Weak Bisimulation for Probabilistic Systems *

Anna Philippou[1], Insup Lee[2], and Oleg Sokolsky[2]

[1] Department of Computer Science, University of Cyprus, Cyprus
annap@cs.ucy.ac.cy
[2] Department of Computer and Information Science, University of Pennsylvania, USA lee@central.cis.upenn.edu, sokolsky@saul.cis.upenn.edu

Abstract. In this paper, we introduce weak bisimulation in the framework of Labeled Concurrent Markov Chains, that is, probabilistic transition systems which exhibit both probabilistic and nondeterministic behavior. By resolving the nondeterminism present, these models can be decomposed into a possibly infinite number of computation trees. We show that in order to compute weak bisimulation it is sufficient to restrict attention to only a finite number of these computations. Finally, we present an algorithm for deciding weak bisimulation which has polynomial-time complexity in the number of states of the transition system.

1 Introduction

In recent years, the need for reasoning about probabilistic behavior, as exhibited for instance in randomized, distributed and fault-tolerant systems, has triggered much interest in the area of formal methods for the specification and analysis of probabilistic systems [6, 11, 13–15, 27, 28, 30]. The general approach taken has been to extend existing models and techniques which have proved successful in the nonprobabilistic setting with probability.

Thus, much work in the area of formal models for probabilistic systems has been based on labeled transition systems [23]. In order to extend labeled transition systems to the probabilistic setting, various mechanisms for capturing probabilistic behavior have been proposed and investigated. On one end of the spectrum, several approaches have replaced nondeterministic branching in labeled transition systems with probabilistic branching [13] by assigning probabilities to each transition, while others explored the possibility of integrating nondeterministic and probabilistic behavior [30, 21, 13, 14, 28]. For example, in the reactive model of [13] as well as in the simple probabilistic automata of [28], probability distributions are dependent on the occurrence of actions, whereas in the stratified model, levelwise probabilistic branching is also possible [13]. A more general model for probabilistic computation is captured in the probabilistic transition systems of [30] and the probabilistic automata of [28], which extend the stratified model with nondeterminism among actions.

* This research was supported in part by NSF CCR-9619910, NSF CISE-9703220, ARO DAAG55-98-1-0393, ARO DAAG55-98-1-0466, and ONR N00014-97-1-0505 (MURI).

Verification techniques for these models have been inspired by successful approaches in the nonprobabilistic case. On one hand, temporal logics have been enriched with probability and on the other hand, probabilistic notions of equivalence and preorder relations have been explored. Among these, bisimulations [21], simulations [17, 29] and testing preorders [8, 10, 16], have been defined and algorithms given for their automatic verification [2, 9]. The majority of this work has focused on fully probabilistic systems, that is, systems where only probabilistic branching is involved.

In the nonprobabilistic setting, weak bisimulations have proved fundamental for the compositional verification of systems where abstraction from internal computation is essential. However, such notions have been rare in the setting of probabilistic systems and, as noted, although desirable their formalization has been problematic [14, 18, 3]. One paper that addresses this issue is [29], where notions of weak and branching bisimulations are introduced for a certain class of probabilistic transition systems defined by *simple probabilistic automata*. This model captures nondeterministic and probabilistic behavior by allowing from each state the nondeterministic choice of a number of probability distributions, each of which involves probabilistic transitions associated with a distinct action. The definition presented replaces the weak transition in the weak bisimulation definition of Milner [23], by assigning a (possibly infinite) set of distributions to each state, representing the (non-deterministic) alternatives of probabilistic distributions for states that are reachable by weak transitions. However, no method for computing the notion is considered in [29].

More recently, [3] has introduced a notion of weak bisimulation for fully probabilistic systems and presented a polynomial-time algorithm for deciding it. In this definition, weak transitions are replaced with the probability of making a transition to reach a certain state and the requirement of weakly bisimilar states is that the probability of a step by each process can be matched by the other. Another algorithmic approach for deciding weak equivalences for the model of [29] is proposed in [5]. The equivalence hereby obtained lies between strong and weak bisimulation.

In this paper, we propose a notion of weak bisimulation for probabilistic systems that allows for both nondeterministic and probabilistic branching. Such systems arise as formal models of randomized distributed systems, as well as real-life reactive systems that exhibit uncertainty. While probabilistic choice in these systems becomes relevant due to faults or random assignments, nondeterministic branching is also present due to the asynchronicity of a system's subprocesses or external intervention, as for instance an action taken by the environment.

Our definition of weak bisimulation in this model extends the definition of [3] by treating nondeterministic in addition to probabilistic behavior. It achieves this by incorporating the notion of a *scheduler*, an entity that resolves nondeterminism in a system by choosing the next step to take place, out of a set of nondeterministic alternatives. Indeed, due to the presence of nondeterminism it is not possible in general to determine the probability with which a weak transition may take place. Instead, we associate such a probability with each of the

possible schedulers and in order to establish weak bisimulation we compare the set of possible probabilities for each of two states. Our first main result is that according to our definition, weak bisimulation can be characterized in terms of maximum probabilities of transitions over all schedulers. We then turn to tackle the problem of computing such probability bounds. Although the set of schedulers for a system is in general infinite, our second main result shows that in order to compute maximum probabilities it is sufficient to consider only a finite number of them. In particular, we introduce the notion of *determinate* schedulers and we isolate a finite set of schedulers in which maximal probabilities arise. On the basis of the above, we present an algorithm for deciding weak bisimulation equivalence classes, in time polynomial in the number of states of the underlying probabilistic system. Thus the main contribution of this paper is the definition of weak bisimulation in a general framework of probabilistic systems and the corresponding algorithm for computing weak bisimulation equivalence classes.

The remainder of the paper is organized as follows: the following section contains an account of the Labeled Concurrent Markov Chain model and some background material, section 3 introduces and studies the notion of weak bisimulation, section 4 presents determinate schedulers and their relevance to weak bisimulation, while section 5 describes an algorithm for deciding weak bisimulation classes. We conclude with a comparison of our proposal with that of [29] and a discussion of further work. Due to the limitation of space, proofs of results are only briefly sketched, the complete proofs can be found in [25].

2 The Model

In this section we introduce Labeled Concurrent Markov Chains and some background definitions and notations we will be using.

Definition 1. A *Labeled Concurrent Markov Chain* (LCMC) is a tuple $\langle S_n, S_p, Act, \longrightarrow_n, \longrightarrow_p, s_0\rangle$, where S_n is the set of nondeterministic states, S_p is the set of probabilistic states, $Act = L \cup \{\tau\}$ is the set of labels, (where τ is the internal action), $\longrightarrow_n \subset S_n \times Act \times (S_n \cup S_p)$ is the nondeterministic transition relation, $\longrightarrow_p \subset S_p \times (0,1] \times S_n$ is the probabilistic transition relation, satisfying $\Sigma_{(s,\pi,t)\in\longrightarrow_p} \pi = 1$ for all $s \in S_p$, and $s_0 \in S_n \cup S_p$ is the initial state. □

Thus the set of states of an LCMC is partitioned into two sets, S_n and S_p. States in S_n are capable of performing nondeterministic transitions while states in S_p may perform probabilistic transitions. We assume both of these sets to be finite. Note that the nonprobabilistic model is derivable from the LCMC model by setting $S_p = \emptyset$. In what follows we will write S for $S_n \cup S_p$ and we will let s, s' range over S, α, β over Act and ℓ over $Act \cup (0,1]$. In addition, when it is clear within a context, we will refer to a LCMC $\langle S_n, S_p, Act, \longrightarrow_n, \longrightarrow_p, s_0\rangle$ by s_0.

