{P}_1 \preceq_{PF} \mathcal{P}_2 \) (resp., \( \mathcal{P}_1 \preceq_{wPF} \mathcal{P}_2 \)) 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. \( \mathcal{A}_1 \preceq_{F} \mathcal{A}_2 \) implies \( \mathcal{A}_1 \preceq_{PF} \mathcal{A}_2 \), and
2. \( \mathcal{A}_1 \preceq_{wF} \mathcal{A}_2 \) implies \( \mathcal{A}_1 \preceq_{wPF} \mathcal{A}_2 \).

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

1. If \( \mathcal{P}_1 \preceq_{PF} \mathcal{P}_2 \) then \( \mathcal{P}_1 \preceq_{wPF} \mathcal{P}_2 \).
2. If \( H_1 = H_2 = \emptyset \) then \( \mathcal{P}_1 \preceq_{PF} \mathcal{P}_2 \) iff \( \mathcal{P}_1 \preceq_{wPF} \mathcal{P}_2 \).
3. If \( \mathcal{P}_1 \preceq_{wPF} \mathcal{P}_2 \) then \( \mathcal{P}_1 \preceq_{DC} \mathcal{P}_2 \).

**Proof.** See [9].

**Tree-Structured Probabilistic Automata.** The unfolding of a probabilistic automaton \( \mathcal{P} \), denoted by \( \text{Unfold}(\mathcal{P}) \), is the tree-structured probabilistic automaton \( Q \) obtained from \( \mathcal{P} \) by unfolding its transition graph into a tree. Formally, \( Q = \text{exces}^*(\mathcal{P}), \bar{q}_Q = \bar{q}_\mathcal{P}, E_Q = E_\mathcal{P}, H_Q = H_\mathcal{P}, \text{ and } D_Q = \left\{ (\alpha, a, \mu) \mid \exists \mu', (\text{list}(\alpha), a, \mu') \in D_\mathcal{P}, \forall q \mu'(q) = \mu(q(\alpha a)) \right\} \).

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

**Proof.** It is easy to check that the relation \( R \) where \( \alpha R \delta(q) \) iff \( \text{list}(\alpha) = q \) is a probabilistic forward simulation from \( \text{Unfold}(\mathcal{P}) \) to \( \mathcal{P} \) and that the “inverse” of \( R \) is a probabilistic forward simulation from \( \mathcal{P} \) to \( \text{Unfold}(\mathcal{P}) \).

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

**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 $A_1 \leq_{DC} A_2$ implies the existence of a simulation relation from $A_1$ to $A_2$.

Our proofs of completeness for nondeterministic automata use the simple characterization in Proposition 6, applied to a special context for $A_1$ that we call the dual probabilistic automaton of $A_1$. Informally speaking, the dual probabilistic automaton of a nondeterministic automaton $A$ is a probabilistic automaton $C$ whose traces contain information about states and transitions of $A$. $C$’s states and start state are the same as those of $A$. For every state $q$ of $A$, $C$ has a self-loop transition labeled by $q$. Also, if $Tr$ is the (nonempty) set of transitions from $q$ in $A$, then from state $q$, $C$ has a uniform transition labeled by $ch$ to $\{\text{target}(tr) \mid tr \in Tr\}$.

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

- $Q_C = Q_A$, $\bar{q}_C = \bar{q}_A$,
- $E_C = E_A \cup \{ch\}$, $H_C = \emptyset$,
- $D_C = \{(q, ch, u((\{q' \mid q \rightarrow A q'\})) \mid q \rightarrow A\} \cup \{(q, q, q) \mid q \in Q_A\}$.

Since $C$ and $A$ share no actions, $C$ cannot ensure that its traces faithfully emulate the behavior of $A$. 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 $A_1$ and $A_2$ be tree-structured nondeterministic automata without internal actions, such that $A_1 \leq_{DC} A_2$. Then $A_1 \leq_{F} A_2$.

**Proof.** Assume that $A_1 \leq_{DC} A_2$. Let $C$ be the dual probabilistic automaton of $A_1$. Without loss of generality, we assume that the set of actions of $C$ is disjoint from those of $A_1$ and $A_2$. This implies that $C$ is compatible with both $A_1$ and $A_2$.

Consider the scheduler $\sigma_1$ for $A_1 || C$ that starts by scheduling the self-loop transition labelled by the start state of $C$, leading to state $(\bar{q}_1, \bar{q}_1)$, which is of the form $(q, q)$. Then $\sigma_1$ repeats the following as long as $q \rightarrow_1$:

1. Schedule the $ch$ transition of $C$, thus choosing a new state $q'$ of $A_1$. 

Compositionality for Probabilistic Automata 215
2. Schedule \((q, a, q')\) in \(A_1\), where \(a\) is uniquely determined by the selected state \(q'\) (recall that \(A_1\) is a tree).