Computations of LCMC's arise by resolving the nondeterministic and probabilistic choices:

Definition 2. A *computation* in $\Xi = \langle S_n, S_p, Act, \longrightarrow_n, \longrightarrow_p, s_0\rangle$ is either a finite sequence $s_0\,\ell_1\,s_1 \ldots \ell_k\,s_k$, where s_k has no transitions, or an infinite sequence $s_0\,\ell_1\,s_1 \ldots \ell_k\,s_k \ldots$, such that $(s_i, \ell_{i+1}, s_{i+1}) \in \longrightarrow_p \cup \longrightarrow_n$, for all $0 \leq i$.

We denote by $\mathsf{Comp}(\Xi)$ the set of all computations of Ξ and by $\mathsf{Comp}_{fin}(\Xi)$ the set of all partial computations of Ξ, i.e. $\mathsf{Comp}_{fin}(\Xi) = \{s_0\ell_1 \ldots \ell_k s_k \mid \exists c \in \mathsf{Comp}(\Xi) \cdot\ c = s_0\ell_1 \ldots \ell_k s_k \ldots\}$. Given $c = s_0\ell_1 \ldots \ell_k s_k \in \mathsf{Comp}_{fin}(\Xi)$, we define $\mathsf{trace}\, c = \ell_1 \ldots \ell_k \restriction L$, $\mathsf{inter}\, c = \{s_0, \ldots, s_{k-1}\}$, $\mathsf{first}\, c = s_0$ and $\mathsf{last}\, c = s_k$.

To define probability measures on computations, it is necessary to resolve the nondeterminism present. To achieve this, the notion of a scheduler has been employed [30, 14, 29]. A scheduler is an entity that given a partial computation (ending in a nondeterministic state) chooses the next transition to be scheduled:

Definition 3. A *scheduler* of a LCMC Ξ is a function $\sigma : \mathsf{Comp}_{fin}(\Xi) \mapsto (\longrightarrow_n \cup \bot)$, such that, if $\sigma(c) = tr \in \longrightarrow_n$ then $tr = (\mathsf{last}\, c, \alpha, s)$ for some α and s. □

Here we use $\sigma(c) = \bot$ to express that a scheduler may schedule nothing at some point during computation. In the rest of the paper we will use $\mathsf{Sched}(\Xi)$ to denote the set of schedulers of Ξ, and we will let σ range over all schedulers. For a LCMC Ξ and a scheduler $\sigma \in \mathsf{Sched}(\Xi)$ we define the set of *scheduled computations* $\mathsf{Scomp}(\Xi, \sigma) \subseteq \mathsf{Comp}(\Xi)$, to be the finite computations $c = s_0\, \ell_1 \ldots \ell_k\, s_k$ where for all $i < k$, $s_i \in \mathsf{S}_n$ $\sigma(s_0\, \ell_1 \ldots \ell_i s_i) = (s_i, \ell_{i+1}, s_{i+1})$, and $\sigma(s_0\, \ell_1 \ldots \ell_k\, s_k) = \bot$, and the infinite computations $c = s_0\, \ell_1 \ldots \ell_k\, s_k \ldots$ where for all i, $s_i \in \mathsf{S}_n$, $\sigma(s_0\, \ell_1 \ldots \ell_i s_i) = (s_i, \ell_{i+1}, s_{i+1})$. Each scheduler σ induces a probability space on $\mathsf{Scomp}(\Xi, \sigma)$ in the usual way [29].

We conclude with the definition of strong bisimulation for the model. First we have a definition.

Definition 4. Given $s, s' \in \mathsf{S}$, and $\mathcal{M} \subseteq \mathsf{S}$, we define

1. $$\mathsf{pr}(s, s') = \begin{cases} \pi, & \text{if } s \xrightarrow{\pi}_p s' \\ 1, & \text{if } s = s', s \in S_n \\ 0, & \text{otherwise} \end{cases}$$
2. $\mu(s, \mathcal{M}) = \Sigma_{s' \in \mathcal{M}} \mathsf{pr}(s, s')$. □

Thus, $\mathsf{pr}(s, s')$ denotes the probability that s may perform at most one probabilistic transition to become s', and $\mu(s, \mathcal{M})$ denotes the cumulative probability that s may perform a probabilistic action to a state in $\mathcal{M}$.

Definition 5. An equivalence relation $\mathcal{R} \subseteq \mathsf{S} \times \mathsf{S}$ is a *strong bisimulation* if, whenever $s\mathcal{R}t$

1. for all $\alpha \in Act$, if $s, t \in S_n$ and $s \xrightarrow{\alpha}_n s'$ then $t \xrightarrow{\alpha}_n t'$ and $s'\mathcal{R}t'$;
2. for all $\mathcal{M} \in \mathsf{S}/\mathcal{R}$, $\mu(s, \mathcal{M}) = \mu(t, \mathcal{M})$.

Two states s and t are *strong bisimulation equivalent*, written $s \sim t$, if there exists a strong bisimulation $\mathcal{R}$ such that $s\mathcal{R}t$. □

An example of strong bisimulation equivalent systems is shown in Figure 1.

The above definition is almost identical to the one proposed in [14], where an *alternating* model is considered, However, with a slight reformulation of the definition of $\mathsf{pr}(s, s')$, Definition 5 allows for pairs of probabilistic and nondeterministic systems, such as (s, x), to be considered as bisimulation equivalent.

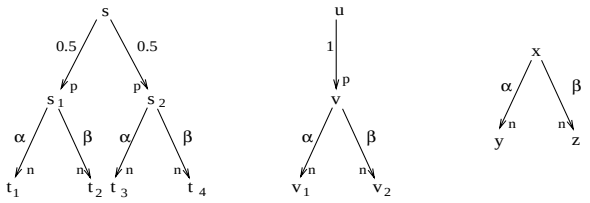

Fig. 1. $s \sim u \sim x$

3 Weak Bisimulation

In this section we define weak bisimulation for probabilistic transition systems. Weak bisimulation was introduced in the context of nonprobabilistic transition systems in [23]. It abstracts away from internal computation by focusing on *weak* transitions, that is transitions of the form $\Longrightarrow \xrightarrow{\alpha} \Longrightarrow$ (where $\Longrightarrow$ is the transitive, reflexive closure of $\xrightarrow{\tau}$) and requires that weakly bisimilar systems can match each other's observable behavior.

However, in the probabilistic setting, while considering a transition with observable content $\alpha \in Act$ ($\mathsf{trace}\, c = \alpha$), it is necessary to take account of the probability of the transition taking place, and to ensure that weakly bisimilar systems may not only match one another's transitions but also perform the transitions with matching probabilities. To achieve this, given a state s, an action α, and an equivalence class of the weak bisimulation relation, $\mathcal{M}$, we are interested in computing the probability of s reaching a state in $\mathcal{M}$ via a transition with observable content α. This probability depends on how the nondeterminism of s is resolved. So, given a LCMC Ξ, $\Phi \subset Act^*$, $\mathcal{M} \subseteq \mathsf{S}$ and $\sigma \in \mathsf{Sched}(\Xi)$ we define

$$\mathsf{Paths}(\Xi, \Phi, \mathcal{M}, \sigma) \;=\; \{c \in \mathsf{Scomp}(\Xi, \sigma) \mid \mathsf{last}\, c \in \mathcal{M},\ \sigma(c) = \bot,\ \mathsf{trace}\, c \in \Phi\}.$$

Thus, $\mathsf{Paths}(\Xi, \Phi, \mathcal{M}, \sigma)$ denotes the set of computations of Ξ, scheduled by σ, leading to a state in $\mathcal{M}$ via a sequence of actions in Φ. In what follows, we use ε to denote the empty word. We also often abbreviate the singleton set $\{x\}$ by x.

The probability $\mathsf{Pr}(s, \Phi, \mathcal{M}, \sigma) \stackrel{\text{def}}{=} \mathcal{P}(\mathsf{Paths}(\Xi, \Phi, \mathcal{M}, \sigma))$, s is the initial state of Ξ, is given by the smallest solution to $X(s, \Phi, \mathcal{M}, \sigma, s)$ defined by the following set of equations:

$$X(s, \Phi, \mathcal{M}, \sigma, c) = \begin{cases} 1, & \text{if } \varepsilon \in \Phi, s \in \mathcal{M},\ \sigma(c) = \bot \\ 0, & \text{if } (\varepsilon \notin \Phi, s \notin \mathcal{M},\ \sigma(c) = \bot) \text{ or } \Phi = \emptyset \\ \Sigma_t\ \mathsf{pr}(s,t)\ \cdot X(t, \Phi, \mathcal{M}, \sigma, c\, \mathsf{pr}(s,t)\, t), & \text{if } s \in S_p \\ X(t, \Phi - \alpha, \mathcal{M}, \sigma, c\, \alpha\, t), & \text{if } s \in S_n, \sigma(c) = (s, \alpha, t) \end{cases}$$

where $\Phi - \alpha = \{\phi \mid \alpha\phi \in \Phi\}$. Note that the last argument of $X(s, \Phi, \mathcal{M}, \sigma, c)$, c, records the history of reaching s. This is needed for performing subsequent scheduling of s under scheduler σ. Moreover, we use regular expressions (such as $\tau^*\alpha\tau^*\beta\tau^*$) to represent sets of traces. So, for example, $\mathsf{Pr}(s, \tau^*\alpha\tau^*, \mathcal{M}, \sigma)$

denotes the probability to reach some state in $\mathcal{M}$ from state s, at the endpoints of scheduler σ, by performing a weak transition with observable content α.

The definition of weak bisimulation follows. As usual we write $\widehat{\alpha}$ for α if $\alpha \in L$ and ε otherwise. Furthermore, given a relation $\mathcal{R}$, we write $\mu_{\mathcal{R}}(s, \mathcal{M})$ for the probability of reaching $\mathcal{M} \subseteq \mathsf{S}$ from state s weighted by the probability of exiting the equivalence class $[s]_{\mathcal{R}}$:

$$\mu_{\mathcal{R}}(s, \mathcal{M}) = \begin{cases} \mu(s, \mathcal{M})/(1 - \mu(s, [s]_{\mathcal{R}})), & \text{if } \mu(s, [s]_{\mathcal{R}}) \neq 1, \\ \mu(s, \mathcal{M}), & \text{otherwise} \end{cases}$$

Definition 6. An equivalence relation $\mathcal{R} \subseteq \mathsf{S} \times \mathsf{S}$ is a *weak bisimulation* if whenever $s \mathcal{R} t$,

1. for all $\alpha \in Act$, if $s, t \in S_n$ and $s \xrightarrow{\alpha}_n s'$, then there exists $\sigma \in \mathsf{Sched}(t)$ such that $\mathsf{Pr}(t, \tau^* \widehat{\alpha} \tau^*, [s']_{\mathcal{R}}, \sigma) = 1$;
2. there exists $\sigma \in \mathsf{Sched}(t)$ such that for all $\mathcal{M} \in \mathsf{S}/\mathcal{R} - [s]_{\mathcal{R}}$, $\mu_{\mathcal{R}}(s, \mathcal{M}) = \mathsf{Pr}(t, \tau^*, \mathcal{M}, \sigma)$.

We say that s and t are weakly bisimilar, written $s \approx t$, if $(s, t) \in \mathcal{R}$ for some weak bisimulation $\mathcal{R}$. □

Thus the definition specifies how deterministic and probabilistic transitions are matched by weakly bisimilar states. Nondeterministic behavior is matched as follows: if one of the system can engage in a transition involving an action there exists a scheduler of the other which weakly performs the same transition with probability one. On the other hand, if $s \approx t$ then, given a probabilistic branching of s, that is a set of transitions $s \xrightarrow{\pi_i}_p s_i$, there exists a *single* scheduler of t that weakly matches the branching, in the sense that the weighted probabilities of reaching any equivalence class $\mathcal{M}$ are the same for both systems. The purpose for considering weighted probabilities is highlighted in the example of Figure 2(a).

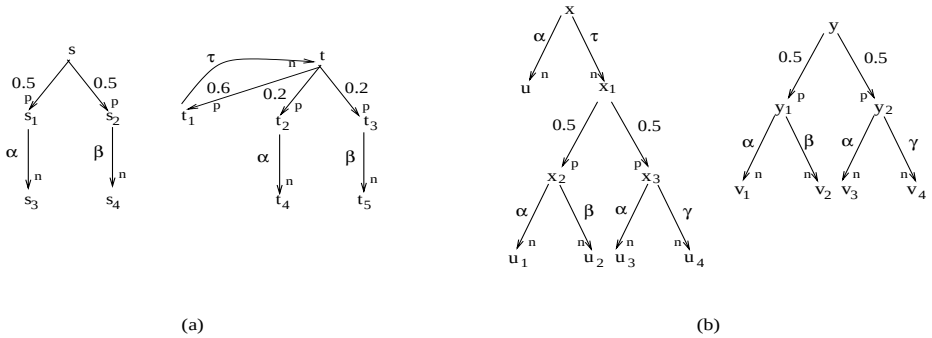

Fig. 2. (a)$s \approx t$ and (b) $x \approx y$

We observe that states s and t can both engage in actions α and β equally likely. Furthermore, although, $\mathsf{pr}(t, t_2) = \mathsf{pr}(t, t_3) = 0.2$, t may also probabilisti-

cally reach t_1 where clearly $t \approx t_1$, in this way contributing to the probability of t eventually performing either α and β. Thus, t can weakly perform both α and β with probability 0.5 which suggests s and t should be considered weakly bisimilar. To allow this, on matching probabilistic transitions of weakly bisimilar states, we consider the probability of reaching an equivalence class weighted by the probability of exiting the equivalence class of the initial state. Thus, the weak bisimulation that connects s and t is $\mathcal{R} = \{\{s, t, t_1\}, \{s_1, t_2\}, \{s_2, t_3\}, \{s_3, s_4, t_4, t_5\}\}$. To establish satisfaction of the relation for the pair (s, t), we observe that clause 2 of Definition 6 is satisfied as $\mu_{\mathcal{R}}(s, [s_1]_{\mathcal{R}}) = 0.5$ and since $\mu(t, [t]_{\mathcal{R}}) = 0.6$, we have that $\mu_{\mathcal{R}}(t, [s_1]_{\mathcal{R}}) = \mu_{\mathcal{R}}(t, [t_2]_{\mathcal{R}}) = \frac{0.2}{1 - 0.6} = 0.5$ as required. Similarly it can be shown that $\mu_{\mathcal{R}}(s, [s_2]_{\mathcal{R}}) = \mu_{\mathcal{R}}(t, [s_2]_{\mathcal{R}}) = 0.5$.