3. Schedule the self-loop transition of \(C\) labeled by \(q'\), resulting in the state \((q', q')\), which is again of the form \((q, q)\).

Scheduler \(\sigma_1\) induces a trace distribution \(\mu_T\). Observe that \(\mu_T\) satisfies the following three properties, for all finite traces \(\beta\) and for all states \(q\):

\[
\mu_T(C_{q_1}) = 1
\]

\[
q \rightarrow_1 \Rightarrow \mu_T(C_{\beta q_1 ch}) = \mu_T(C_{\beta q_1})
\]

\[
\mu_T(C_{\beta q_1 ch}) > 0 \Rightarrow \sum_{q', q'' \mid q'' \rightarrow_1 q'} \mu_T(C_{\beta q_1 ch q''}) = \mu_T(C_{\beta q_1 ch})
\]

Since \(A_1 \leq_{DC} A_2\), Proposition 6 implies that \(\mu_T\) is also a trace distribution of \(A_2||C\). That is, there exists a probabilistic execution \(\mu\) of \(A_2||C\), induced by some scheduler \(\sigma_2\), whose trace distribution is \(\mu_T\). Now we define a relation \(R\): \(q_1 \rightarrow_R q_2\) if and only if there exists an execution \(\alpha\) of \(A_2||C\) 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 \(A_1\) to \(A_2\). For the start condition, we must show that \(\bar{q}_1 \rightarrow_R q_2\). Define execution \(\alpha\) 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), \(\mu_T(C_{q_1}) = 1\). Therefore, since there are no internal actions in \(A_2\) or \(C\), the only action that can be scheduled initially by \(\sigma_2\) is \(q_1\). Therefore, \(\sigma_2(\alpha)\) assigns probability 1 to the unique transition whose label is \(q_1\), as needed.

For the step condition, assume \(q_1 \rightarrow_R q_2\), and let \(q_1 \rightarrow_1 q_1'\). By definition of \(R\), there exists a finite execution \(\alpha\) of \(A_2||C\), 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 \(A_2||C\) 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 \(A_1\), Equation (2) implies that \(\mu_T(C_{\beta q_1 ch}) = \mu_T(C_{\beta q_1})\). Then since \(A_2\) and \(C\) have no internal actions, \(\sigma_2\) must schedule action \(ch\) from \(\alpha'\) with probability 1.

Since action \(ch\) leads to state \(q_1'\) of \(C\) with non-zero probability, which enables only actions \(q_1'\) and \(ch\), by Equation (3), \(\sigma_2\) 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 \(A_2\). Let \((q_2, a, q_2')\) be such a transition. Then, the sequence \(\alpha'' = \alpha' ch(q_2, q_1')(q_2, q_1')\) is an execution of \(A_2||C\) 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 \(q_1' \rightarrow_R q_2\), and completes the proof since we have found a state \(q_2'\) such that \(q_2 \rightarrow_2 q_2'\) and \(q_1' \rightarrow_R q_2'\).
Now we present our main result, for general (non-tree-structured) nondeterministic automata without internal actions.

**Theorem 1.** Let $A_1, A_2$ be nondeterministic automata without internal actions. Then $A_1 \leq_{DC} A_2$ if and only if $A_1 \leq_F A_2$.

**Proof.** First we prove soundness of forward simulations:

$$A_1 \leq_F A_2 \implies \text{(Proposition 7, Part 1)}$$
$$A_1 \leq_{PF} A_2 \implies \text{(Proposition 8, Part 1)}$$
$$A_1 \leq_{wPF} A_2 \implies \text{(Proposition 8, Part 3)}$$
$$A_1 \leq_{DC} A_2 \implies \text{(Proposition 3)}$$

Completeness is established by:

$$A_1 \leq_{DC} A_2 \implies \text{(Proposition 10)}$$
$$\text{Unfold}(A_1) \leq_{DC} A_1 \leq_{DC} A_2 \leq_{DC} \text{Unfold}(A_2) \implies (\leq_{DC} \text{ is transitive})$$
$$\text{Unfold}(A_1) \leq_{DC} \text{Unfold}(A_2) \implies \text{(Proposition 11)}$$
$$\text{Unfold}(A_1) \leq F \text{ Unfold}(A_2) \implies \text{(Proposition 3)}$$
$$A_1 \leq F \text{ Unfold}(A_1) \leq F \text{ Unfold}(A_2) \leq F A_2 \implies (\leq_F \text{ is transitive})$$
$$A_1 \leq_F A_2 \implies \text{(Proposition 3)}$$

### 4.2 Automata with Internal Actions

Next we extend the results of Sect. 4.1 to automata that include internal actions. The proofs are analogous to those in Sect. 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 $A_1, A_2$ be tree-structured nondeterministic automata such that $A_1 \leq_{DC} A_2$. Then $A_1 \leq_{wF} A_2$.