Another example of weakly bisimilar systems is exhibited in Figure 2(b). We have that states x and y are related by the weak bisimulation $\mathcal{R} = \{\{x, y, x_1\}, \{x_2, y_1\}, \{x_3, y_2\}, \{u, u_1, u_2, u_3, u_4, v_1, v_2, v_3, v_4\}\}$. We point out that while $x \xrightarrow{\alpha}_n u$, there is $\sigma \in \mathsf{Sched}(y)$ such that $\mathsf{Pr}(y, \tau^*\alpha\tau^*, [u]_{\mathcal{R}}, \sigma) = 1$, as required.

Weak bisimulation satisfies the following properties:

Lemma 1. $\approx$ is the largest weak bisimulation and $\sim \subset \approx$. □

Given $\alpha \in Act$ and $\mathcal{M} \subseteq \mathsf{S}$, we introduce the following notation for the largest probabilities of reaching $\mathcal{M}$ via a path with observable content α, over all schedulers: $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M}) \stackrel{\text{def}}{=} \max_{\sigma \in \mathsf{Sched}(s)} \mathsf{Pr}(s, \tau^*\widehat{\alpha}\tau^*, \mathcal{M}, \sigma)$. Our first important result states that weak bisimulation preserves maximum probabilities:

Theorem 1. If $s \approx t$ then, for all $\alpha \in Act$ and $\mathcal{M} \in \mathsf{S}/\approx$, $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M}) = \mathsf{Pr}_{max}(t, \alpha, \mathcal{M})$. □

PROOF: The proof involves showing that if $s \approx t$, for any $\sigma \in \mathsf{Sched}(s)$, $\alpha \in Act$ and $\mathcal{M} \in \mathsf{S}/\mathcal{R}$ there exists $\sigma' \in \mathsf{Sched}(t)$ such that $\mathsf{Pr}(s, \tau^*\alpha\tau^*, \mathcal{M}, \sigma) \leq \mathsf{Pr}(t, \tau^*\widehat{\alpha}\tau^*, \mathcal{M}, \sigma')$. To do this we must match every move made by s via σ, by a scheduler of t, using the weak bisimulation definition, concatenating schedulers on the way. It then remains to show that the sets of equations that define the two probabilities have the same least solutions. □

We continue with a result which is central to the understanding of the interplay between probabilistic behavior and the definition of weak bisimulation. Suppose $s \xrightarrow{\pi_i}_p s_i$. The result states that on matching such a transition in t, $t \approx s$, a scheduler only passes via states that are weakly bisimilar to s.

Lemma 2. Suppose $s \approx t$. Then, if $\sigma \in \mathsf{Sched}(t)$ is such that for all $\mathcal{M} \in \mathsf{S}/\approx$, $\mu_{\approx}(s, \mathcal{M}) = \mathsf{Pr}(t, \tau^*, \mathcal{M}, \sigma)$, then, for all partial computations $c \in \mathsf{Scomp}_{fin}(t, \sigma)$, if $\sigma(c) \neq \perp$, and $t' = \mathsf{last}\, c$, either $t' \approx t$ or $\mathsf{Pr}(t', \tau^*, [t']_{\approx}, \sigma) = 1$. □

PROOF: Suppose $s \approx t$ and pick $\sigma \in \mathsf{Sched}(t)$ such that for all $\mathcal{M} \in \mathsf{S}/\approx$, $\mu_{\approx}(s, \mathcal{M}) = \mathsf{Pr}(t, \tau^*, \mathcal{M}, \sigma)$. We assume for the sake of contradiction that there exists computation $c \in \mathsf{Scomp}(t, \sigma)$, such that $\sigma(c) \neq \perp$, $t' = \mathsf{last}\, c$ with $t' \not\approx t$ and $\mathsf{Pr}(t', \tau^*, [t']_{\approx}, \sigma) \neq 1$. (Note that $\mathsf{Pr}(t', \tau^*, [t']_{\approx}, \sigma) = 1$ implies that $\mu_{\approx}(s, [t']_{\approx}) > 0$.) Computation of the probabilities $\mathsf{Pr}_{max}(s, \tau, [t']_{\approx})$, $\mathsf{Pr}_{max}(t, \tau, [t']_{\approx})$ results in violation of Theorem 1 which completes the proof. □

This result implies that on matching a probabilistic branching of weakly bisimilar states the branching structure of the states is preserved. It comes in agreement with the senario of [3] where it is shown that for fully probabilistic systems weak and branching bisimulations coincide. Of course this does not hold in our model due to the presence of nondeterminism.

Let $\mathsf{Sched}'(t, M)$ be the subset of schedulers of t containing all schedulers that schedule within the set of states M and consider the case $M = [t]_{\approx}$. We may prove by structural induction of t that $\mathsf{Sched}'(t, [t]_{\approx}) = \emptyset$, and that each $\sigma_t \in \mathsf{Sched}'(t, [t]_{\approx})$ satisfies the requirement of Lemma 2: If $s \approx t$ and $\sigma_t \in \mathsf{Sched}'(t, [t]_{\approx})$, then for all $\mathcal{M} \in \mathsf{S}/\approx$, $\mu_{\approx}(s, \mathcal{M}) = \mathsf{Pr}(t, \tau^*, \mathcal{M}, \sigma_t)$. Thus letting $\mathsf{Pr}'_{max}(s, \alpha, \mathcal{M}, M) \stackrel{\text{def}}{=} \max_{\sigma \in \mathsf{Sched}'(s,M)} \mathsf{Pr}(s, \tau^* \widehat{\alpha} \tau^*, \mathcal{M}, \sigma)$, we have that for all $\mathcal{M} \in \mathsf{S}/\approx$, $Pr'_{max}(t, \tau, \mathcal{M}, [t]_{\approx}) = \mu_{\approx}(s, \mathcal{M})$.

As a consequence, we have an alternative definition of weak bisimulation in terms of maximum probabilities.

Theorem 2. An equivalence relation $\mathcal{R} \subseteq \mathsf{S} \times \mathsf{S}$ is a *weak bisimulation iff* whenever $s\mathcal{R}t$, then

1. if $s, t \in S_n$, $\alpha \in Act$ and $\mathcal{M} \in \mathsf{S}/\mathcal{R}$, then $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M}) = \mathsf{Pr}_{max}(t, \alpha, \mathcal{M})$;
2. for all $\mathcal{M} \in \mathsf{S}/\mathcal{R} - [s]_{\mathcal{R}}$, $\mu_{\mathcal{R}}(s, \mathcal{M}) = \mathsf{Pr}'_{max}(t, \tau, \mathcal{M}, [t]_{\mathcal{R}})$. □

PROOF: Let $\mathcal{R}$ be a weak bisimulation and suppose $s\mathcal{R}t$. Then, by Theorem 1, $\mathcal{R}$ satisfies condition 1 above. In addition, by the observation above, $\mu_{\mathcal{R}}(s, \mathcal{M}) = \mathsf{Pr}'_{max}(t, \tau^*, \mathcal{M}, [t]_{\mathcal{R}})$, thus condition 2 is also satisfied.

To prove the converse, suppose $\mathcal{R}$ satisfies the conditions of the theorem. By condition 1, $\mathcal{R}$ satisfies Definition 6(1). To establish the second condition it is sufficient to note that whenever $t \stackrel{\tau}{\longrightarrow}_n t_1$, $t \stackrel{\tau}{\longrightarrow}_n t_2$ with $t\mathcal{R}t_1\mathcal{R}t_2$ and $(s, t) \in \mathcal{R}$, for all $\mathcal{M} \in \mathsf{S}/\mathcal{R}$, $\mu_{\mathcal{R}}(s, \mathcal{M}) = \mathsf{Pr}'_{max}(t_1, \tau^*, \mathcal{M}, [t_1]_{\approx})$, and $\mu_{\mathcal{R}}(s, \mathcal{M}) = \mathsf{Pr}'_{max}(t_2, \tau^*, \mathcal{M}, [t_2]_{\approx})$. This implies that for any $\sigma \in \mathsf{Sched}'(t, [t]_{\mathcal{R}})$, $\mu_{\mathcal{R}}(s, \mathcal{M}) = \mathsf{Pr}(t, \tau^*, \mathcal{M}, \sigma)$, for all $\mathcal{M} \in \mathsf{S}/\mathcal{R}$. Thus, Definition 6(2) holds. □

4 Determinate Schedulers

In this section we turn to the issue of deciding weak bisimulation for probabilistic systems. According to Theorem 2, establishing weak bisimilarity of two systems amounts to computing certain maximum probabilities. These probabilities are quantified over the set of all schedulers, and as noted earlier, such sets are in general infinite. The question then arises whether it is possible to compute maximum probabilities by looking only at a finite subset of schedulers.

So let $s \in \mathsf{S}$, $\alpha \in Act$ and $\mathcal{M} \in \mathsf{S}/\approx$ and consider $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M})$. First we point out that the only schedulers relevant to computing this probability are such that $\sigma(c) = \perp$ unless $\mathsf{trace}\, c \in \{\varepsilon, \alpha\}$. Given a set of schedulers D, let D^{α} be the subset of D which only contains such schedulers. We may see that in general D^{α} is an infinite set. For example, consider agent s in Figure 3. The family of schedulers $\{\sigma_i\}_{i \geq 0}$ defined below is such that $\sigma_k \in \mathsf{Sched}(s)^{\alpha}$.

$$\sigma_k(s\, 0.5\, (u\, \tau s\, 0.5)^i\, u) = (u, \tau, s), \text{ if } i < k$$
$$\sigma_k(s\, 0.5\, (u\, \tau s\, 0.5)^k\, u) = (u, \beta, y)$$

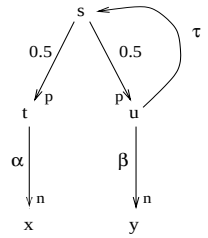

Fig. 3. Determinate schedulers

We define the following set of schedulers:

Definition 7. Let Ξ be a LCMC and $\sigma \in \mathsf{Sched}(\Xi)$. We say that scheduler σ is *determinate* if for all $c, c' \in \mathsf{Comp}_{fin}(\Xi)$ with $\mathsf{last}\, c = \mathsf{last}\, c'$ and $\mathsf{trace}\, c = \mathsf{trace}\, c'$, $\sigma(c) = \sigma(c')$. We write $\mathsf{DSched}(\Xi)$ for the set of determinate schedulers of Ξ. □

It is not difficult to see that $\mathsf{DSched}(\Xi)^\alpha$ is a finite set. Furthermore, it turns out that in order to compute $\mathsf{Pr}_{max}(\Xi, \alpha, \mathcal{M})$ it is sufficient to restrict our attention to $\mathsf{DSched}(\Xi)$ (and consequently to $\mathsf{DSched}(\Xi)^\alpha$).

Theorem 3. Suppose $s \in \mathsf{S}$. Then if $\alpha \in Act$, $\mathcal{M} \in \mathsf{S}/\approx$, (1) if $\sigma \in \mathsf{Sched}(s)$ there exists $\sigma' \in \mathsf{DSched}(s)$ such that $\mathsf{Pr}(s, \tau^*\alpha\tau^*, \mathcal{M}, \sigma) \leq \mathsf{Pr}(s, \tau^*\widehat{\alpha}\tau^*, \mathcal{M}, \sigma')$, and (2) if $\sigma \in \mathsf{Sched}(s, [s]_\approx)$, there exists $\sigma' \in \mathsf{DSched}(s) \cap \mathsf{Sched}'(s, [s]_\approx)$ such that $\mathsf{Pr}(s, \tau^*, \mathcal{M}, \sigma) \leq \mathsf{Pr}(s, \tau^*, \mathcal{M}, \sigma')$. □

Proof: The proof involves transforming an arbitrary scheduler into a determinate one without decreasing the probability of interest, by scheduling from each state the transition that maximizes the desired probability. □

The problem of computing maximum (and minimum) probabilities of properties was also tackled in the context of model checking for probabilistic extensions of the CTL temporal logic [7, 12, 4]. In these works, the challenge had been to compute the probability bounds that certain logical properties are satisfied by probabilistic systems. As was shown in the papers just cited, to compute such probabilities it is sufficient to consider only the (finite) set of *simple* schedulers, where a scheduler σ is *simple* if for all c, with $\mathsf{last}\, c = \mathsf{last}\, c'$, $\sigma(c) = \sigma(c')$. Thus, a simple scheduler is also determinate. However, simple schedulers are insufficient for computing $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M})$. For instance consider the LCMC s in Figure 4. None of the two simple schedulers of the system achieves probability $\mathsf{Pr}_{max}(s, \alpha, [u]_\approx)$. On the other hand, $\mathsf{DSched}(s)$ contains σ, where $\sigma(s) = (s, \tau, t)$, $\sigma(s\tau t) = (t, \alpha, t)$, $\sigma(s\tau t\alpha t) = (t, \tau, u)$, and indeed, $\mathsf{Pr}(s, \alpha, [u]_\approx, \sigma) = \mathsf{Pr}_{max}(s, \alpha, [u]_\approx) = 1$.