**Proof.** Assume that $A_1 \leq_{DC} A_2$. Let $C$ be the dual probabilistic automaton of $A_1$, and define scheduler $\sigma_1$ 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 $x$ of $A_2 \parallel C$ such that:

1. $\text{llstate}(x) = (q_2, q_1)$,
2. $\mu(C_{\alpha}) > 0$, and
3. there exists an execution fragment, $\alpha'$, of $A_2 \parallel C$, such that $\text{trace}(\alpha') = q_1$ and $\mu(C_{\alpha'-q}) > 0$.

We claim that $R$ is a weak forward simulation from $A_1$ to $A_2$. For the start condition, we show that $q_1 R q_2$. Define $\alpha$ to be the trivial execution consisting of the start state ($q_2, q_1$); this clearly satisfies Conditions 1 and 2. For Condition 3, observe that, by Equation (1), $\mu_T(C_{q_1}) = 1$. The inverse image under the trace mapping for $A_2 \parallel C$, of $C_{q_1}$, is a union of cones of the form $C_{\alpha'}$, where $\alpha'$ is
an execution of $A_2 \parallel C$ with trace $\bar{q}_1$; therefore, there exists such an $\alpha'$ with $\mu(C_{\alpha'}) > 0$. Since the first state of $\alpha'$ is $(\bar{q}_2, \bar{q}_1)$, $\alpha \parallel \alpha' = \alpha'$. Thus, $\mu(C_{\alpha' \parallel \alpha'}) > 0$, 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 $\alpha$ of $A_2 \parallel C$, with last state $(q_2, q_1)$, such that $\mu(C_{\alpha}) > 0$ and there exists an execution fragment, $\alpha'$, of $A_2 \parallel C$, such that $\text{trace}(\alpha') = q_1$ and $\mu(C_{\alpha' \parallel \alpha'}) > 0$. Let $\beta = \text{trace}(\alpha)$; then $\text{trace}(\alpha \parallel \alpha') = \beta q_1$, and so $\mu_T(C_{\beta q_1}) > 0$. Since $q_1$ enables at least one transition in $A_1$, Equation (2) implies that $\mu_T(C_{\beta q_1, ch}) = \mu_T(C_{\beta q_1})$. Thus there exists an execution fragment $\alpha''$ of $A_2 \parallel C$ with trace $ch$ such that $\mu(C_{\alpha'' \parallel \alpha''}) > 0$. Furthermore, since the transition of $C$ labeled by $ch$ leads to state $q'_1$ with non-zero probability, we can assume that the last state of $\alpha''$ 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 $\alpha''$ the only external actions of $C$ that are enabled are $ch$ and $q_1'$, there exists an execution fragment $\alpha'''$ with trace $a q'_1$ (a is uniquely determined by $q'_1$ since $A_1$ is tree-structured), such that $\mu(C_{\alpha'' \parallel \alpha''}) > 0$.

Now we split $\alpha'''$ into $\alpha'''' \parallel \alpha''''$, where $\text{trace}(\alpha''') = a$. Then the last state of $\alpha''''$ is of the form $(q''', q'_1)$. We claim that $q_1' R q'''$. Indeed, the execution $\alpha \parallel \alpha'' \parallel \alpha'''$ ends with state $(q''', q'_1)$ (Condition 1) and satisfies $\mu(C_{\alpha'' \parallel \alpha''}) > 0$ (Condition 2). Furthermore, $\alpha''''$ is an execution fragment that satisfies Condition 3.

It remains to show that $q_2 \not\xrightarrow{a} q'''$. For this, it suffices to observe that the execution fragment $(\alpha' \parallel \alpha'' \parallel \alpha_1') \parallel A_2$ has trace $a$, first state $q_2$, and last state $q'''$.

**Theorem 2.** Let $A_1$, $A_2$ be nondeterministic automata. Then $A_1 \leq_{DC} A_2$ if and only if $A_1 \leq_{WF} A_2$.

**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 $\leq_{WF}$, 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* $C$ for a PA $P$, which is slightly different from the one for nondeterministic automata. The main differences are that the new $C$ keeps track, in its state, of transitions as well as states of the given PA $P$, and that the new $C$ has separate transitions representing nondeterministic and probabilistic choices within $P$. Specifically, the states of $C$ include a distinguished start state, all the states of $P$, and all the transitions of $P$. $C$ has a special transition from its own start state $\bar{q}_C$ to
the start state of \( P \), \( \bar{q}_P \), labeled by \( \bar{q}_P \). Also, from every state \( q \) of \( P \), \( C \) has a uniform transition labeled by \( ch \) to the set of transitions of \( P \) that start in state \( q \). Finally, for every transition \( tr \) of \( P \), and every state \( q \) in the support of \( target(tr) \), \( C \) has a transition labeled by \( q \) from \( tr \) to \( q \).