5 The Algorithm

In this section we develop an algorithm for deciding weak bisimulation for labeled concurrent Markov chains. The basic idea originates from the bisimulation

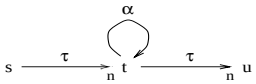

Fig. 4. Simple vs. determinate schedulers

algorithms of [24, 19], where a partitioning technique is employed to compute strong bisimulation equivalence classes for nonprobabilistic systems. Briefly, this technique involves considering the set of states of the system in question, S, and beginning with the trivial partition of this set $\mathcal{W} = \{S\}$, successively refining the partition until the set of weak bisimulation equivalence classes is reached. The refinement method is based on the notion of 'splitters'.

Definition 8. Let $\mathcal{W}$ be a partition of $S \subset \mathsf{S}$. Then a *splitter* of $\mathcal{W}$ is a triple $(\mathcal{C}, \alpha, \mathcal{M})$, where $\mathcal{C} \in \mathcal{W}$, $\alpha \in Act$ and $\mathcal{M} \in \mathcal{W}$, such that there are $s, s' \in \mathcal{C}$ and $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M}) \neq \mathsf{Pr}_{max}(s', \alpha, \mathcal{M})$ or $\mu_{\mathcal{W}}(s, \mathcal{M}) > \mathsf{Pr}'_{max}(s', \tau, \mathcal{M}, \mathcal{C})$, or $\mu_{\mathcal{W}}(s', \mathcal{M}) > \mathsf{Pr}'_{max}(s, \tau, \mathcal{M}, \mathcal{C})$.

Thus a splitter $(\mathcal{C}, \alpha, \mathcal{M})$ of a partition $\mathcal{W}$ isolates a class of $\mathcal{W}$ which contains states that prevent $\mathcal{W}$ from being a weak bisimulation. So suppose $(\mathcal{C}, \alpha, \mathcal{M})$ a splitter of a partition $\mathcal{W}$. The purpose of function Split is to partition $\mathcal{C}$ into classes $\mathcal{C}_1 \ldots \mathcal{C}_k$, none of which can be split by the pair $(\alpha, \mathcal{M})$. Formally, $\mathsf{Split}(\mathcal{C}, \alpha, \mathcal{M}) = \mathcal{C}/\equiv$, where $s \equiv t$ *iff* $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M}) = \mathsf{Pr}_{max}(t, \alpha, \mathcal{M})$, $\mu_{\mathcal{W}}(s, \mathcal{M}) \leq \mathsf{Pr}'_{max}(s', \tau, \mathcal{M}, [s]_{\equiv})$, and $\mu_{\mathcal{W}}(s', \mathcal{M}) \leq \mathsf{Pr}'_{max}(s, \tau, \mathcal{M}, [s']_{\equiv})$. Taking this a step further, given a splitter $(\mathcal{C}, \alpha, \mathcal{M})$ of a partition $\mathcal{W}$, it is possible to refine $\mathcal{W}$ as follows: $\mathsf{Refine}(\mathcal{W}, \mathcal{C}, \alpha, \mathcal{M}) = \mathsf{Split}(\mathcal{C}, \alpha, \mathcal{M}) \cup (\mathcal{W} - \mathcal{C})$. The correctness of the procedure is given by a generalization of Theorem 2.

Theorem 4. Suppose $\mathcal{M} = \bigcup_{i \in I} \mathcal{M}_i$ and $\mathcal{M}' = \bigcup_{i \in I'} \mathcal{M}'_i$, where each $\mathcal{M}_i, \mathcal{M}'_i$ is in $\mathsf{S}/\approx$ and further let $s \approx t$, $s \in \mathcal{M}'$. The following hold:

1. for all $\alpha \in Act$, $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M}) = \mathsf{Pr}_{max}(t, \alpha, \mathcal{M})$, and
2. $\mu_{\approx}(s, \mathcal{M}) \leq \mathsf{Pr}'_{max}(t, \tau, \mathcal{M}, \mathcal{M}')$.

It is easy to see that given a partition $\mathcal{W}$ of a set S, if $\mathcal{W}$ is coarser than $S/\approx$ then there exists a splitter $(\mathcal{C}, \alpha, \mathcal{M})$ of $\mathcal{W}$. Furthermore, given the previous result, $\mathsf{Refine}(\mathcal{W}, \mathcal{C}, \alpha, \mathcal{M})$ is strictly finer than $\mathcal{W}$ and coarser than $S/\approx$. Finally, if no splitter exists for partition $\mathcal{W}$ we may conclude that $\mathcal{W} = S/\approx$. The algorithm for weak bisimulation follows.

Algorithm for computing weak bisimulation equivalence classes

Input: $(S_n, S_p, Act, \longrightarrow_n, \longrightarrow_p, s_0)$
Output: $(S_n \cup S_p)/\approx$
Method: $\mathcal{W} := \{S\}$;
$(\mathcal{C}, \alpha, \mathcal{M}) := \mathsf{FindSplit}(\mathcal{W})$;
`while` $\mathcal{C} \neq \emptyset$ `do`

$\mathcal{W} := \mathsf{Refine}(\mathcal{W}, \mathcal{C}, \alpha, \mathcal{M})$;
$(\mathcal{C}, \alpha, \mathcal{M}) := \mathsf{FindSplit}(\mathcal{W})$
od
return $\mathcal{W}$

It is based on the procedure $\mathsf{FindSplit}(\mathcal{W})$ which, given a partition $\mathcal{W}$, it isolates and returns a splitter $(\mathcal{C}, \alpha, \mathcal{M})$ of $\mathcal{W}$, if one exists, and a triple $(\emptyset, \alpha, \mathcal{M})$, otherwise. It achieves this by considering each $\alpha \in Act$ and each $\mathcal{M} \in \mathcal{W}$ and computing $\mu_{\mathcal{W}}(s, \mathcal{M})$, $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M})$ and $\mathsf{Pr}'_{max}(s, \tau, \mathcal{M}, [s]_{\mathcal{W}})$ for all $s \in S$, thus determining a splitter if one exists. To compute maximum probabilities, $\mathsf{FindSplit}(\mathcal{W})$ makes use of the functions $\mathsf{FindMax}(s, \alpha, \mathcal{M})$ and $\mathsf{FindMax}'(s, \tau, \mathcal{M}, [s]_{\mathcal{W}})$, responsible for computing probabilities $\mathsf{Pr}_{max}(s_0, \alpha, \mathcal{M})$ and $\mathsf{Pr}'_{max}(s_0, \tau, \mathcal{M}, [s_0]_{\mathcal{W}})$ respectively. We describe the former, computation of the latter is similar. To do this, we associate with each state s the variables X^τ_s and, if $\alpha \neq \tau$, X^α_s. The relationship between the variables is given by the following equations:

$$X^\alpha_s = \begin{cases} \sum\limits_{s \xrightarrow{\pi}_p s'} \pi \cdot X^\alpha_{s'}, & s \in S_p \\ \max(\{X^\tau_{s'} \mid s \xrightarrow{\alpha}_n s'\} \cup \{X^\alpha_{s'} \mid s \xrightarrow{\tau}_n s'\}), & s \in S_n \end{cases}$$

$$X^\tau_s = \begin{cases} 1, & s \in \mathcal{M} \\ \sum\limits_{s \xrightarrow{\pi}_p s'} \pi \cdot X^\tau_{s'}, & s \in S_p - \mathcal{M} \\ \max\limits_{s \xrightarrow{\tau}_n s'} X^\tau_{s'}, & s \in S_n - \mathcal{M} \end{cases}$$

We can find a solution for this set of equations by solving a linear programming problem. More precisely, for all equations of the form $X = \max\{X_1, \ldots X_n\}$, we introduce the set of inequations $X \geq X_i$ and we minimize the function $\sum_{s \in \mathsf{S}} X^\alpha_s + X^\tau_s$. Using algorithms based on the ellipsoid method, this problem can be solved in time polynomial to the number of variables (see, e.g. [20]).

Given a solution to this problem we let $\mathsf{FindMax}(s_0, \alpha, \mathcal{M}) = X^\alpha_{s_0}$ and claim that $X^\alpha_{s_0} = \mathsf{Pr}_{max}(s_0, \alpha, \mathcal{M})$. The correctness of this claim can be proved by appealing to Theorem 3, according to which, $\mathsf{Pr}_{max}(s_0, \alpha, \mathcal{M})$ can be achieved by a determinate scheduler. In particular, we make use of the following observation: since a determinate scheduler $\sigma \in \mathsf{DSched}(S)^\alpha$ schedules partial computations with trace either τ or α, it can be viewed as either (1) a simple scheduler (if $\alpha = \tau$) or (2) the concatenation of two simple schedulers (if $\alpha \in L$), the first responsible for scheduling a transition ending with $\xrightarrow{\alpha}_n$, and the second responsible for scheduling invisible transitions to reach $\mathcal{M}$. Thus, according to the equations above, X^α_s and X^τ_s correspond to $\mathsf{Pr}_{max}(s, \alpha, \mathcal{M})$ and $\mathsf{Pr}_{max}(s, \tau, \mathcal{M})$, respectively.

Therefore, assuming that the size of Act is constant and the size of $S_n \cup S_p$ is N, we have the following theorem.

Theorem 5. The above algorithm for computing weak bisimulation equivalence classes can be computed in time polynomial in N.

We point out that a more efficient formulation of the algorithm is possible, which avoids unnecessary recomputation of probabilities and in the searching of splitters. Such concerns will be relevant in the implementation of the algorithm.

6 Concluding Remarks

In this paper we have defined the notion of weak bisimulation for Labeled Concurrent Markov Chains. We have developed a method for deciding weak bisimulation and presented an algorithm which computes weak bisimulation equivalence classes with polynomial-time complexity in the size of the transition system. Due to the generality of our framework, our results can be adopted to other models in which nondeterminism and probabilistic behavior co-exist.

Although not described in the paper, we have also investigated the relationship of the proposed definition with existing proposals of weak bisimulation [25]. In particular, we have shown, that when restricted to a fully probabilistic model the definition presented here coincides with that of [3]. Furthermore, we have compared the definition we propose to that defined by [29] for probabilistic automata. A probabilistic automaton is an automaton whose states allow the nondeterministic choice among a number of probability distributions, each of which involves probabilistic transitions associated with a single action. The weak bisimulation definition of the model, here denoted as $\approx_A$, requires that if automata A_1 and A_2 are weakly bisimilar and A_1 can engage in a transition involving a probability distribution f, then A_2 can engage in a weak transition which combines probability distributions (in a serial manner) to one that is equivalent to f, in the sense that both distributions assign the same probability to the same equivalence classes. For example, automata A and M of Figure 5(a) are bisimilar to each other: A can match every transition of M. In addition, A's ini-

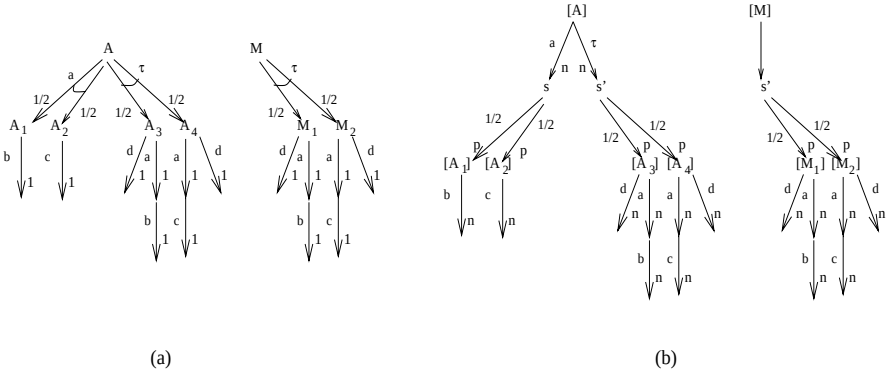

Fig. 5. (a) $A \approx_A M$, (b) $[A] \not\approx [M]$

tial a-labeled transition can be weakly matched by M by combining the initial τ-labeled distribution and consequently the two a labeled ones.

To compare the two bisimulations we have considered translations between LCMC's and probabilistic automata, which describe how LCMC's can be captured by probabilistic automata and vice versa. This is done with the aid of two functions $[\cdot] : \mathsf{Aut} \longrightarrow \mathsf{S}$ and $[[\cdot]] : \mathsf{S} \longrightarrow \mathsf{Aut}$, illustrated in Figure 6, where S and Aut are the sets of LCMC's and probabilistic automata respectively. We may

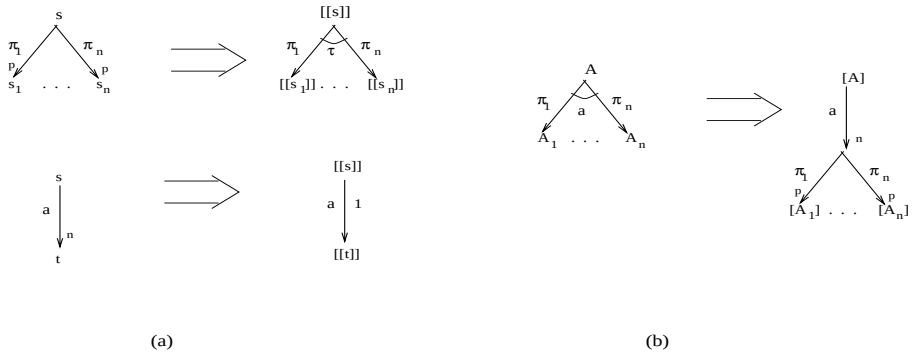

Fig. 6. Translation functions $[[\cdot]]$ (a) and $[\cdot]$ (b)

then prove that if two LCMC's are bisimilar to each other then their translations are also bisimilar with respect to the definition of [29], that is if $s \approx s'$ then $[[s]] \approx_A [[s']]$, except in the case when some LCMC state s performs a probabilistic transition to $s_{[\approx]}$ (see Figure 2). This is due to the fact that [29] does not consider weighted probabilities in the manner of Definition 6(2). However, a similar result does not hold in the opposite direction. That is, it is not the case that if $A \approx_A A'$ then $[A] \approx [A']$. A counter-example is provided by the automata of Figure 5(a). As illustrated Figure 5(b) the LCMC $[A]$ can probabilistically reach state s. However there is no matching transition of $[M]$. This fact is not surprising considering the separation made between nondeterministic and probabilistic states in the LCMC-model.