**Definition 2.** The dual probabilistic automaton of a \( PA \) \( P \) is a \( PA \) \( C \) such that

- \( Q_C = \{\bar{q}_C\} \cup Q_P \cup D_P \),
- \( E_C = Q_P \cup \{ch\}, H_C = \emptyset \),
- \( D_C = \{(q, ch, U(\{tr \in D_P \mid source(tr) = q\})) \mid q \in Q_P \} \cup \{(tr, q, q) \mid tr \in D_P, q \in supp(target(tr))\} \).

**Proposition 13.** Let \( P_1, P_2 \) be tree-structured probabilistic automata without internal actions, such that \( P_1 \leq_{DC} P_2 \). Then \( P_1 \leq_{PF} P_2 \).

**Proof.** (Sketch:) Assume that \( P_1 \leq_{DC} P_2 \). Let \( C \) be the dual probabilistic automaton of \( P_1 \). Consider the scheduler \( \sigma_1 \) for \( P_1 || C \) that starts by scheduling the transition of \( C \) from the start state of \( C \) to the start state of \( P_1 \), leading to state \((q_1, q_1)\), which is of the form \((q, q)\). Then \( \sigma_1 \) repeats the following as long as \( q \rightarrow_1 \):

1. Schedule the \( ch \) transition of \( C \), thus choosing a transition \( tr \) of \( P_1 \).
2. Schedule transition \( tr \) of \( P_1 \), leading \( P_1 \) to a new state \( q' \).
3. Schedule the transition of \( C \) labeled by the state \( q' \), resulting in the state \((q', q')\), which is again of the form \((q, q)\).

Scheduler \( \sigma_1 \) induces a trace distribution \( \mu_T \). Since \( P_1 \leq_{DC} P_2 \), Proposition 6 implies that \( \mu_T \) is also a trace distribution of \( P_2 || C \). That is, there exists a probabilistic execution \( \mu \) of \( P_2 || C \), induced by some scheduler \( \sigma_2 \), whose trace distribution is \( \mu_T \).

For each state \( q_1 \) in \( Q_1 \), let \( \Theta_{q_1} \) be the set of finite executions of \( A_2 || C \) whose last transition is labeled by \( q_1 \). For each state \( q_2 \) of \( P_2 \), let \( \Theta_{q_1, q_2} \) be the set of finite executions in \( \Theta_{q_1} \) whose last state is the pair \((q_2, q_1)\). Now define relation \( R: q_1 R \mu q_2 \) 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 \( P_1 \) to \( P_2 \). The proof of this claim appears in [4].

**Theorem 3.** Let \( P_1, P_2 \) be probabilistic automata without internal actions. Then \( P_1 \leq_{DC} P_2 \) if and only if \( P_1 \leq_{PF} P_2 \).

**Proposition 14.** Let \( P_1, P_2 \) be tree-structured probabilistic automata such that \( P_1 \leq_{DC} P_2 \). Then \( P_1 \leq_{wPF} P_2 \).
Proof. (Sketch:) We use the same dual automaton C. Define scheduler σ₁ 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 \( P₁, P₂ \) be probabilistic automata. Then \( P₁ \preceq_{DC} P₂ \) if and only if \( P₁ \preceq_{wPF} P₂ \).

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, \( \preceq_F \), \( \preceq_{wF} \), \( \preceq_{PF} \), and \( \preceq_{wPF} \). 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 characterizations of the resulting new notions of \( \preceq_{DC} \), for instance in terms of button pushing scenarios.

References


8. R. Segala. Compositional trace-based semantics for probabilistic automata. In
9. R. Segala. Modeling and Verification of Randomized Distributed Real-Time Sys-
tems. PhD thesis, Department of Electrical Engineering and Computer Science,
MIT/LCS/TR-676.
11. M.I.A. Stoelinga. Alea jacta est: Verification of Probabilistic, Real-Time and Par-
Katoen, editor, Proceedings 5th International AMAST Workshop on Formal Me-
thods for Real-Time and Probabilistic Systems, Bamberg, Germany, volume 1601
14. M.I.A. Stoelinga and F.W. Vaandrager. A testing scenario for probabilistic auto-
mata. In J.C.M. Baeten, J.K. Lenstra, J. Parrow, and G.J. Woeginger, editors,
Proceedings 30th ICALP, volume 2719 of Lecture Notes in Computer Science, pages