An additional notion defined in [29] is that of *probabilistic* weak bisimulation, which extends $\approx_A$ by allowing schedulers to range over the set of randomized schedulers. One of the motivations behind this extension was to define a notion that preserves properties expressed in PCTL. We believe that such an approach will not be necessary for the weak bisimulation we propose. The logical characterization of this definition is an issue we are currently investigating.

In related work, we are studying the notion of weak bisimulation within the context of the PACSR process algebra [26], a probabilistic extension of ACSR [22] which is a real-time process algebra that captures the notions of priorities and resources. In this area work is being carried out with aims the axiomatization

of the congruence induced by weak bisimulation, and the extension of existing tools with automated weak-bisimulation checking and state-space minimization.

Acknowledgments: We are grateful to Rance Cleaveland and Scott Smolka for enlightening initial discussions as well as Marta Kwiatkowska and Roberto Segala for interesting comments and suggestions.

References

1. J. Baeten and J. Klop, editors. *Proceedings CONCUR 90,* Amsterdam, volume 458 of *Lecture Notes in Computer Science*. Springer-Verlag, 1990.
2. C. Baier. Polynomial time algorithms for testing probabilistic bisimulation and simulation. In R. Alur and T. Henzinger, editors, *Proceedings of the 8th International Conference on Computer Aided Verification,* New Brunswick, NJ, USA, volume 1102 of *Lecture Notes in Computer Science*, pages 38–49. Springer-Verlag, July/August 1996.
3. C. Baier and H. Hermanns. Weak bisimulation for fully probabilistic processes. In *Proceedings of the 9th International Conference on Computer Aided Verification,* Haifa, volume 1254 of *Lecture Notes in Computer Science*. Springer-Verlag, 1997.
4. C. Baier and M. Kwiatkowska. Automatic verification of liveness properties of randomized systems. In *Proceedings of the* 14^{th} *Annual ACM Symposium on Principles of Distributed Computing,* Santa Barbara, California, August 1997.
5. C. Baier and M. Stoelinga. Norm functions for bisimulations with delays. In *Proc. FOSSACS'00*. Springer-Verlag, 2000.
6. M. Bernardo and R. Gorrieri. Extended markovian process algebra. In U. Montanari and V. Sassone, editors, *Proceedings CONCUR 96,* Pisa, Italy, volume 1119 of *Lecture Notes in Computer Science*, pages 315–330. Springer-Verlag, 1996.
7. A. Bianco and R. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proceedings Foundations of Software Techonology ans Theoretical Computer Science*, volume 1026 of *Lecture Notes in Computer Science*, pages 499–513. Springer-Verlag, 1995.
8. I. Christoff. Testing equivalences and fully abstract models for probabilistic processes. In Baeten and Klop [1], pages 126–140.
9. L. Christoff and I. Christoff. Efficient algorithms for verification of equivalences of probabilistic processes. In K. Larsen and A. Skou, editors, *Proceedings of the 3rd International Workshop on Computer Aided Verification,* Aalborg, Denmark, volume 575 of *Lecture Notes in Computer Science*, pages 310–321. Springer-Verlag, 1991.
10. R. Cleaveland, S. Smolka, and A. Zwarico. Testing preorders for probabilistic processes. In W. Kuich, editor, *Proceedings* 19^{th} *ICALP,* Vienna, volume 623 of *Lecture Notes in Computer Science*, pages 708–719. Springer-Verlag, 1992.
11. C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *Proceedings* 29^{th} *Annual Symposium on Foundations of Computer Science*, pages 338–345. IEEE, 1988.
12. C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. In M. Paterson, editor, *Proceedings* 17^{th} *ICALP,* Warwick, volume 443 of *Lecture Notes in Computer Science*, pages 336–349. Springer-Verlag, July 1990.
13. R. van Glabbeek, S. Smolka, B. Steffen, and C. Tofts. Reactive, generative, and stratified models of probabilistic processes. In *Proceedings* 5^{th} *Annual Symposium on Logic in Computer Science,* Philadelphia, USA, pages 130–141. IEEE Computer Society Press, 1990.

14. H. Hansson. *Time and Probability in Formal Design of Distributed Systems.* PhD thesis, Department of Computer Systems, Uppsala University, 1991. DoCS 91/27.
15. J. Hillston. PEPA: Performance enhanced process algebra. Technical Report CSR-24-93, University of Edinburgh, UK, 1993.
16. B. Jonsson, C. Ho-Stuart, and W. Yi. Testing and refinement for nondeterministic and probabilistic processes. In H. Langmaack, W.-P. de Roever, and J. Vytopil, editors, *Proceedings of the Third International School and Symposium on Formal Techniques in Real Time and Fault Tolerant Systems (FTRTFT'94),* Lübeck, Germany, September 1994, volume 863 of *Lecture Notes in Computer Science*, pages 418–430. Springer-Verlag, 1994.
17. B. Jonsson and K. Larsen. Specification and refinement of probabilistic processes. In *Proceedings* 6^{th} *Annual Symposium on Logic in Computer Science,* Amsterdam, pages 266–277. IEEE Computer Society Press, 1991.
18. C. Jou and S. Smolka. Equivalences, congruences and complete axiomatizations for probabilistic processes. In Baeten and Klop [1].
19. P. Kanellakis and S. Smolka. CCS expressions, finite state processes, and three problems of equivalence. *Information and Computation*, 86(1):43–68, May 1990.
20. H. Karloff. *Linear Programming.* Progress in Theoretical Computer Science. Birkhauser, 1991.
21. K. Larsen and A. Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94, 1991.
22. I. Lee, P. Brémond-Grégoire, and R. Gerber. A process algebraic approach to the specification and analysis of resource-bound real-time systems. *Proceedings of the IEEE*, pages 158–171, Jan 1994.
23. R. Milner. *Communication and Concurrency.* Prentice-Hall International, Englewood Cliffs, 1989.
24. R. Paige and R. Tarjan. Three partition refinement algorithms. *SIAM Journal on Computing*, 16(6):973–989, 1987.
25. A. Philippou, I. Lee, and O. Sokolsky. Weak bisimulation for probabilistic systems. Technical report, University of Cyprus, 1999.
26. A. Philippou, O. Sokolsky, R. Cleaveland, I. Lee, and S. Smolka. Probabilistic resource failure in real-time process algebra. In D. Sangiorgi and R. de Simone, editors, *Proceedings CONCUR 98,* Nice, France, volume 1446 of *Lecture Notes in Computer Science*, pages 389–404. Springer-Verlag, 1998.
27. A. Pnueli and L. Zuck. Probabilistic verification. *Information and Computation*, 103:1–29, 1993.
28. R. Segala. *Modelling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 1995.
29. R. Segala and N. Lynch. Probabilistic simulations for probabilistic processes. In B. Jonsson and J. Parrow, editors, *Proceedings CONCUR 94,* Uppsala, Sweden, volume 836 of *Lecture Notes in Computer Science*, pages 481–496. Springer-Verlag, 1994.
30. M. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proceedings* 26^{th} *Annual Symposium on Foundations of Computer Science*, pages 327–338. IEEE, 1985.